Studies in Big Data 195

Sanjeev Patel
Pardeep Kumar
Ghanshyam Singh *Editors*

Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies

Springer

Studies in Big Data

Volume 195

Series Editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland

The series “Studies in Big Data” (SBD) publishes new developments and advances in the various areas of Big Data- quickly and with a high quality. The intent is to cover the theory, research, development, and applications of Big Data, as embedded in the fields of engineering, computer science, physics, economics and life sciences. The books of the series refer to the analysis and understanding of large, complex, and/or distributed data sets generated from recent digital sources coming from sensors or other physical instruments as well as simulations, crowd sourcing, social networks or other internet transactions, such as emails or video click streams and other. The series contains monographs, lecture notes and edited volumes in Big Data spanning the areas of computational intelligence including neural networks, evolutionary computation, soft computing, fuzzy systems, as well as artificial intelligence, data mining, modern statistics and Operations research, as well as self-organizing systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

The books of this series are reviewed in a single blind peer review process.

Indexed by SCOPUS, EI Compendex, SCIMAGO and zbMATH.

All books published in the series are submitted for consideration in Web of Science.

Sanjeev Patel · Pardeep Kumar · Ghanshyam Singh
Editors

Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies

Springer

Editors
Sanjeev Patel
Department of Computer Science
and Engineering
National Institute of Technology Rourkela
Rourkela, Odisha, India

Pardeep Kumar
Department of Computer Science
and Engineering
Jaypee University of Information
Technology
Solan, Himachal Pradesh, India

Ghanshyam Singh
Department of Electrical and Electronic
Engineering Science
University of Johannesburg
Johannesburg, South Africa

ISSN 2197-6503 ISSN 2197-6511 (electronic)
Studies in Big Data
ISBN 978-3-032-22248-0 ISBN 978-3-032-22249-7 (eBook)
https://doi.org/10.1007/978-3-032-22249-7

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

If disposing of this product, please recycle the paper.

Preface

The rapid evolution of *Cyber-Physical Systems (CPS)*, which tightly integrates computation, networking, and physical processes. It has transformed the technological landscape across sectors such as manufacturing, healthcare, energy, transportation, and defence. The major role of this transformation lies the critical role of communication networks that enable seamless interaction between physical devices and computational systems. However, CPS applications become more complex, there is an urgent need to develop and deploy *next-generation networking paradigms* that can meet stringent requirements for reliability, latency, scalability, and security. This book *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, will focus on redefining the communication foundations of CPS. It brings together leading-edge contributions in three major pillars: (1) **Transmission Control Protocol (TCP)**: Although TCP remains a cornerstone of internet communication, traditional TCP mechanisms face limitations in high-speed, real-time CPS environments. This book will explore the challenges of TCP performance in CPS and discuss innovative enhancements, congestion control algorithms, and adaptations tailored for multi-path, delay-sensitive, and resource-constrained systems. (2) **Software-Defined Networking (SDN)**: SDN offers a paradigm shift by decoupling control and data planes, enabling programmable and adaptive network management. (3) **Emerging Technologies**: Beyond TCP and SDN, CPS networks are increasingly embracing technologies such as 5G/6G, edge computing, Time-Sensitive Networking (TSN), and artificial intelligence-driven network optimization. The chapters in this book span theoretical models, experimental designs, protocol innovations, and real-world case studies. Collectively, they provide valuable insights for researchers, practitioners, and students engaged in networking, control systems, embedded computing, and interdisciplinary CPS domains.

Outline of the Book

This book elaborates on the areas of computer networks, cloud computing, and their emerging applications including security aspect. A comprehensive discussion of TCP congestion control and its applicability on vehicular ad-hoc networks (VANETs) are well presented. Further, it explores about Software-defined networking (SDN) and various promising use-cases of edge computing in real life. The book is written for a broad readership, including experts working in the fields of TCP modelling, SDN, edge computing, network security, etc. It gives useful insights into the risks and challenges that occurred in Internet, SDN, and cloud computing. This book is helpful to the reader who wants to learn about working of TCP congestion control algorithms on the Internet and VANET. It gives full insight into the problems of edge computing and SDN. A brief and orderly introduction to the chapters is provided in the following.

- Provides an extensive overview and state-of-the-art enabling technology that can be put into practical use in order to assist for industrial revolution.
- Explored the foundation for understanding Transmission Control Protocol (TCP) behavior in modern high-speed networks with its potential applications and open research challenges.
- Presented the role of artificial intelligence as a potential key component for several emerging use cases including VANET and smart and connected healthcare.
- Explored hybrid congestion control algorithms that combine traditional and AI-based approaches which will outperform existing methods in diverse network conditions
- Introduces intent-based networking as an evolution of SDN for autonomous IoT management with a conceptual framework that enables high-level policy specification and automatic enforcement.
- Addresses recent advancements, along with relevant prospects on opportunities and challenges with potential research direction in the development of Transmission control protocol.

The topics covered in the book are concerned with cyber physical systems and their development for better human beings and industrial revolution. In this way it will attract various stages of readers as given below:

(1) This book will assist researchers working in academia and industry in conceptualizing the development of next generation's communication networks.
(2) It encourages researchers in academia and industry to learn more in the evolving field of cyber physical systems with their emerging use cases.
(3) This book brings together a comprehensive collection of significant research works to fulfill the pressing demand for next generation networking solutions which ensure reliability, scalability, low-latency and security from the perspectives of IoT and artificial intelligence.
(4) This book focuses to compiling cutting edge research on transmission control protocol, software defined networking and emerging communication and

computing technologies for fulfilling the demand industrial revolution such as Industry 4.0/Industry 5.0 as well as sustainable development goals.

This book is intended for senior undergraduate and graduate students working in electronics/electrical/computer and communications engineering. It also be used as a reference book for engineering professionals and scientists working in academia and industry. In this book, we have tried to cover most representative achievements, with emphasis on thoughts, philosophy, and methodology for the cyber physical systems which are reshaping industrial revolution such as healthcare, manufacturing and defense by integrating physical process, communication, computing and networking. Indeed, existing knowledge is undoubtedly important. However, in this book, we have provided potential thoughts and methodology which supports in creating new knowledge for the future generation high speed networks. The book contains twenty-one chapters.

Chapter 1 lays the foundation for understanding Transmission Control Protocol (TCP) behavior in modern high-speed networks. It introduces analytical and experimental TCP modeling techniques that help to evaluate congestion control and throughput dynamics. By addressing the limitations of classical TCP in gigabit and next-generation networks, the chapter sets the stage for systematic network optimization and protocol evolution.

Chapter 2 presents a structured overview of the evolution of TCP congestion control mechanisms. It discusses loss-based, delay-based, and hybrid approaches while highlighting their relevance across diverse networking environments. The chapter also explores emerging trends such as machine learning, IoT-aware congestion control, and cross-layer optimization, offering insight into the future trajectory of TCP.

Chapter 3 introduces Multipath TCP (MPTCP) as an advanced transport-layer solution designed to address the limitations of traditional TCP in heterogeneous networks. It explains the motivation, design principles, and operational challenges of MPTCP. Key congestion control algorithms are discussed to provide insight into fairness, efficiency, and stability in multipath environments.

Chapter 4 provides a comprehensive introduction to vehicular ad-hoc networks (VANETs), emphasizing their role in Intelligent Transportation Systems. It outlines VANET architecture, communication models, and key performance challenges. The chapter also highlights security, privacy, and scalability issues while pointing toward future integration with AI, edge computing, and next-generation communication technologies.

Chapter 5 presents a federated learning–based hybrid intrusion detection framework for Software Defined Networking (SDN) environments. While SDN offers centralized control and programmability, it is vulnerable to advanced security threats and limitations of traditional centralized intrusion detection systems. To address these challenges, a distributed Federated Learning model integrating Logistic Regression, KNN, Random Forest, MLP, and XGBoost are studied, with class imbalance handled using SMOTE.

Chapter 6 examines the performance limitations of TCP in highly dynamic vehicular environments. It evaluates classical and modern TCP variants and discusses why conventional congestion control mechanisms fail in VANETs. The chapter further explores adaptive solutions, cross-layer approaches, and emerging paradigms to enhance reliable data delivery in vehicular networks.

Chapter 7 provides a systematic review of post-2020 research in VANETs, focusing on intelligence and security enhancements. It surveys recent advancements in routing, intrusion detection, and communication technologies. The chapter also identifies unresolved challenges and proposes future research directions toward scalable, explainable, and privacy-preserving vehicular networks.

Chapter 8 offers a unified perspective on various forms of ad hoc networks, including MANETs, VANETs, FANETs, SANETs, and underground networks. It discusses their architecture, applications, routing protocols, and security concerns. By consolidating diverse subdomains, the chapter serves as a reference guide for current challenges and future research opportunities.

Chapter 9 introduces Software-Defined Networking (SDN) as a transformative networking paradigm. It explains SDN architecture, components, and interfaces while highlighting its applications across data centers, IoT, cloud computing, and telecommunications. The chapter also discusses deployment challenges and real-world adoption scenarios.

Chapter 10 explores the role of SDN in addressing scalability, security, and management challenges in IoT ecosystems. It discusses SDN-IoT architecture, controller-based security enforcement, and policy-driven networking. The chapter highlights how SDN enhances flexibility, centralized control, and secure communication in large-scale IoT deployments.

Chapter 11 introduces intent-based networking as an evolution of SDN for autonomous IoT management. It presents a conceptual framework that enables high-level policy specification and automatic enforcement. By integrating NLP, policy engines, and self-verification mechanisms, the chapter outlines a shift toward ethical, adaptive, and human-aware networking.

Chapter 12 reviews the integration of SDN with cloud computing to enhance network agility and resource utilization. It discusses SDN-enabled cloud architecture, scheduling strategies, and load-balancing mechanisms. The chapter also highlights open challenges and future research directions for intelligent cloud infrastructures.

Chapter 13 focuses on load balancing in cloud-fog-based IoT systems through efficient virtual machine migration. It presents a performance-driven migration strategy and compares it with traditional scheduling algorithms. The chapter demonstrates how adaptive VM migration improves execution time, makespan, and system fairness.

Chapter 14 examines adaptive workload management strategies for cloud-edge ecosystems. It highlights containerized deployment, lightweight orchestration, and autoscaling mechanisms for data-intensive applications. The chapter emphasizes efficient resource utilization for distributed computer-vision and federated learning workloads.

Chapter 15 explores the convergence of machine learning and edge computing for sustainable next-generation networks. It discusses intelligent resource allocation, energy-aware optimization, and real-time analytics. The chapter highlights application scenarios and outlines future challenges in building green, intelligent communication systems.

Chapter 16 presents a comparative evaluation of AI-based intrusion detection techniques for modern networks. It contrasts traditional machine learning models with deep learning approaches using benchmark datasets. The chapter demonstrates the superiority of deep learning in detecting complex and evolving cyber threats.

Chapter 17 introduces a zero-trust, blockchain-anchored security architecture for IoT-edge cyber-physical systems. It discusses decentralized identity, smart contracts, and lightweight consensus mechanisms. The chapter provides practical design guidelines for secure, low-latency, and auditable CPS deployments.

Chapter 18 presents the Digital Trust Chains model for secure and inclusive financial transactions. It integrates CPS, SDN, and blockchain technologies to strengthen digital trust. The chapter offers a policy-technology roadmap inspired by real-world payment systems in developing economies.

Chapter 19 proposes a decentralized portfolio and identity management framework using blockchain and Web3 technologies. It discusses system architecture, smart contracts, and decentralized storage. The chapter highlights how self-sovereign identity models eliminate intermediaries and enhance trust.

Chapter 20 investigates routing attacks in IoT networks using the Contiki-Cooja simulator. It analyzes the impact of blackhole and flooding attacks on network performance. The chapter also introduces cryptographic countermeasures to enhance data security.

Chapter 21 reviews the role of cyber-physical systems (CPS) integrated with AI and ML in modern healthcare systems. It discusses real-time monitoring, predictive analytics, and personalized care. The chapter highlights challenges and future directions toward intelligent, secure, and scalable healthcare infrastructures.

In summary, the book provides a unified view of the state-of-the-art sustainable smart city, which should be accessible to the readers with basic knowledge emerging technology and their role to accomplish sustainable smart city. The readership may find the rich set of references in each of the chapters very useful. The authors have performed a good job by providing a concise summary of all the chapters at the preface of the book. We would strongly recommend the book to senior undergraduate/graduate students, researchers and engineers working or intending to work in the area of sustainable smart city and emerging technologies required to establish sustainable smart city. Although numerous journal/conference publications, tutorials, and books on the sustainable smart city have been published in the last few years, the vast majority of them focus on the various components of the smart city. However, this book distinguishes itself from the existing prosperous literature on sustainable smart city and its enabling technologies.

Special Acknowledgements

The editors of this book are greatly indebted to numerous authors/colleagues for their valuable contribution during the entire period of manuscript preparation. Their immense patience and unconditional support have played an invaluable role throughout the writing of this manuscript. The authors would not justify their work without showing gratitude to their family members who have always been the source of strength to tirelessly work to accomplish the assignment. The first editor gratefully acknowledges the authorities of National Institute of Technology, Rourkela, India for their kind support to come up with this book. The second editor gratefully acknowledges the authorities of Jaypee University of Information Technology, Waknaghat, Solan, Himachal Pradesh, India for their kind support to come up with this book. The third editor gratefully acknowledges the authorities University of Johannesburg, South Africa for kind support to come up with this book. We would also like to thank publishers at Springer, in particular Janusz Kacprzyk, Thomas Ditzinger, Sylvia Schneider and Saranya Sakkarapani for their helpful guidance and encouragement during the creation of this book. In the last but not least, We thank the ultimate source of energy of every particle in the universe, the Almighty, for giving the enough energy and strength to complete the work.

Rourkela, India — Sanjeev Patel
Solan, India — Pardeep Kumar
Johannesburg, South Africa — Ghanshyam Singh

Contents

1 TCP Modelling and Its Application Under High-Speed Networks 1
Reetu Rani Pradhan, Sudarshan Bera, and Sanjeev Patel

2 Evolution and Future Trends in TCP Congestion Control 23
Subhra Priyadarshini Biswal and Sanjeev Patel

3 Introduction to Multipath TCP Congestion Control Algorithms 45
Subhra Priyadarshini Biswal and Sanjeev Patel

4 Introduction to VANET: Architecture, Issues, and Challenges 61
Aditya Dev Mishra, Arshad Husain, and Hemant Kumar Saini

5 Analysis of Hybrid FL Models for Intrusion Detection in Decentralized Software-Defined Networks 77
Aparna Kumari, Pooja Chaturvedi, and Deepika Bishnoi

6 Performance Study of TCP in Vehicular Ad Hoc Networks 93
Prangya Priyadarshini, Putcha Sai Kiran, and Arun Kumar

7 Towards Intelligent and Secure Vehicular Ad Hoc Networks: A Systematic Review of Post-2020 Research 117
Kaushik Adhikary, Bidisha Bhabani, and Tanaya Das

8 Navigating Ad Hoc Networks: Current Challenges and Future Perspectives 137
Aradhana Behura, Pabitra Mohan Khilar, and Abir Sen

9 Application of Software-Defined Networks 181
U. Prabu, N. Malarvizhi, V. Geetha, and Anand Nayyar

10 SDN in Internet of Things (IoT) 211
Umesh Sahoo and Priyanka Kujur

11 Policy-Driven Intent-Based Networking in SDN-Enabled IoT: A Paradigm Shift in Autonomous Network Management 243
Ankur Sisodia, Swati Vishnoi, and Ajay Kumar Yadav

12 SDN in Cloud Computing 265
Akanksha Tandon, Kiran Nagare, and Sanjeev Patel

13 An Efficient Virtual Machine Migration Scheme for Load Balancing in Cloud-Fog Based IoT Systems 283
Sourav Kumar Bhoi and Kalyan Kumar Jena

14 Adaptive Workload Management in Cloud and Edge Environments 303
D. Nandhini and M. Brindha

15 Machine Learning and Edge Computing Synergies for Next-Generation Sustainable Networks 323
Dinesh Sahu, Nidhi, Shiv Prakash, Amira Fauad, and Bhupendra Kumar

16 Network Intrusion Detection Using AI: A Comparative Study and Evaluation 343
Binayak Koirala, Ishika Adhikari, Aradhana Behura, Abir Sen, and Pabitra Mohan Khilar

17 Blockchain-Anchored Security for IoT–Edge Cyber-Physical Systems: A Zero-Trust Architecture with Decentralized Identity and Smart Contracts 367
Rashmi Kumari and Raghwendra Kishore Singh

18 Digital Trust Chains: A CPS–SDN Integrated Framework for Inclusive and Trust-Based Financial Transactions 407
M. Zaheer Ahmed and Ranjan Kumar

19 Decentralized Identity and Authentication Framework for Self-Sovereign Digital Portfolios in Web3-Enabled Cloud-Edge Networks 429
Kuldeep Pal, Lovish Jain, Aklavya, and Renu Mishra

20 Analysis of Attacks in IoT Network Using Contiki-Cooja Simulator 457
Bhawna Sharma, Shrey Gupta, and Rahul Johari

21 Role of Cyber-Physical Systems in Healthcare Using Artificial Intelligence and Machine Learning 489
Ranjan Kumar and Sanjeev Patel

Chapter 1
TCP Modelling and Its Application Under High-Speed Networks



Reetu Rani Pradhan, Sudarshan Bera, and Sanjeev Patel

Abstract The Transmission Control Protocol (TCP) is provided with the prime base infrastructure to support the trustful communication process on the Internet. The swift development of very high-speed networks ranging from Gigabit (Gb) Ethernet networks to next generation 5G and 6G cellular networks and data centre networks reveals the limitations and shortcomings of the traditional congestion and throughput control mechanism in the TCP protocol. Modelling of the TCP provides analytical and simulation-driven platforms to understand and forecast the development and optimization of TCP protocols on diverse high-speed networks. This chapter progresses to the development stages of TCP modelling using logical and experimental techniques to primarily concentrate on the development and optimization stages of TCP protocols using logical and experimental TCP modelling techniques.

Keywords TCP modelling · High-speed networks · Congestion control · TCP variants · Throughput analysis

1.1 Introduction

The Transmission Control Protocol (TCP) has been the primary transport-layer protocol of the Internet for more than 40 years. Initially designed for reliable, connection-oriented communication between hosts, TCP has continually evolved to address the dual challenges of congestion and scalability. The earlier Internet, which has relatively low-speed, wired connections and average traffic volumes, required

R. R. Pradhan (✉) · S. Bera · S. Patel
Department of Computer Science and Engineering, National Institute of Technology Rourkela, Rourkela, Odisha, India
e-mail: 525cs1015@nitrkl.ac.in

S. Bera
e-mail: 925cs5005@nitrkl.ac.in

S. Patel
e-mail: patels@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_1

only basic congestion avoidance mechanism. However, as networks have grown in high speed, scale, and diversity, TCP's congestion control tools have experienced significant refinement. Today, TCP remains to serve as the backbone of the Internet communications, finally its efficiency in modern high speed, wireless, and multipath surroundings relies heavily on accurate modelling and careful algorithm design. For the early Internet, with moderate volumes of traffic traversing relatively homogeneous network paths, simple congestion avoidance strategies sufficed. However, the unprecedented expansion of the Internet, along with the introduction of high-bandwidth links, wireless access networks, and mobile devices, shed light on several weaknesses in the classical variants of TCP, such as Tahoe and Reno. These challenges paved the way for the development of more advanced congestion control techniques such as NewReno, Westwood, Veno, BIC, CUBIC and most recently, model-driven approaches like BBR [1–3]. Despite the existence and spread of alternative transport protocols, TCP remains the prevalent Internet traffic solution based on its reliability and adaptability. The reason for its long-term resilience can be credited to efforts in optimizing congestion control algorithms and modelling in understanding and optimizing its performance under different network scenarios [4, 5]. Following this trend of research, it was noted by researchers about the limits of loss-notification algorithms such as Reno in a diverse environment, especially when wireless networks involve random losses that are uncorrelated to congestions. To overcome this problem, Veno was proposed as an enhancement of Reno [6]. The authors' contributions highlighted the necessity for adaptable approaches to TCP, especially when dealing with both wired and wireless networks, as vision-less, reduction policies based on reducing the congestion window (cwnd) based on non-congestion losses will ultimately worsen network performance. The importance of the analytical modelling in this process of improvement was later elucidated by a throughput model for the TCP Veno algorithm, this model was an efficient way of analysing the throughput of Veno's TCP algorithm [7].

The works recommend the importance of modelling as a process of validation as well as guidance for the development of congestion control algorithms that are better suited for the modern Internet environment [6, 7]. It ensured that research on TCP models covered multi-path wired and wireless networks. This includes surveys such as the one in which the author described the evolution of host-to-host congestion controls ranging from traditional solutions such as Tahoe and Reno to high-speed variants such as High-Speed TCP, BIC, and CUBIC [8]. In this survey, the author placed the problem of high-speed networks within the context of traditional solutions that are typically too conservative for complete utilization. The survey in further reflected on the issue of host-to-host congestion control and placed it in the evaluation of Internet scalability and stability [9]. Congestion control has emerged as a most important component of TCP and has a direct impact on network stability and efficiency. In this regard, TCP congestion control algorithms control the transmission rate dynamically depending on the network situation, mainly measured by the loss or delay of packets. Analytical model studies have been the backbone for developments in TCP technology [5, 10, 11]. TCP modelling has made significant progress through the establishment of a simple and powerful throughput model for

TCP Reno [12, 13]. This work succeeded in defining a closed formula that relates TCP throughput to several key network characteristics, like round-trip time (RTT), packet loss rate, and timer expiration. This novel work provided a powerful platform that was utilized not only as a theoretical channel to understand TCP performance but also a useful tool for analysing, validating, and comparing protocols. As networking environments became increasingly heterogeneous, especially with the deployment of wireless links, it was realized that loss-based congestion control algorithms, like Reno, which view every packet loss as a congestion signal, had some serious drawbacks. TCP Veno was thus developed and proposed as an improvement that could differentiate between congestion-induced losses and random wireless losses [6]. Later, analytical models about TCP Veno also illustrated how modelling can be effective in explaining observed performance gains, as well as informing protocol parameter optimization [7, 14, 15]. Such modelling is not limited to single-path wired networks. Surveys and analytical studies have documented the evolution of congestion control from traditional algorithms to high-speed variants such as BIC and CUBIC, which are designed to better utilize high bandwidth-delay product (BDP) networks [8, 9]. Very recently, with the proliferation of network devices capable of simultaneously using multiple heterogeneous network paths, modelling efforts have also started to address such multipath environments. Analytical modelling of Multipath TCP (MPTCP) has indeed been useful for control parameter tuning and performance prediction in complex multi-interface scenarios [14–17]. In modern high-speed networks like optical transport networks, data centre networks, and next-generation 5G and 6G networks, accurate modelling of TCP is essential. Schemes like slow start, capacity underutilization, lack of fairness among flows, and instability due to a competing flow of data interact greatly with high-speed networks and are well addressed through mathematical modelling and the development of congestion control algorithms [18].

1.2 Fundamentals of TCP and Congestion Control

The evolution of TCP congestion control has been a main factor in the strength and scalability of the Internet. In the late 1980s, the Internet experienced severe congestion collapses due to unreserved traffic growth, which highlighted the urgent requirements for mechanisms to control sending rates and avoid overburdening routers. Since then, a series of congestion control algorithm has been developed, improving its predecessors to adapt to new network demands, technologies, and application requirements. The history of TCP congestion control can be viewed as a progressive journey from loss-based algorithms to delay-based, hybrid, high-speed, and finally, model-driven and intelligent approaches.

1.2.1 TCP Tahoe

TCP Tahoe represents the first systematic implementation of congestion control mechanisms within TCP. It introduced three fundamental algorithms: Slow Start, Congestion Avoidance, and Fast Retransmit. Tahoe was developed under the assumption that packet loss is a reliable indicator of network congestion, an assumption which held approximately in early wired Internet environments [10, 19]. The congestion control of TCP Tahoe begins with Slow Start, where the cwnd grows exponentially up until it reaches a threshold called slow start threshold. Once this threshold is reached, the mechanism enters Congestion Avoidance, where the cwnd increases linearly to probe for more available bandwidth. Detection of packet loss by timeout or duplicate acknowledgments results in a reset of cwnd to one maximum segment size and re-enters Slow Start (SS) in Tahoe. From the control-theoretic perspective, Tahoe operates as a conservative feedback control with stability preferred over efficiency. However, while quite successful in avoiding congestion collapse, it results in significant degradation in throughput, especially in networks with large bandwidth-delay products stimulated refinements.

1.2.2 TCP Reno

TCP Reno is an extension to Tahoe that incorporates the Fast Recovery mechanism, intended to optimize performance in the presence of non-catastrophic packet loss events. TCP Reno keeps the Slow Start, Congestion Avoidance, and Fast Retransmit mechanisms but does not reset the cwnd on packet loss indicated by triple duplicate acknowledgments [13, 19]. In place of reducing cwnd to one maximum segment size (MSS), TCP Reno decreases cwnd by half and enters Fast Recovery mode where it is still possible for the sender to send new data. This improvement enhances throughput performance for those situations with scattered losses and congested wired networks to some degree [9]. Analytical modelling of TCP Reno was a significant milestone in congestion control research. It developed a closed-form throughput model that relates the performance of TCP Reno in packet loss probability, RTT, and timeout behaviour [12, 13]. This model outlined, for the first time, a quantitative basis for Reno's dynamics and formed the basis for further protocol analysis. However, Reno performs poorly for multiple packet losses within a window and wireless networks where packet losses may not be congestion induced. Control-theoretic analyses also show that the loss based feedback loop of Reno may become overly aggressive in such environment, leading to instability and inefficient network utilization [4, 20].

1.2.3 TCP NewReno

TCP NewReno was developed as a cumulative improvement over Reno, particularly addressing the inefficiencies of Reno in the absence of Selective Acknowledgement (SACK) while coping with multiple packet losses [9, 10]. TCP NewReno enhances the Fast Recovery phase by staying in the Fast Recovery state until all the outstanding packets in the current window are acknowledged. Partial acknowledgments are considered as notifications of further losses, which are marked by acknowledgments that do not acknowledge all the data sent prior to entering the Fast Recovery state. When it receives a partial acknowledgment, NewReno sends the next outstanding packet and stays in the Fast Recovery state.

In this manner, TCP NewReno is able to recover from multiple losses within a single window without having to timeout, which results in a dramatically higher throughput performance over Reno on networks with losses. TCP NewReno, from an end-to-end adaptive control perspective, can be seen as improving the level of feedback available to the window sender, allowing a better inference about the network conditions [4]. In regard to TCP NewReno, it does not modify the loss based congestion avoidance mechanism in TCP but prevents unnecessary SS resets. Despite this, NewReno still uses cumulative acknowledgments and is less efficient than SACK-based options when the loss rate is high or the cwnd is large [5, 21].

1.2.4 TCP SACK (Selective Acknowledgement)

TCP SACK is a substantial improvement over TCP loss recovery mechanisms as it permits receivers to provide explicit information to the sender regarding successfully received data segments. Such improvements have contributed substantially to increased network performance with multiple losses received over TCP connections [10]. When SACK is turned on, the acknowledgement packet contains additional information regarding successful arrival of data at the receiving end in terms of data blocks. This benefit the sender as it enables it to transmit those data segments which are missing. SACK works in concert with existing congestion control algorithms such as Reno or NewReno, improving loss recovery without changing the underlying congestion avoidance mechanism.

As far as control theory is concerned, TCP SACK enhances system observability because it provides more informative feedback, which corresponds to more informed control actions [4]. Modelling confirms that TCP SACK versions converge faster, offer fairness, and realize better link utilization, especially in high BDP and loss networks [21, 22]. SACK is particularly relevant in the context of today's optical backbone, data centre, or wireless networks, where effective loss recovery affects network efficiency and as a consequence, bandwidth and latency. As a result, SACK has been incorporated into virtually all TCP algorithm.

1.2.5 Delay Based Algorithms

While Tahoe and Reno base their algorithms entirely on packet loss, TCP Vegas utilizes a delay-based mechanism to detect congestion through variations in RTTs. TCP Vegas was proactive in its approach in addressing the delay associated with the queues; however, its sensitivity to delay jitter was a setback. Delay-based algorithms infer the occurrence of congestion not on the loss of packets but on the increment in RTT, which is proportional to the queue build up in the router. Delay-based algorithms assume that congestion can be inferred from the growth of queue size before packet losses occur. This perspective aligns with control-theoretic principles, where early detection of congestion enables smoother and more stable regulation of the system [4, 5].

There are several notable delay-based TCP variants include: TCP Vegas, a variant of TCP that estimates the difference between the expected and actual transmission rates to indicate the emerging congestion. As the RTT exceeds a certain limit, it reduces the cwnd in a proactive way to avoid buffer overflow and loss of packets. The delay-based algorithms have the reduced latency because of less queue accumulation, lower packet loss, and increased network stability in less congested networks. Nonetheless, they exhibit fairness concerns when coexisting with loss-based TCP aggressors, which are tend to dominate the bandwidth of the link. Analytical studies show that delay-based controllers are excessively conservative to a certain extent when coexisting with other traffic sources and are thereby underutilized for their capabilities [10, 11]. Hence, delay-based TCP are not prominently used despite their superiority in theory.

1.2.6 Hybrid Algorithms

However, being dependent on either losses or delays has limitations for both approaches. Therefore, hybrid algorithms have evolved to counter such drawbacks. The Westwood TCP algorithm used acknowledgment rates for estimating bandwidth and thus worked well under wireless conditions. The Compound TCP (CTCP) algorithm, set as default for Windows, integrated aggressive bandwidth increases with delay-based components. The hybrid algorithms leverage the loss and delay signals. Hybrid methods for Congestion Control in TCP are multi-input feedback controllers that provide robust performance with varied network conditions [4].

TCP Veno, Compound TCP, and Elastic TCP algorithms measure the delay to distinguish congestive losses from random losses, mostly in wireless environments [6, 23, 24]. The cwnd is changed less aggressively if the losses are assumed as non-congestive. Analytical and stochastic models have indicated that hybrid algorithms entail improved throughput in heterogeneous wired-wireless networks, enhancing fairness over purely delay-based methods, and reduced sensitivities to random events.

Nonetheless, the tuning of parameters can prove to be challenge and end up in instability or fairness issues in high BDP networks [5, 7, 15].

1.2.7 High Speed TCP

With the rise of high-speed and high-latency paths, traditional algorithms could not effectively make use of the bandwidth. However, in 2008, the CUBIC algorithm emerged; it uses a cubic window increase function. It has been the default congestion control algorithm in the Linux kernel and has been fundamental to the traffic of the current Internet backbone. High BDP networks such as optical backbones, data centres, or 5G–6G core networks demonstrate the inefficiency of classical window-based TCP Reno-like congestion control algorithms. Due to packet loss, window increase becomes too slow, whereas multiplicative reduction causes dramatic throughput degradation [8, 9]. High speed versions of TCP are: High-Speed TCP (HSTCP), BIC TCP, CUBIC. CUBIC replaces the linear increase in window size with a cubic function of the time elapsed since the last loss event. CUBIC allows for faster convergence and also remains compatible with Reno flows [8]. From the control theory perspective, high-speed TCP algorithms pose non-linear control functions to achieve faster convergence. A vast number of modelling and simulating works confirm the efficiency of high-speed algorithms in improving utilization on high BDP networks, although fairness and RTT bias become ever-emphasized issues [4, 9].

1.2.8 Model Based Algorithms

Google proposed the congestion control algorithm named BBR (Bottleneck Bandwidth and RTT), which came into effect in 2016. This represented a paradigm shift from reactive approaches to model-based control. The basic idea behind the development of the congestion control algorithm named BBR is to create a model for the network capacity as well as the RTT and transmit data at the optimal rate. Model-based congestion control is a paradigm that differs substantially from loss-based approaches because it models the process in the system. This approach estimates the bandwidth and delay on the link and adjusts the rate based on this information. Modelling is a key aspect approach to control complex systems [4].

The early TCP models, like TCP Reno throughput model developed and proved the strength of analytical modelling in protocol design [12, 13]. In extension, stochastic, diffusion, and hybrid models of TCP modelling came into existence, facilitating scalable analysis of TCP dynamics [11, 20, 22]. Model-based algorithms offer several advantages, including higher throughput with lower packet loss, the use of packet drops as indicators of congestion, and improved performance in shallow-buffer and

wireless environments. However, their effectiveness strongly depends on the accuracy of the underlying models, as inaccurate estimations can lead to suboptimal performance.

1.2.9 BBRv2 and Emerging Enhancements

Although BBR achieved high-throughput gains, fairness with loss-based flows and performance under random loss were challenging for the Satellite and Wireless networks. Further BBRv2, succeeded in efficient coexistence and robustness. Researchers also investigated machine learning-based congestion control-such as PCC, Remy, Aurora and cross-layer feedback mechanisms for further optimization of TCP in 5G/6G networks, IoT devices, and Edge. BBR (Bottleneck Bandwidth and Round-trip propagation time) is the largest model-based congestion controller. It specifically addresses the optimal point and sends along the estimated bottleneck link bandwidth while keeping minimal queues. BBRv2 combines loss and ECN signals into its model, thus making BBR a hybrid model-aware controller. From adaptive control systems, BBRv2 is a multi-signal feedback system dealing with the maximization of throughput and ensuring network stability at the same time. Current research also shows positive results regarding coexistence with Reno/CUBIC traffic and performance in real Internet environments [1, 25].

1.3 Need for TCP Modelling in High-Speed Networks

The growing presence of high-speed networks, such as optical networks, data centre networks, and the next-generation 5G and 6G networks, introduces a number of challenges for TCP performance. Traditional congestion controllers that were designed for past Internet environments cannot effectively use the available bandwidth in high BDP environments. In these environments, inefficiencies in traditional controllers result in poor window progress speed, high convergence delays, and unfairness among different flows. To effectively deal with these problems, accurate TCP models are needed.

TCP modelling serves multiple purposes:

a. **Performance Prediction**: The models lead to closed-form expressions for throughput and delay, thus making it possible to predict the performance of the flows based on the network conditions.
b. **Protocol Evaluation**: Modelling allows designers to assess the performance of different congestion control mechanisms without exhaustive simulations or experiments.
c. **Algorithm Design**: The insights obtained from modelling serve as a guide in the tuning of parameters concerning but not limited to the additive increase step and

multiplicative decrease factor in the variants of TCP, such as Reno, Veno, and MPTCP.

d. **Scalability Analysis**: Models enable testing of TCP behaviour as link capacities and RTT scale upward in high-speed environments.

1.3.1 Mathematical Foundations of TCP Modelling

The classical model gives a basic relationship between the TCP throughput and parameters of the network. In TCP-Reno, the mean stable throughput is estimated by the formula [13]:

$$T_P \approx \frac{1}{RTT\sqrt{\frac{2bp}{3} + T_0 \min\left(1.3\sqrt{\frac{3bp}{8}}\right)p(1+32p2)}} \tag{1.1}$$

where RTT = round-trip time, p = packet loss probability, b = number of packets acknowledged with a single ACK (usually b = 1), and T_0 = retransmission timeout often referred to as RTO. This illustrates the typical inverse square root relationship between throughput and loss probability p. In high-speed communications, which assume low values for p, this relation indicates that small changes in p can lead to a large decrease in throughput.

In a mixed wired and wireless network, packet loss is not always caused by congestion. Implemented TCP Veno, which tries to distinguish congested losses from random wireless losses. In light of this improvement simplified TCP Veno throughput model was developed, as given in [6, 7]:

$$T_{Veno} \approx \frac{(1-\beta)}{RTT \cdot \frac{\sqrt{2bp}}{3}} \tag{1.2}$$

where β represents the proportion of random (non-congestion) losses. In this example, it is shown that Veno has higher throughput than Reno in wireless networks since Veno does not perform multiplicative reduction for losses unrelated to congestion. In high-speed networks with wireless connectivity (for example, 5G backhaul networks), such modelling is very important in understanding and predicting protocol performance behaviour.

1.3.2 MPTCP Modelling

Modern devices often utilize multiple network interfaces simultaneously, such as, Wi-Fi and cellular. In such environments, TCP modelling needs to go beyond the single-path assumptions. Designed a framework for MPTCP modelling, which also covered control parameter tuning across multiple sub flows. The simplified throughput model for MPTCP with n sub flows is given in [16]:

$$T_{MPTCP} = \sum_{i=1}^{n} Ti(RTTi, pi, wi) \tag{1.3}$$

where Ti is the throughput of subflow i, RTTi and pi are the round-trip time and packet loss probability of path i, wi is the cwnd of subflow i. This allows the fine-tuning of the MPTCP settings to achieve maximum overall bandwidth and importantly, fairness between these TCP connections and regular TCP communications.

In high-speed networks, the BDP will become very large: BDP = C·RTT, Where C is the link capacity. To use the bandwidth efficiently, the size of the cwnd must be at least as big as the BDP. If the cwnd is that big, it ensures that cwnd $\geq$ BDP. TCP varients with insufficient window growth quantities ultimately end up consuming less than available capacity. Applicability of the model helps in comprehending the magnitude of the deficiency in scaled versions of TCPs such as High Speed TCP, CUBIC, and Veno [7, 13].

TCP models seem imperative in fast networks based on the dynamics involved in congestion, loss rates, and scaling. Several models such as TCP Veno and TCP Reno presented by Fu et al. [6], Zhou et al. [7], and Padhye et al. [13], respectively, demonstrate the ability of mathematical modeling to describe how TCP protocol works in a multitude of situations. The models are not only accurate in describing TCP behavior, but also prescribe how TCP can change or adjust based on the requirements of future internet infrastructure.

1.4 TCP Modelling Approaches

Modelling TCP is not a single technique but a family of complementary approaches that together provide insight into protocol dynamics, performance limits, and design trade-offs. In this section we present the main modelling paradigms used in the literature, give their mathematical building blocks. Analytical throughput (or "black-box") models aim to produce closed-form expressions that relate average TCP throughput to observable network parameters (RTT, loss probability, RTO, MSS, etc.). These models are compact, easy to use for quick prediction, and are widely used for comparison and parameter tuning.

1.4.1 TCP Reno

The canonical gives an approximation for the average steady-state throughput T (bytes/sec) of a long-lived TCP Reno flow [13]:

$$T_P \approx \frac{MSS}{RTT \cdot \frac{\sqrt{2bp}}{3} + T_0 \cdot \min\left(\frac{1.3\sqrt{3bp}}{8}\right) p(1 + 32p2)} \tag{1.4}$$

where: MSS is the maximum segment size (bytes), RTT = round-trip time, p = packet loss probability, b = number of packets acknowledged by a single ACK (typically b = 1), T_0 = retransmission timeout (RTO).

The model captures two regimes: the loss-driven congestion avoidance behaviour (represented by the $\sqrt{p}$ term) and the occasional timeout penalty (second term). The inverse-square-root dependence on p is the key insight: throughput degrades roughly as $(1/\sqrt{p})$ in the loss-dominated steady state. This has been widely validated and used as a baseline for single-path TCP performance prediction [9, 13].

1.4.2 Veno: Modelling Random vs. Congestion Loss

TCP Veno attempts to distinguish congestion losses from random (wireless) losses [6]. Derived a simple throughput model for Veno by modifying the framework to account the fraction of losses that are non-congestion: Let β denote the fraction (or probability) of losses that are random/non-congestion losses ($0 \leq \beta < 1$). A simplified throughput estimate for Veno can be given as [7]:

$$T_{Veno} \approx \frac{MSS}{RTT \cdot \frac{\sqrt{2b(1-\beta)p}}{3} + \ldots} \tag{1.5}$$

Here the effective congestion loss rate is $(1 - \beta)p$; the RTO/timeout term is similarly modified for the precise form used in their validation. Since TCP Veno avoids a full multiplicative decrease for losses classified as random, the effective congestion loss rate is reduced, leading to higher throughput in mixed wired–wireless environments. Empirical studies confirm that this model accurately captures Veno's performance advantages when β is significant [8].

1.4.3 Fluid-Flow (Continuous) Models and Control-Theoretic Formulations

Rather than treating TCP as a discrete packet process, fluid models approximate TCP window dynamics by continuous differential equations. These models are particularly useful for stability analysis, control design, and insight into transient behaviour. A commonly used fluid approximation for a single Reno flow's congestion window w (t) (in packets) is:

$$\frac{dw(t)}{dt} = \frac{1}{RTT} - \frac{w(t)}{2} \cdot \frac{p(t)}{RTT} \tag{1.6}$$

where (t) denotes instantaneous loss (drop) probability seen by the flow. Explanation: additive increase adds ≈1 packet per RTT (hence 1/RTT); multiplicative decrease occurs with probability p and reduces the window by ≈w/2, so the expected decrease rate is (w/2)·p/RTT. This can be rearranged to show equilibrium w^* when dw/dt = 0

$$\frac{1}{RTT} = \frac{w^*}{2} \cdot \frac{p}{RTT} \tag{1.7}$$

And hence average throughput T≈MSS·w^*/RTT∼MSS/(RTT $\sqrt{p}$), recovering the $1/\sqrt{p}$ scaling intuition [13].

Control-theoretic approaches linearize the nonlinear AIMD dynamics around an operating point and model the network (buffers, routers) as additional dynamic blocks. A typical linearized form relates small perturbations in window ΔW(s) to observed queuing delays and loss signals, enabling stability margin analysis and controller synthesis (e.g., design of AIMD parameters to avoid oscillations) summarizes numerous control perspectives and stability results [14].

1.4.4 Stochastic and Markov Models

Packet arrivals, losses, and timer expirations are probabilistic. Stochastic models capture randomness explicitly using Markov chains, renewal theory, or queuing-theoretic descriptions. They can provide exact (or asymptotic) expressions for distributions of window size, throughput, and delay.

One classical approach models the congestion window W_n at the end of the nth RTT as a Markov chain with transition probabilities determined by the loss process. Balance equations for the stationary distribution π(w) can be written and solved (analytically or numerically) to obtain metrics such as E[W] and the throughput:

$$T = \frac{MSS}{RTT} E[W] \tag{1.8}$$

Markov models are especially useful when loss is correlated (burst losses), or when explicit modelling of retransmission timers and fast-recovery states is needed. Renewal-theoretic techniques treat cycles between loss events as renewal intervals and compute average rewards (bytes sent per cycle) divided by average cycle length to obtain throughput.

1.4.5 QUIC Throughput Modelling

Most analytic models of QUIC **adapt existing TCP throughput models** because QUIC's congestion control (e.g., NewReno or CUBIC) behaves similarly in steady state. For example, when QUIC uses a TCP-like congestion control, a *Mathis-style* throughput model can be adapted.

A commonly used analytic formula for steady-state throughput under loss-based congestion control (adapted from TCP models) is:

$$TP_{QUIC} = \frac{maxQuicPacketSize}{RTT}\sqrt{\frac{3}{2p}} \tag{1.9}$$

where, TP_{QUIC} defines QUIC throughput, maxQuicPacketSize is maximum QUIC payload size, RTT is Round-trip time, p defines packet loss probability. This equation is conceptually identical to TCP throughput models and can be used for QUIC when it implements TCP congestion control such as TCP NewReno or CUBIC in simulations or analysis [26].

1.4.6 MPTCP Models

MPTCP complicates modelling because multiple sub flows share resources and are coupled by the congestion control algorithm. A common modelling decomposition is to treat each sub flow like a single-path TCP and then add coupling constraints that reflect how the multipath scheduler and congestion controller allocate increments.

A baseline model for an MPTCP connection with n sub flows is simply [13]:

$$\mathrm{T_{MP}} = \sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{Ti(RTTi, pi)} \tag{1.10}$$

where $\mathrm{T_i}(\cdot)$ is the single-path throughput model for path i with $\mathrm{RTT_i}$ and loss $\mathrm{p_i}$. This additive model is useful as a first approximation and for parameter tuning.

Practical MPTCP controllers (Linked Increase Algorithm, OLIA, BALIA, etc.) couple the increase term across sub flows to achieve TCP-friendliness and to avoid grabbing unfair share on low-RTT paths. Modelling must therefore include the

coupling rule; mathematically this is typically represented by modifying the AI term for sub flow i

$$\frac{dwi}{dt} = \frac{\alpha(w)}{RTTi} - \frac{wi}{2} \cdot \frac{pi}{RTTi} \tag{1.11}$$

where the coupling function α(w) depends on the vector of subflow windows w use such framework to tune control parameters so that aggregate throughput is maximized subject to fairness constraints [16].

1.4.7 Simulation-Based (Packet-Level) Models

Analytical models make simplifying assumptions and simulate packet-level dynamics and protocol implementations to capture details such as timer granularity, packet scheduling, header effects, and real topology interactions.

In typical modelling workflows, researchers use simulations to validate and calibrate analytical models [7, 13] explore corner cases such as burst losses and correlated cross-traffic, and generate reproducible numerical results [6].

Practical TCP research frequently combines analytical and simulation methods: derive a closed-form model to get intuition and quick predictions, then use targeted packet-level simulations to validate assumptions and to measure quantities that are hard to model analytically (e.g., the value of β—the fraction of random losses—for Veno).

a. Analytical derivation: Use Padhye-style or fluid models to derive performance dependences.
b. Parameter estimation: Use measurements or short simulations to estimate parameters (e.g., loss correlation, RTO distribution, β).
c. Packet-level simulation: Validate the analytical predictions on realistic topologies and traffic mixes.
d. Refinement: Adjust the analytical model (e.g., include timeout penalties, burst loss models) and re-validate.

1.5 TCP Variants for High-Speed Networks

Over the past three decades, TCP congestion control has evolved into a diverse ecosystem of algorithms, each tailored to address specific network challenges such as high bandwidth-delay products, wireless losses, multipath communications, and low-latency applications. To better understand this evolution, researchers have proposed various classification schemes. The tabular comparison presented in Table 1.1. A widely adopted taxonomy categorizes congestion control algorithms according to

Table 1.1 The classification of congestion control algorithms

Type	Examples	Strengths	Limitations
Loss-based	TCP reno, CUBIC	Simple design, well-tested in diverse networks	Performs poorly on high-speed or high-latency paths due to aggressive loss response
Delay-based	TCP vegas	Detects congestion before packet loss	Highly sensitive to network jitter and RTT fluctuations
Hybrid	Compound TCP, TCP westwood	Combines loss and delay signals for balanced performance	Requires careful parameter tuning for different network conditions
Model-based	BBR, BBRv2	Achieves high throughput with low latency	Fairness challenges with competing loss-based flows
ML-based	Remy, PCC, Aurora	Learns and adapts dynamically to network patterns	Needs training data and continuous feedback, may lack explainability

their primary congestion signals and underlying design philosophies, including loss-based, delay-based, hybrid, model-based, and machine learning–driven approaches.

1.5.1 Loss-Based Algorithms

TCP cubic interprets a loss of packets as a sign of congestion. The algorithms, which started from Tahoe and Reno, has come to represent the vast majority of Internet history. Another loss-based algorithm, CUBIC, the default TCP variant on Linux, has an aggressive cubic function that increases in proportion to fast bandwidth.

Strengths: Simple to implement, widely used, performs well in wired networks.

Limitations: Underutilization in high bandwidth-delay paths, performance in wireless/satellite networks which are non-congestion related network link losses.

1.5.2 Delay-Based Algorithms

Delay-based algorithms, led by TCP Vegas, are based on RTT measurements to predict congestion before actually experiencing packet losses. These algorithms optimize the network performance by improving the throughput before any loss takes place.

Advantage: Capable of recognizing congestion earlier, lower queuing delay, better user experience.

Limitations: Very sensitive to jitter and clock errors. Not much usage in heterogeneous Internet environment.

1.5.3 Hybrid Algorithms

Hybrid methods use a combination of various signals, such as loss, delay, or estimates of bandwidth, and provide a more rounded control scheme. TCP Westwood and Compound TCP (CTCP), which regulates the aggressive loss component with the help of delay, are considered under hybrid algorithm. CTCP has become the default TCP variant for Windows operating systems.

Strengths: Well-rounded performance under different conditions.

Limitations: Higher complexity, difficulty in handling hyper parameters, possibility of unfairness in a partially observable environment.

1.5.4 Model-Based Algorithms

Model based solutions can be considered a paradigm shift as they actually model the behaviour of the network. Google's BBR (Bottleneck Bandwidth and RTT) algorithm, which came into 2016, tries to estimate the bottleneck bandwidth and propagation delay in an effort to determine the optimal sending rate.

Advantages: High throughput, low latency, works well in current broadband networks.

Shortcomings: Lack of fairness for conventional loss-based flows, tuning needed for coexistence.

1.5.5 Machine Learning-Driven Algorithms

Recent developments in AI have motivated congestion control algorithms to learn from data. Such algorithms include Remy (MIT), which produces congestion control algorithms through offline learning, PCC (Performance-oriented Congestion Control), which learns through trial and error with real-time learning, and Aurora (Meta/Facebook), which uses deep reinforcement learning.

Strengths: Highly adaptable and capable of being optimized for a variety of applications.

Constraints: Features dependent on the training dataset, increased complexity in terms of computing requirements.

1.6 Applications of TCP Modelling in High-Speed Networks

The Transmission Control Protocol (TCP) remains a crucial component of modern Internet communications, especially when link capacities and traffic dynamics become more significant under high-speed networks. In response to these challenges, a dedicated field of research has evolved based on the modelling of TCP protocol to analyse and optimize the performance of TCP.

The section below discusses critical applications of TCP modelling within high-speed networks, based upon recognized research within this area of study.

1.6.1 Performance Prediction and Throughput Estimation

TCP modelling has a major applications in predicting throughput and overall performances of the network. In high-speed networks, BDP increases, and thus it is essential to forecast the performances of TCP as the cwnd increase.

This model marked the birth of analytical modelling for TCP Reno, as it significantly contributed to the development of equations that link loss rate and RTT to overall throughput. This model has greatly helped network engineers in determining the expected result with different loss rates.

TCP Veno Upgrades: Introduced the concept of TCP Veno to improve the performance of TCP in wireless networks. The models presented in the paper depict the ability of Veno to distinguish congestion loss from random loss. Such models are important in capacity planning, providing quality of service, as well as bandwidth estimation in data centre setup.

1.6.2 Congestion Control Mechanism Design

With TCP modelling, the researchers will able to simulate and analyse various congestion control algorithms under high-speed conditions that may help in deriving new variants or fine-tuning the existing ones. Control-theoretic TCP Analysis: it applied TCP modelling to MPTCP with the aim of performing a fine-tuning of control parameters. This shows how TCP modelling allows adaptive algorithms to dynamically distribute sending rates among the set of available paths, achieving throughput and fairness self-adjustments in high-speed, heterogeneous networks [16].

TCP Veno Modelling and Congestion Avoidance: Veno's method of detecting congestion in wireless networks is emulated by a model, in which it yields high throughput in high-speed wireless networks. Its modelling result gives insight into developing the congestion avoidance mechanism without reduction in the window

that may not be necessary. The models give insight into the behavior of TCP and therefore, offer an understanding on which the engineer can design robust algorithms to yield better link utilization, reduction in retransmission, and fairness [7].

1.6.3 Wireless and Heterogeneous Network Optimization

High-speed networks are likely support a wide range of both wired and wireless environments. High-speed performance modelling of TCP enables the optimization of end-to-end performance over paths that consist of a combination of wired and wireless links. Studies on TCP Veno performance over mixed networks have analyzed its behavior by accounting for the effects introduced by different network conditions. The proposed model is used to optimize protocol behavior in wireless last-mile networks.

Loss Differentiation Using Veno's Model: Highlighted the need for distinguishing between congestion loss and wireless loss. This approach combines these factors in the congestion window control, which enhances end-to-end throughput on high-speed wireless networks with fairness to TCP. This is an important area in mobile edge computing, 5G, and IoT, where link quality varies.

1.6.4 MPTCP Control Parameter Tuning

As high-speed networks become more multipath-based (for instance, having both Wi-Fi and 5G connectivity), MPTCP has become an extension of TCP. Nevertheless, it has complex dynamics associated with simultaneous data streams. TCP model in the context of MPTCP to propose a parameter adaptation technique to achieve a balance between performance and fairness. Their work contributes to an understanding of cwnd behavior and loss allocation [16].

This can become particularly important in the case of cloud data centres, multi-homed clients, or hybrid WANs where the usage of multiple paths can result in bandwidth enhancements if carefully managed.

1.6.5 Congestion Control Evaluation and Standardization

Indeed, TCP modelling is one of the key methods used in evaluating congestion control proposals for potential standardization by organizations such as the IETF. A comprehensive survey on host-to-host congestion control mechanisms is presented in [9]. The authors showed very nicely how modelling supports the comparison of various TCP versions, such as Reno, Cubic, Vegas, and BBR. Models can be used

to derive the properties of convergence, stability, and fairness. Such modelling will enable network designers to choose or derive appropriate variants of TCP that will perform optimally over high-speed backbone networks, satellite links, and applications that involve high volumes of data transfer, such as video transmission or scientific visualization.

1.6.6 Protocol Validation and Real-World Deployment Readiness

In the high-speed networks, the models are utilized to verify the theoretical performance characteristics prior to the deployment of different variants as well as the optimization of parameters. For instance, empirical validation was carried out, which proved its reliability [7]. This become the practice for checking the validity of freshly observed protocol behaviours. In fact, model validation by simulation has proven absolutely indispensable in the development of carrier-grade networks, CDNs, and corporate backbone networks, because it's crucial that the protocols' behavior be completely understood before large-scale deployment.

1.7 Case Studies and Experimental Evaluation

A major area of concern within the realm of TCP carries not only the requirement to develop these mathematical models but also to validate these models experimentally. These experiments would be conducted from the foundations laid by various influential works on TCP model solutions available within the literature. These works would help validate different TCP models theoretically.

The following studies include a wide range of variants of the TCP protocol, such as Reno, Veno, and MPTCP, and illustrate the impact that model-based methodologies can have on protocol designs as presented in Table 1.2.

1.8 Conclusion

Modelling in TCP has been a crucial enabler in the progress that has been made in the development of congestion control algorithms over the years. It started with theoretical models that represented the relationship between throughput, loss, and delay in TCP Reno [13] and progressed further in wireless scenarios using TCP Veno [6, 7]. Modelling in this manner has reinforced the point that precise models play a vital role in identifying the shortcomings in loss-control schemes and in advancing new solutions adapted to different environments.

Table 1.2 Summary of experimental evaluations

TCP variant	Evaluation method	Network type	Key findings
TCP reno	NS + real testbeds	Wired high-speed	Model matches real throughput closely
TCP veno	NS-2 simulations	Wireless access	RTT-based model improves loss handling
TCP veno	Simulation	Mixed wired/wireless	Sustains throughput in wireless errors
MPTCP	Fluid modelling + simulation	Multipath	Model-informed tuning improves fairness
Multiple	Survey and experiments	Heterogeneous	Highlights model gaps in dynamic networks

Over the years, as networks have evolved to become faster and more complex, so has the scope of TCP modelling research to higher-speed links and multipath networks. Notably more recent research on MPTCP modelling and parameter optimization [14, 16], and various surveys of advances related to congestion control for host-to-host networks [9] indicate the continued importance and relevance of theoretical and model-driven approaches to ensure fairness and effectiveness in networks. In summary, TCP modelling research remains a very important link between theory and practice for next-generation networks of the Internet for the future.

References

1. Cardwell, N., Cheng, Y., Gunn, C.S., Yeganeh, S.H., Jacobson, V.: BBR: congestion-based congestion control. Commun. ACM **60**(2), 58–66 (2017)
2. Patel, S., Shukla, Y., Kumar, N., Sharma, T., Singh, K.: A comparative performance analysis of TCP congestion control algorithms: newreno, westwood, veno, bic, and cubic. In: Proceedings of the 2020 6th International Conference on Signal Processing and Communication (ICSC), pp. 23–28. IEEE (2020)
3. Patel, S., Rani, K.: Comparative performance analysis of TCP-based congestion control algorithms. Int. J. Commun. Netw. Distrib. Syst. **17**(1), 61–75 (2016)
4. Houmkozlis, C.N., Rovithakis, G.A.: End-to-End Adaptive Congestion Control in TCP/IP Networks. CRC Press (2017)
5. Lestas, M., Pitsillides, A., Ioannou, P.: Congestion control in computer networks. In: Modeling and Control of Complex Systems, pp. 219–262. CRC Press (2007)
6. Fu, C.P., Liew, S.C.: TCP veno: TCP enhancement for transmission over wireless access networks. IEEE J. Select. Areas Commun. **21**(2), 216–228 (2003)
7. Zhou, B., Fu, C.P., Chiu, D.-M., Lau, C.T., Ngoh, L.H.: A simple throughput model for TCP veno. In: Proceedings of the 2006 IEEE International Conference on Communications, vol. 12, pp. 5395–5400. IEEE (2006)
8. Ha, S., Rhee, I., Xu, L.: CUBIC: a new TCP-friendly high-speed TCP variant. ACM SIGOPS Oper. Syst. Rev. **42**(5), 64–74 (2008)
9. Afanasyev, A., Tilley, N., Reiher, P., Kleinrock, L.: Host-to-host congestion control for TCP. IEEE Commun. Surv. Tutor. **12**(3), 304–342 (2010)
10. Welzl, M.: Network Congestion Control: Managing Internet Traffic. Wiley (2005)

11. Jacobsson, K.: Dynamic Modeling of Internet Congestion Control. PhD diss., KTH (2008)
12. Padhye, J., Firoiu, V., Towsley, D., Kurose, J.: Modeling TCP throughput: a simple model and its empirical validation. In: Proceedings of the ACM SIGCOMM'98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, pp. 303–314 (1998)
13. Padhye, J., Firoiu, V., Towsley, D.F., Kurose, J.F.: Modeling TCP reno performance: a simple model and its empirical validation. IEEE/ACM Trans. Netw. **8**(2), 133–145 (2002)
14. Cheung, C.-C.: Performance analysis of TCP Veno over mixed wired/wireless networks. In: Proceedings of the 2008 14th European Wireless Conference, pp. 1–6. IEEE (2008)
15. Biswal, S.P., Patel, S.: Optimization of Veno parameter based on stochastic approximation: OVeno. Simul. Model. Pract. Theory **142**, 103121 (2025)
16. Aung, H.N., Goto, K., Ohsaki, H.: Modeling multipath TCP for control parameter tuning. In: Proceedings of the 2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 950–953. IEEE (2023)
17. Biswal, S.P., Chatterjee, A., Alanam, K., Patel, S.: Performance analysis of multipath TCP congestion control algorithms. In: Proceedings of the 2025 IEEE 14th International Conference on Communication Systems and Network Technologies (CSNT), pp. 1205–1209. IEEE (2025)
18. Lorincz, J., Klarin, Z., Ožegović, J.: A comprehensive overview of TCP congestion control in 5G networks: research challenges and future perspectives. Sensors **21**(13), 4510 (2021)
19. Padhye, J., Firoiu, V., Towsley, D.: A stochastic model of tcp reno congestion avoidence and control (1999)
20. Hespanha, J.P., Bohacek, S., Obraczka, K., Lee, J.: Hybrid modeling of TCP congestion control. In: International Workshop on Hybrid Systems: Computation and Control, pp. 291–304. Springer, Berlin (2001)
21. Ohsaki, H., Ujiie, J., Imase, M.: On scalable modeling of TCP congestion control mechanism for large-scale IP networks. In: The 2005 Symposium on Applications and the Internet, pp. 361–368. IEEE (2005)
22. Domański, A., Domańska, J., Czachórski, T., Klamka, J., Szyguła, J., Marek, D.: Diffusion approximation model of TCP newreno congestion control mechanism: A. Domański et al. SN Comput. Sci. **1**(1), 43 (2020)
23. Biswal, S.P., Patel, S.: A bandwidth delay product based modified veno for high-speed networks: BDP-VENO. J. Netw. Comput. Appl. **231**, 103983 (2024)
24. Jaiswal, V.K., Patel, S.: A comparative performance analysis of TCP congestion control algorithm: elastic TCP versus e-Elastic TCP. Int. J. Internet Protocol Technol. **16**(3), 161–175 (2023)
25. Biswal, S.P., Barik, N.R., Patel, S.: Comparative analysis of congestion window size and throughput for BBR versus elastic TCP. In: Proceedings of the 2023 IEEE 15th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 14–18. IEEE (2023)
26. Völker, T., Volodina, M.: A QUIC Simulation Model for INET and its Application, IFIP Networking 2020—This Work Builds a QUIC Performance Model in Simulation and Adapts TCP Congestion Control Behavior for QUIC
27. Biswal, S.P., Gupta, S., Kumar, A., Patel, S.: Comparative analysis of compound TCP with various end-to-end congestion control mechanisms. In: Proceeidngs of the 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), pp. 1762–1767. IEEE (2022)
28. Patel, S.: Comparative analysis of cumulative distribution function for TCP congestion window size and triple-duplicate period. Int. J. Commun. Syst. **32**(15), e4114 (2019)
29. Akujobi, F., Lambadaris, I., Makkar, R., Seddigh, N., Nandy, B.: BECN for congestion control in TCP/IP networks: study and comparative evaluation. In: Global Telecommunications Conference, 2002. GLOBECOM'02, vol. 3, pp. 2588–2593. IEEE (2002)
30. Martyshkin, A.I.: Modelling algorithms and protocols for congestion control in TCP/IP networks. In: Proceedings of the 2024 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), pp. 1164–1168. IEEE, 2024.

31. Akujobi, F., Seddigh, N., Nandy, B., Makkar, R., Lambadaris, I.: Congestion control in TCP/IP networks: a combined ECN and BECN approach. In: IEEE Military Communications Conference, 2003. MILCOM 2003, vol. 1, pp. 248–254. IEEE (2003)

Chapter 2
Evolution and Future Trends in TCP Congestion Control



Subhra Priyadarshini Biswal and Sanjeev Patel

Abstract Congestion and the overall efficiency of TCP are managed by the protocol itself through a myriad of different techniques that have developed over the years to decrease congestion through the different measures of TCP performance. These techniques can be classified into three main types: loss-oriented techniques, delay-oriented techniques, and hybrid techniques. The introduction of these improvements to TCP's performance in various network types (wireless networks, wired networks, high-speed link networks) is a clear representation and an evolution of the TCP protocols to adapt to new technologies in the last decade or so. In order to properly manage congestion through its various encoding and decoding mechanisms, it has had to change its approach to managing congestion due to the emergence of new techniques, such as machine learning, cross-layer optimization, and technology-specific congestion control (such as 5G, Internet of Things (IOT), etc.). There has been an increased amount of research on adaptive mechanism(s) that allows for the management of congestion prior to packet loss, and many of these mechanisms use various techniques to create intelligent and context-aware forms of TCP, such as reinforcement learning and neural network-based controllers. These advancements provide TCP with the ability to continue to provide efficient and reliable service to users, as well as an ability to operate in more complicated and expanded network scenarios as they continue to develop and change over time.

Keywords TCP · IoT · Reinforcement learning · Congestion control

S. P. Biswal (✉)
School of Computer Engineering, Kalinga Institute of Industrial Technology, Bhubaneswar, India
e-mail: subhrabiswal94@gmail.com

S. Patel
Department of Computer Science and Engineering, National Institute of Technology Rourkela, Rourkela, Odisha, India
e-mail: patels@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_2

2.1 Introduction

The quick development and advancement of Internet technologies increases the ability for end systems to communicate with one another in an uninterrupted manner, and the transport layer Protocols are designed to assist in managing the transfer of data from one end system to another across the transport layer. The Transmission Control Protocol (TCP) is the most widely used transport layer protocol and is used for the management of the byte stream that connects end systems and ensures that all data is transmitted reliably and error-free [1]. TCP has become the most popular transport protocol on the Internet because of its ability to support multiple types of services (e.g., web browsing, email, file transfer). Researchers are actively looking at new TCP extensions and modifications as ways to increase TCP's performance in heterogeneous and dynamic network environments; however, TCP performance can be heavily impacted by modern high-speed, wireless network technologies. These networks can introduce new challenges, such as variable delays between packets arriving at their destination, packet loss, and variable bandwidth availability [2]. As a result, many TCP variants (i.e., modified versions of TCP) have been developed, each having different ways of handling congestion control (i.e., controlling how many packets TCP sends to the network), flow control (i.e., managing how quickly TCP delivers packets to an application on the receiving end) and loss recovery (i.e., recovering lost packets). The multipath extension of TCP allows for multiple simultaneous flows of information through multiple distinct paths to the destination, thus increasing reliability and throughput. All of these enhancements are focused on optimizing the way we utilize our limited network resources while ensuring fairness between multiple flows of data that are being sent over TCP. As new network applications continue to emerge, understanding how to improve TCP's performance will continue to be an important area for research.

The TCP congestion control mechanism greatly improves the performance of the network as a whole. When network congestion occurs, it means that too much data traffic is coming into the network, more that the network can efficiently process. When there is congestion, packets are queued in a buffer at the bottleneck. When packets are queued in a buffer, the number of packets lost and the amount of throughput produced are reduced [3]. This happens due to an increase in the amount of data being sent through the network, which leads to congestion. This results in increased packet loss and delays, and therefore a reduction in the overall quality of network performance. In order to prevent any of these types of problems from occur, the sender's transmission performance must be controlled with a congestion control mechanism that allows for continued efficient communication. TCP uses a number of different algorithms that allow it to automatically control the flow of data from one application to another based on the current status of the network. These algorithms monitor many different conditions on a network including packet round trip time (RTT),and the amount of recent time since an ACK has been received so that it can determine if a particular channel is congested. Once congestion is detected, TCP reduces its transmitting rate until conditions are back to normal again. TCP maintains a level of balance

between maximum throughput and minimal packet loss. By maintaining this level of balance, TCP also aids in maintaining the stability of the network, reduces latency and increases the reliability of data being delivered.

To avoid congestion problems, TCP uses congestion control policies to dynamically adjust the sending rate by observing the network condition. It helps TCP to maintain the efficient data flow in the network by ensuring fair resource allocation. Congestion control methods are generally categorized into open-loop, closed-loop, end-to-end, and network-assisted mechanisms. The open-loop congestion control takes a proactive approach by implementing various policies regarding retransmission, ACK, congestion window, etc. This approach is suitable for networks with predictable traffic patterns. The closed-loop techniques dynamically handle network congestion using feedback mechanism. This includes various techniques for congestion detection, retransmission, Explicit Congestion Notification (ECN), etc. to effectively manage network traffic. The end-to-end congestion control depends on feedback obtained only from the sender and receiver, without requiring support from intermediate network devices. This approach allows TCP to adjust its transmission rate based on the observed packet loss or delay. This makes TCP suitable for high-speed large-scale networks. The network-assisted congestion control requires information from intermediate devices such as routers and switches. This provides explicit feedback about current network conditions. By combining these techniques, TCP can respond effectively to varying traffic loads and prevent buffer overflow at bottleneck points. The modern TCP variants use these strategies to optimize throughput while minimizing packet loss and delay.

TCP mainly depends on the closed-loop based congestion control approach to manage data traffic. These policies include slow start, additive increase with multiplicative decrease, fast retransmission and quick recovery process. At the beginning of data transmission, TCP enters the slow start phase. In this level, TCP expand the congestion window size in an exponential rate with each received acknowledgment (ACK). When the size of the congestion window is larger than the threshold of slow start, the sender switches to the congestion avoidance phase. During this phase, the window size increases gradually based on the additive increase and multiplicative decrease rule. In the linear increase mode, the congestion window rise at a constant rate in linear manner. It avoid increase exponentially to prevent the network from becoming overwhelmed. When packet loss occurs, TCP immediately enters the multiplicative decrease mode to reduce the transmission rate [4]. When a timeout indicates packet loss, the congestion window is reset to a single Maximum Segment Size (MSS) and the protocol transitions back to the initial slow start phase. In case of packet loss due to 3 duplicate ACK, it immediately reduces the window size by half. After that it enters to the fast recovery phase where for each duplicate ACK, increases the window size by 1 MSS to maintain the flow of data. Once a new ACK is received, the congestion window is deflated back to slow start threshold, and TCP transits to the congestion avoidance phase. The most recent advancement within TCP congestion control methods have an emphasis on enhanced effectiveness, improved reaction and response time (responsiveness) throughout various conditions of networks. Selective Acknowledgments have enabled the sender to re-transmit specifically only

those packets that are lost – thereby saving an enormous amount of excess traffic and thus, promoting effective bandwidth-use. With ECN, the TCP network protocol now has available methods to proactively adjust their sending rate – so they will not inundate their associated networks. By doing this, they reduce delay and give equal access to multiple connections on the same network and platforms.

2.2 Types of TCP Congestion Control Algorithms

TCP congestion control mechanisms try to optimize network performance through dynamic adjustment in transmission rate depending upon the current network conditions. Based on their operational mechanism, these usually fall into three broader categories: loss-based, delay-based, and hybrid congestion control algorithms [5]. The categorization of these algorithms is presented in Table 2.1.

2.2.1 Loss Based Congestion Control

It is one of the traditional and most widely used TCP congestion control approaches, which considers packet loss as a signal of network congestion. When packet loss occurs, the sender immediately cut down the congestion window size to prevent further congestion. When loss-based flows compete with delay-sensitive flows, they can create problems at the bottleneck. In such situations, loss-based flows continue to fill the bottleneck buffer until an overflow occurs. This leads to an increased queuing delay and degrades the overall network performance. Some examples of congestion control algorithms that use packet loss to detect congestion are explained below.

Reno It is a traditional TCP congestion control approach that extends the original TCP Tahoe by introducing improvements in congestion recovery. During the slow start period, the congestion window increases exponentially. On the other hand, in the additive increase phase, the growth of the window is linear, as given by Eq. (2.1) [6].

$$w(t+1) = w(t) + 1 \tag{2.1}$$

where $w(t)$ presents the congestion window size at time t.If packet loss is detected, the size of the congestion window is halved, as depicted in Eq. (2.2).

$$w(t+1) = \frac{w(t)}{2} \tag{2.2}$$

TCP Reno incorporates fast retransmit and fast recovery mechanisms to improve overall network efficiency. When three duplicate ACKs are received, it interprets this as a sign of packet loss and initiates the fast retransmit phase, allowing the sender to resend the missing packet without waiting for a timeout [7]. During the Rapid Recov-

Table 2.1 Classification of TCP congestion control algorithms

Category	Algorithm	Description
Loss-based	Tahoe [19]	Traditional approach that uses slow start, congestion avoidance, and fast retransmit to adjust the window size
	Reno [6]	Enhancement over Tahoe by including fast recovery
	Newreno [8]	Improves the performance of Reno by handling multiple packet drops within a single congestion window
	TCP SACK [20]	Enhances TCP by allowing selective acknowledgment of received packets, enabling efficient recovery from packet loss
	HSTCP [21]	Suitable for TCP with a large congestion window and adjusts its growth rate to maintain efficiency over high-speed, long-delay networks
	Scalable TCP [22]	Modifies the additive increase and multiplicative decrease phase where cwnd grows independent of its current size
	BIC [9]	Uses a binary search approach to adjust the congestion window efficiently
	CUBIC [10]	Uses cubic window growth function for high-speed and long-distance networks
Delay-based	TCP DUAL [23]	Utilizes queuing delay to adjust the congestion window and avoid early congestion
	Vegas [12]	Detects congestion by monitoring RTT variations before packet loss occurs
	Vegas-A [24]	Refines TCP Vegas by using queuing delay to adjust the sending rate for better fairness and efficiency
	FAST [13]	Maintain stable throughput and low delay over high BDP networks
Hybrid-based	TCP Africa [25]	Combines delay and loss signals to enhance performance and scalability in high BDP networks
	Compound TCP [15]	Combines loss-based and delay-based components for balanced aggressiveness
	TCP Libra [26]	Ensure fair bandwidth sharing and stable performance in high-speed networks
	TCP Illinois [18]	Improves scalability and fairness by adjusting the congestion window dynamically
	BBR [27]	Estimates bottleneck bandwidth and RTT to maintain optimal sending rate
	Veno [4]	Uses Vegas delay sensitivity with Reno loss response for better efficiency

ery state of Reno's TCP protocol, the congestion window will repeatedly increase with every duplicate acknowledgement (ACK) received until it receives an acknowledgment telling it that a new packet has successfully been sent/received. When Reno experiences several packet losses during that same window of transmission, it has to wait to receive a separate ACK before it can attempt to recover any additional packets; thus, it may become less efficient when packets are being lost frequently.

Newreno: The primary function of TCP Newreno is to provide a means for properly dealing with more than one packet loss within a given congestion window during a single transmission. TCP Newreno incorporates the same slow-start and congestion avoidance algorithms as TCP Reno; however, it has been designed with enhanced fast recovery functionality which allows TCP Newreno to support the retransmission of packets upon receiving three duplicate acknowledgements from the destination and to minimize the drastic reduction of the congested window size after losing multiple packets [8]. In comparison to Reno, Newreno does not leave the fast recovery stage until all unacknowledged packets have been acknowledged (received). Therefore, this would allow Newreno to recover from multiple packet losses during the same time period (RTT). Because of this characteristic, it may achieve higher throughput when recovering from a larger number of packet losses or during periods of high rates of packet loss. Following this, the congestion window (window size) behaves in the additive increase policy that was established when all of the previously unacknowledged lost packets have been accounted for.

BIC: It stands for Binary Increase Congestion Control, developed to enhance TCP performance over high-bandwidth and long-delay networks. It uses the binary search-based approach to adjust the window size in order to improve scalability and fairness. The increase of window size depends on both the binary search increase and the additive increase phase [9]. When the current window size is far below the previous maximum, BIC uses additive increase, growing linearly to avoid stressing the network. As the window gets closer to the previous maximum, it switches to binary search increase, probing the midpoint between minimum and maximum to more precisely approach the available bandwidth. The algorithm enters the slow start mode when the previous maximum window is unknown. In such a situation, it increases the congestion window exponentially at first, but only up to a fixed limit. After reaching this limit, it switches to binary search increase to carefully probe the available bandwidth.

CUBIC: It is an enhancement of TCP BIC that helps to improve the performance over high-bandwidth, high-latency networks. Unlike traditional TCP algorithms that use linear or additive increase methods, TCP CUBIC utilizes a cubic function to regulate the growth of the congestion window, as illustrated in Eq. (2.3) [10].

$$w(t+1) = \alpha \times (\tau - \theta)^3 + w_m \tag{2.3}$$

where w_m is the window size just before the last packet loss, α is the scaling constant factor, and θ is the time required to increase w to w_m which can be calculated using

Eq. (2.4).

$$\theta = \sqrt[3]{\frac{w_m}{\alpha}} \tag{2.4}$$

TCP CUBIC is a highly popular protocol for congestion control protocol in modern networks, including the Linux operating system. This protocol works well for high bandwidth-delay product channels and also ensures fair resource allocation among competing flows. Its adaptability and improved throughput make it a preferred choice for high-speed internet applications.

2.2.2 Delay Based Congestion Control

Delay-based congestion control algorithms prevent congestion by observing changes in network delay. The increase in queuing delay indicates the increase in network congestion [11]. These algorithms take into account these delay variations and adjust the sending rate accordingly. Loss-based algorithms react only after congestion has occurred. However, the delay-based algorithm reacts proactively to manage network traffic, leading to lower latency. However, their effectiveness can be reduced in mixed environments where loss-based flows dominate, as those flows tend to be more aggressive in utilizing bandwidth. Some examples of delay-based congestion control algorithms are explained below.

Vegas It follows the identical slow start procedure used in TCP Reno. In the congestion avoidance phase, the algorithm determines the gap between the anticipated and actual throughput as illustrated in Eq. (2.5) [12].

$$d = \frac{w(t)}{RTT_b} - \frac{w(t)}{RTT}, \tag{2.5}$$

where RTT_b represents the base RTT, d represents the difference and $w(t)$ is the window size at time t. After the computation of the difference d, adjust the window size as mentioned in Eq. (2.6).

$$\mathrm{w}(t+1) = \begin{cases} w(t)+1, & \text{if } d < \alpha \\ w(t), & \text{if } \alpha < d < \beta \\ w(t)-1, & \text{if } d > \beta \end{cases} \tag{2.6}$$

where, α and β are two constants whose values are set as 1 and 3. When the variance between predicted and measured throughput is small, the intermediate buffer space is underutilized. Therefore, it is necessary to increase the window size efficiently. On the other hand, when the difference is large, there is a higher chance of congestion, so it is necessary to decrease the window size to avoid packet loss.

In the fast retransmit phase, it offers an improvement. Upon receiving a duplicate ACK, it verifies whether the RTT exceeds the timeout value. If it is, the algorithm enters the fast retransmit phase immediately, without waiting for three duplicate ACKs, and retransmits the missing packets.

FAST TCP It is a delay-based congestion control approach that ensures fairness and stability. It adjust the sending rate by observing the delay variations instead of packet loss. This allows to give better bandwidth utilization over high-latency channel. In addition, this approach helps prevent excessive queuing and minimizes retransmissions, resulting in a smoother data flow. In the congestion avoidance phase, it increases the window size with the help of the average RTT as illustrated in Eq. (2.7) [13].

$$w(t+1) = \min\left(2w(t),\ (1-\zeta)w(t) + \zeta\left(\frac{RTT_b}{RTT} \times w + \rho(w(t), d_q)\right)\right) \quad (2.7)$$

where $\zeta \in [0, 1]$ is a weight factor, d_q represents the queueing delay in terms of buffered packets at the router, and RTT_b is the base RTT when no queueing occurs.

2.2.3 Hybrid Congestion Control

This type of algorithms combines the loss-based and delay-based approaches to adapt various network conditions effectively. It considers both packet loss as well as delay to detect the network congestion. In the loss-based approach, it aggressively increases the sending rate to maximize throughput. However, this may lead to unfairness among flows and can suffer in high-latency channels. On the other hand, delay-based approaches maintain better fairness by adjusting rates, but this may result in degraded network performance. To overcome these issues, hybrid congestion control approaches are used to ensure better performance [14]. Some examples of such algorithms are Compound TCP (CTCP), Veno, and Illinois, which are discussed below.

Compound TCP It is a hybrid congestion control algorithm that uses both packet loss and delay as signal of network congestion. This algorithm is useful in case of high-speed and long-delay networks by maintaining fairness with other flows. It increase the utilization of bandwidth by adjusting the congestion window using network delay and packet loss signals. CTCP is particularly useful in environments where traditional TCP struggles to scale. This approach helps enhance stability and performance without causing excessive retransmissions. First, it computes the difference between the expected and actual throughput as shown in Eq. (2.5). After that, adjust the size of the congestion window as illustrated in Eq. (2.8) [15].

$$\mathrm{w}(t+1) = \begin{cases} w(t) + 1 + (\alpha(w(t)^k) - 1)^+, & \text{if } d < \beta \\ (w(t) - \gamma \times d)^+, & \text{if } d \geq \beta \\ w(t) - (1-\rho) - \frac{w(t)}{2}, & \text{if } loss \end{cases} \tag{2.8}$$

where α, k, and ρ are tunable parameters used to achieve better performance. The parameter values are set as $\alpha = 0.125$, $\rho = 0.5$, and $k = 0.75$. The threshold value β is set as 30 and γ must be > 0.

Veno TCP Veno is designed to distinguish between packet losses due to network congestion and those caused by random errors [16]. In its congestion avoidance phase, it calculates the difference between the estimated and actual throughput, as given by Eq. (2.5). The obtained value is compared with a threshold to decide if the available bandwidth is fully utilized. If the delay observed remains less than the threshold, then TCP Veno concludes that the network is not congested and increases the congestion window more aggressively. This behavior is similar to that of TCP Reno in such situations. However, if the delay is above the threshold value, it is considered a signal of congestion and the algorithm increases the congestion window after every two RTTs [17].

Illinois TCP Illinois is a congestion control algorithm developed to achieve efficiency in the context of high-speed and long-latency networks. It manages the congestion window based on network delay and buffer occupancy [18]. While there is little network delay, it increases the sending rate to utilize more bandwidth; on the other hand, it decreases the sending rate when it suffers from high delay in order to avoid congestion. This kind of adaptability helps it reduce packet loss and maintain a stable connection. It also ensures fairness among concurrent flows while ensuring better throughput, thus being suitable for dynamic modern network contexts.

2.3 Future Trends and Emerging Research in TCP Congestion Control

2.3.1 Multipath TCP

The growth of the Internet continues to grow every day. When created, TCP was designed as the primary method for sending information through a single interface throughout the 'Internet' or 'network'. Due to overwhelming increase in required throughput and desired reliability for services, new networks are now being built with many networks' interfaces [28]. The development of protocols allowing for the simultaneous transfer of data has resulted in the creation of more sophisticated means of transferring information. Among them is the creation of Multipath TCP, which builds upon standard TCP and provides the ability to transfer data concurrently using multiple routes for enhanced performance and better fault tolerance.

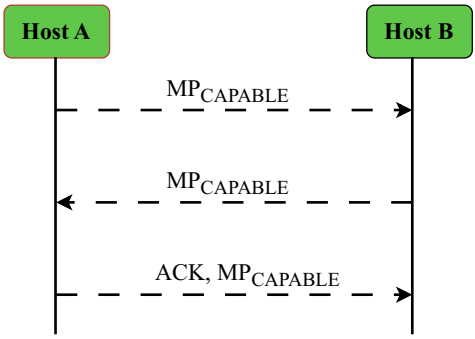


Fig. 2.1 Initiating for MPTCP connection

MPTCP enhances the way data is transmitted within a single connection by enabling the connection to send and receive data over all available network routes simultaneously. This means that MPTCP effectively shares out how much data each route needs to handle which minimises the likelihood of congestion. This type of functionality is ideal in mobile devices, large-scale data centres and cloud infrastructures where many network interfaces (Wi-Fi, cellular) are frequently used together. By allowing the network to server multiple connections simultaneously via multiple routes, MPTCP improves the use of available network resources which results in an improved performance, even in cases where one or more routes might experience latency or data loss. Additionally, MPTCP not only increases throughput, but also increases the ability of the network to provide reliable, stable communication over different and changing network conditions.

The establishment of an MPTCP connection begins with a standard TCP handshake that is extended to support multipath functionality. During the initial connection setup, the communicating devices exchange the $MP_{CAPABLE}$ option to indicate both source and destination side supports MPTCP as shown in Fig. 2.1. After the connection is established, the sender can add additional subflows by exchanging the MP_{JOIN} option. This mechanism enables the endpoints to utilize multiple network interfaces or routes, thereby enhancing throughput and improving fault tolerance. During data transmission, IP addresses may change over the MPTCP connection. Options such as ADD_{ADDR} are used to share additional addresses that can be used to initiate new subflows. These options work together to dynamically manage multiple paths, allowing MPTCP to adapt to changing network conditions and maintain efficient and reliable communication.

There are several congestion control algorithms that have been proposed for MPTCP to improve network performance. These algorithms are divided into two categories, such as coupled and uncoupled algorithms [29]. Coupled congestion control algorithms coordinate the behavior of multiple subflows to improve fairness and resource utilization. Some examples of such algorithms are LIA (Linked Increase Algorithm), OLIA (Opportunistic Linked Increase Algorithm), and BALIA (Balanced Linked Adaptation Algorithm) [30]. The uncoupled congestion control algorithms treat each subflow independently. They can adjust their window size inde-

Table 2.2 Multipath TCP congestion control algorithms

Type	Algorithm	Description
Coupled	LIA	Balances congestion across subflows using coupled increase/decrease rules
	OLIA	Enhances LIA for better fairness and resource pooling in heterogeneous paths
	BALIA	Improves fairness and stability between subflows
Uncoupled	Reno, Newreno, CUBIC, BIC, Vegas etc.	Manage each subflows independently and behave similar to standard TCP

pendently on the basis of the network condition. Some example of such Newreno, BIC, CUBIC etc. Table 2.2 shows different types of MPTCP algorithms.

These congestion control approaches help to manage the data transmission over multiple paths. LIA adjusts the sending rate by balancing performance. It ensures that it does not dominate the available bandwidth and degrade the performance of other flows [31]. It is the default congestion control algorithm for MPTCP. It adjust the sending rate by observing the RTT and packet loss event. When the available bandwidth is not fully utilized, the congestion window is gradually increased to make better use of the network capacity. In the congestion avoidance phase, it adjust the congestion window size as illustrated in Eq. (2.9).

$$W_r = \min\left(\frac{\alpha_r}{W_{total}}, \frac{1}{W_r}\right) \tag{2.9}$$

Here, W_{total} is the sum of the window size of all available subflow, W_r is the congestion window size of path r, and the parameter α is computed as illustrated in Eq. (2.10).

$$\alpha_r = W_{total} \times \frac{\max(\frac{W_r}{RTT_r^2})}{(\sum_{p \in S} \frac{W_p}{RTT_p})^2} \tag{2.10}$$

On the other hand, if packet loss or congestion is detected, the window size is reduced to prevent further overload and to maintain network stability. It follows the same multiplicative decrease phase of Reno. This method allows LIA to adapt to changing network conditions while balancing performance across multiple paths. Although LIA adapts to changing network conditions and improves performance across paths, its aggressive behavior toward single-path TCP flows may result in unfair bandwidth usage. To address these fairness and efficiency issues, other algorithms such as OLIA and BALIA have been proposed.

OLIA takes this a step further by coupling the congestion window size and acts as an alternate for LIA approach [32]. It increases the window size gradually without overwhelming the network.In slow start mode, the slow start threshold (ssthresh) is set to 1 MSS when multiple paths are established, whereas for a single flow it is set

to 2 MSS. It follows the same fast retransmit and fast recovery phases as regular TCP. In the congestion avoidance phase it increases the window size as illustrated in Eq. (2.11) [33].

$$W_r = \frac{\frac{W_r}{RTT_r^2}}{(\sum_{p \in S} \frac{W_p}{RTT_p})^2} + \frac{\alpha_r}{W_r} \tag{2.11}$$

where, W_r represents the congestion window size for path r, RTT_r is the round-trip time for path r,and S is the set of active paths. The factor α_r is computed as illustrated in Eq. (2.12).

$$\alpha_r = \begin{cases} \frac{1/number_of_paths}{|C|}, & \text{if } r \in C \\ -\frac{1/number_of_paths}{|M|}, & \text{if } r \in M \text{ and } |C| > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.12}$$

where M and C are defined as follows:

$$M = \{r \mid W_r = \max_{p \in S} W_p\}, \tag{2.13}$$

$$C = \{r \mid r \in best_paths \text{ and } r \notin M\}. \tag{2.14}$$

BALIA focuses on achieving a better balance between efficiency and fairness, allowing smoother adjustments in response to network changes. These algorithms help MPTCP make better use of available network resources while minimizing issues such as congestion and instability. In the additive increase phase, it increases the window size as illustrated in Eq. (2.15) [34].

$$W_r = \frac{\frac{W_r}{RTT_r}}{(\sum_{p \in S} \frac{W_p}{RTT_p})^2} + \frac{\alpha_r + 1}{2} + \frac{\alpha_r + 4}{5} \tag{2.15}$$

where, the value of α_r is computed as given in Eq. (2.16).

$$\alpha_r = \frac{max_S(\frac{W_p}{RTT_p})}{\frac{W_r}{RTT_r}} \tag{2.16}$$

In case of packet loss, it immediately decreases the window size as illustrated in Eq. (2.17) [35].

$$W_r = \frac{W_r}{2} \times min(\alpha_r, 1.5) \tag{2.17}$$

These congestion control algorithms helps to enhance the performance of MPTCP networks by improving the overall performance, responsiveness, and fairness.

2.3.2 5G and Beyond

The demand for high-speed data transmission is increasing day by day. This has led to the development of new-generation mobile networks that offer faster speeds, lower latency, and enhanced connectivity for various applications [36]. This may create challenges for the mobile network operators in terms of implementation and management. To address these requirement, the International Telecommunications UnionâŁ“Radio communications Sector(ITU-R) has identified three primary capabilities for 5G networks: ultra-reliable and low-latency communications (URLLC) for critical applications, enhanced mobile broadband (eMBB) for high-speed data access, and massive machine-type communications (mMTC) to support large-scale sensor and IoT networks [37]. In past, the frequency range of the mobile network lies with in 6 GHz. However, with 5G, it can goes beyond up to 100 GHz. Basically, the frequencies ranges are classified in to three categories based on its characteristics. The frequency band from 300 MHz to 3 GHz is called the radio frequency range. It is commonly used in wireless communication such as in mobile networks. The frequency band from 3 GHz to 30 GHz, is known as the microwave band and supports high-speed data transmission such as satellite links and radar systems. The range between 30 GHz and 300 GHz is referred to as millimeter wave (mmWave), which allows for extremely fast data rates but is more easily affected by obstacles and weather. These frequencies may get blocked by any solid materials such as buildings, walls, trees, or even human bodies, which significantly attenuate the signal strength and limit the effective coverage range.The performance characteristics and limitations of each frequency range determine how they perform differently as well as how they are utilized. For example, the mmWave frequency bands in 5G Networks have been developed for use with extremely high data rates and provide the ability to handle a higher density of users connecting through the same area. In addition, they enable users to experience a vastly higher level of data transmission capability than previously available technologies.

Providing reliable communication between endpoints over 5G networks that use the mmWave band is very challenging. Providing reliable communication between endpoints is a critical function of the Transmission Control Protocol (TCP). Thus, we must develop a solution that makes TCP functional over 5G networks. The challenges associated with improving TCP performance over 5G networks have attracted many researchers to investigate ways to improve the performance of TCP over 5G networks. The traditional approach to using TCP for controlling congestion is to treat all packet losses due to network congestion; however, this fails to distinguish the many causes of packet loss (due to network congestion), thus hindering the effectiveness of TCP. To effectively address these limitations and achieve these performance targets, researchers must resolve the limitations associated with traditional TCP conges-

tion control algorithms [38]. Therefore, it is essential to understand the interaction between the mmWave link and TCP. The highly variable nature of mmWave channels can significantly impact TCP performance, affecting throughput, latency, and connection stability.

To enhance the performance of 5G networks, it is essential to deal with the signal blockage problems [39]. These blockages can reduce the effectiveness of mmWave transmissions by interrupting data flow, which in turn affects TCP performance. As a result, packet loss and delay will increase. TCP also struggles to operate correctly in such conditions. It cannot differentiate between losses caused by network congestion and those resulting from specific issues of 5G, such as blockages or alignment problems [40]. There are different factors that can either reduce or amplify the impact of blockages on TCP's performance. For example, smaller obstacles may only cause brief interruptions, especially when devices are in motion and can quickly move past them. However, longer or longer obstructions can significantly disrupt communication by increasing delays and forcing TCP to increase the packet retransmission unnecessarily. These conditions lead to longer RTTs and a higher probability of packet loss, which ultimately reduces the efficiency of data transmission [41]. To ensure reliable and efficient communication in 5G networks, it is necessary to develop TCP enhancements that can adapt to these challenges by accurately interpreting network conditions and responding appropriately. This has become a key area of research for improving performance in next-generation mobile networks.

In 5G networks, user devices are expected to connect simultaneously to multiple Radio Access Technologies (RATs) or several mmWave base stations. Many ongoing studies focus on enhancing TCP through Multipath TCP (MPTCP) to analyze its performance across mmWave and LTE connections, as it offers an effective end-to-end approach for achieving multi-connectivity. The mmWave links are highly variable and diverse in nature. Therefore, it is essential to examine how fluctuations in path reliability and the behavior of different congestion control algorithms influence each other before implementing MPTCP in such networks.

In [42], the authors proposed a novel transport-layer approach aimed at addressing the challenges of high-bandwidth yet highly variable channels in urban 5G mmWave networks. The study highlights how conventional TCP congestion control mechanisms struggle to maintain performance under rapid capacity fluctuations and frequent link interruptions that are common in dense urban environments. To overcome these issues, the authors introduced Frame-Based TCP (FB-TCP), a modified TCP variant specifically designed for mmWave urban deployments. This approach employs fuzzy logic within the congestion avoidance phase to dynamically adjust the congestion window growth, thereby enabling the protocol to effectively exploit large instantaneous capacities while responding more smoothly to sudden throughput degradations. The simulation results demonstrate that FB-TCP achieves higher sustained throughput compared to conventional TCP variants and effectively mitigates latency issues by preventing excessive queue buildup during link degradation. The findings indicate that incorporating mmWave linkâŁ“aware adaptations into TCP behavior can significantly enhance transport-layer performance in urban 5G mmWave environments.

A detailed analysis of end-to-end communication in 5G networks is presented in [43], focusing on the implementation of 5G mmWave technology. The researchers describe a detailed framework for simulating mmWave communication, implemented within the ns-3 environment. This framework integrates the physical, MAC, and core network layers to provide a realistic platform for performance analysis. The framework incorporates essential 5G features such as beamforming, dual connectivity, mobility, and handover management. By integrating precise channel modeling with higher-layer protocol operations, the simulator enables researchers to examine the effects of rapid signal fluctuations and blockages–characteristic of mmWave links–on overall network behavior, especially in relation to transport protocols like TCP. The study highlights the value of this open-source framework for conducting cross-layer performance analysis and for developing and testing innovative algorithms and architectures for next-generation wireless systems. In [44], the authors examine the growing challenges associated with increasing mobile data traffic and discuss how mmWave technology provides a promising solution for both access and backhaul in future 5G networks. They highlight that mmWave frequencies offer vast unused spectrum resources, enabling extremely high data rates suitable for dense urban and hotspot environments. However, mmWave communication faces several challenges, such as high path loss, signal blockage, and limited coverage, which can be mitigated through beamforming and directional antennas. The study emphasizes the integration of mmWave links for small-cell backhauling to reduce dependence on wired infrastructure, supporting scalable and cost-efficient network expansion. With proper network design and advanced antenna technologies, mmWave can effectively meet the capacity demands of next-generation mobile networks.

2.3.3 Adaptive Congestion Control Using Machine Learning

As networks become more advanced and complex with technologies like 5G, IoT, and edge computing, traditional TCP congestion control techniques face difficulties in coping with rapidly changing network conditions. It is essential to design an adaptive TCP congestion control approach that can intelligently respond to dynamic network conditions. ML (Machine learning) offers various techniques to improve congestion control by allowing TCP to adapt dynamically based on observed patterns and behaviors. The use of ML techniques has attracted growing interest for improving TCP congestion control by supporting adaptive and data-driven decisions. Unlike traditional algorithms that rely on predefined rules or heuristics, ML-based approaches can learn network behaviors from real-time data. By considering the data related to packet delays, throughput fluctuations, and loss occurrences, machine learning models can identify early signs of congestion [45]. By the use of models like learning based approach such as reinforcement learning and neural networks, it can adaptively modify the congestion window size and data sending rate to optimize performance across varying network conditions. As a result, it can adjust the sending rate before performance is severely impacted. Such intelligent congestion control mech-

anisms improve responsiveness, fairness, and throughput while minimizing latency and packet loss. ML also provides the capability to distinguish among packet losses resulting from network congestion or caused by other issues, such as signal interference or mobility-related disruptions. This distinction is crucial in environments where losses are not solely the result of traffic overload but can occur because of temporary or physical impairments. By incorporating learning-based decision-making, TCP can respond more accurately, improving data flow efficiency and reducing unnecessary retransmissions.

Several ML approaches, including reinforcement learning, decision trees, and deep neural networks, are being investigated to create adaptive congestion control systems that continuously learn from network feedback [46]. These methods help TCP fine-tune its behavior, improving throughput and minimizing delays without requiring constant manual intervention. ML has shown strong performance in tasks like regression, classification, and decision-making, which are fundamental yet critical components of networking problems. Therefore, adopting ML techniques is essential for achieving significant breakthroughs in end-to-end congestion control [47]. ML methods are generally divided into three primary categories, such as supervised learning, unsupervised learning, and reinforcement learning [48]. Supervised learning relies on labeled datasets, where the model is trained to map inputs to correct outputs and make predictions. Unsupervised learning, on the other hand, works with data that has no labels, enabling the model to uncover patterns or group similar data points together. Reinforcement learning focuses on teaching an agent to make decisions by interacting with its environment and learning from the results of its actions through rewards or penalties [49].

Some ML-based TCP algorithms include TCP Remy, where it performs decision-making under uncertainty [50]. It adjusts sending behavior dynamically by learning from network feedback to improve throughput and reduce delays and packet loss. It adjusts sending behavior dynamically by learning from network feedback to improve throughput and reduce delays and packet loss. Similarly, Performance-Oriented Congestion Control (PCC) is a congestion control algorithm that adapts the sending rate by observing the actual network performance. It focuses on optimizing throughput and reducing latency by making decisions based on measured metrics rather than just packet loss [51]. Many challenges remain in deploying machine learning-based congestion control. These include ensuring real-time processing capabilities, managing computational load, and fairness and stability between network users. Despite these challenges, integrating machine learning into TCP's operation holds significant potential for making communication more resilient. As research progresses, these intelligent solutions are expected to play a central role in addressing the demands of future network environments.

In [52], the authors examine various machine learning algorithms for detecting anomalies in industrial control systems that communicate using the Modbus/TCP protocol. A realistic industrial network dataset is utilized, containing both normal and attack traffic to evaluate algorithm performance. Different ML techniques, such as Support Vector Machine (SVM), Random Forest, k-Nearest Neighbors (kNN), and k-Means clustering, are analyzed to assess their effectiveness in distinguishing

between legitimate and malicious activities. The results demonstrate that supervised learning approaches, particularly SVM and Random Forest, outperform unsupervised methods in identifying abnormal behaviors. The study highlights the importance of well-structured and labeled datasets for improving the reliability of ML-based intrusion detection systems in industrial environments. In [53], the author introduces a machine learning-based approach for identifying the internal behavior of TCP connections through passive network observations. Instead of relying on variant-specific analytical models, the proposed framework analyzes traffic traces to estimate the size of the sender's congestion window. It also identifies which congestion control algorithm is being used. By training regression and classification models on features derived from packet flows, the method can accurately track congestion dynamics across different TCP variants such as Reno, BIC, and CUBIC. The evaluation under both controlled and real-world conditions shows that the model generalizes effectively, offering reliable inference of TCP states. This approach enables network analysts to assess performance and congestion characteristics efficiently without requiring access to endpoint configurations.

In [54], author presents a machine learning-based model designed to predict TCP packet loss in real time. The approach extracts key traffic features from active network flows to forecast potential loss events before they occur, allowing for proactive adjustments to improve transmission efficiency. Through experiments conducted under varying network conditions, the model demonstrates strong accuracy and adaptability, effectively anticipating loss patterns and helping to enhance overall TCP performance by minimizing retransmissions and latency. A Reinforcement Learning based framework called Meta's Virtual Fast Stream Reinforcement Learning (MVFST-RL) is proposed in [55]. The approach aims to enhance congestion control in transport-layer protocols and is specifically designed for the Quick UDP Internet Connections (QUIC) protocol. It employs an asynchronous learning mechanism to address the challenges associated with delayed network feedback and control responses. Using an actor-critic model, the system dynamically adjusts the data transmission rate by analyzing key network parameters such as delay, throughput, and packet loss. Experimental evaluations demonstrate that MVFST-RL adapts efficiently to diverse network conditions, offering significant improvements in latency reduction and bandwidth utilization compared to conventional congestion control algorithms. A deep reinforcement learning (DRL) approach for managing Internet congestion is proposed in [56]. In this study, congestion control is modeled as a decision-making process in which a learning agent continuously adjusts its sending rate using network feedback, including parameters like delay, throughput, and packet loss. By employing a neural network to learn the most effective transmission strategies, the approach enables adaptive adjustments in response to varying network conditions. Experimental outcomes show that the DRL-based solution achieves better performance and fairness compared to conventional congestion control schemes like TCP CUBIC and BBR. The findings emphasize the potential of reinforcement learning to create more intelligent and adaptive congestion control mechanisms for future Internet systems.

In [57], author introduces Pantheon, a unified framework designed to evaluate and compare Internet congestion control algorithms under realistic network environments. The platform enables researchers to run experiments across various network paths and conditions, providing consistent and reproducible performance assessments. Pantheon includes support for multiple congestion control techniques, allowing systematic analysis of metrics such as throughput, latency, and fairness. The authors highlight that this framework simplifies experimentation and promotes transparency in congestion control research, making it a valuable resource for developing and benchmarking new algorithms. Pantheon serves as a practical and collaborative environment for advancing congestion control research by bridging the gap between simulation-based evaluation and real-world deployment. It allows researchers to observe how different algorithms behave under diverse network conditions, making it easier to identify their strengths and weaknesses. The framework's modular design also enables seamless integration of new congestion control schemes for comparative testing. By providing a shared platform with standardized tools and datasets, Pantheon encourages reproducibility and accelerates innovation in the design of adaptive and efficient transport protocols. In [51], author introduces an adaptive congestion control algorithm that applies online learning to regulate data flow efficiently. Instead of depending on fixed congestion indicators like packet loss or delay, PCC Vivace learns from continuous feedback on network performance, including metrics such as throughput and latency. By evaluating the outcome of each transmission decision, the system refines its sending rate in real time to achieve optimal performance. This learning-based approach enables the algorithm to respond effectively to changing network conditions, maintaining stable and high throughput. Experimental results show that PCC Vivace offers improved efficiency and fairness compared to traditional congestion control methods, making it well-suited for today's dynamic Internet environments.

In [58], the author explores how emerging ML methods are reshaping congestion control strategies in computer networks. It outlines how ML-driven techniques, including reinforcement and supervised learning, can help transport protocols automatically adjust transmission behavior based on real-time network feedback. These data-driven models are capable of predicting congestion patterns, optimizing throughput, and maintaining stability across diverse network scenarios. The authors also discuss the practical challenges of applying ML in congestion control, such as ensuring robustness, interoperability, and adaptability in real-world deployments. The paper concludes that machine learning introduces a new phase of innovation, enabling congestion control mechanisms to become more intelligent and adaptive than conventional rule-based approaches. By learning from historical and ongoing traffic data, ML-based algorithms can anticipate congestion events before they occur and proactively adjust transmission parameters to prevent performance degradation. The authors suggest that future research should focus on developing lightweight and interpretable models that can operate efficiently in real-time environments. They also highlight the importance of combining traditional control theory with learning-based methods to achieve a balance between stability, fairness, and adaptability in next-generation transport protocols.

2.4 Conclusion

The enhancement of TCP has progressed over the years to keep up with the increasing demand for high speed data transmission across various networks. The traditional approaches mainly focused on managing network congestion and packet retransmission. However, the modern network including 5G and IoT systems, face more complex challenges such as frequent signal interruptions, mobility, and high-speed data requirements. These challenges have highlighted the limitations of conventional TCP algorithms. Therefore, it is required to design smarter and more flexible solutions that can better interpret network behavior and adapt accordingly. Ongoing research in this area is vital to ensure that TCP continues to meet the demands of evolving communication infrastructures and remains a key protocol for efficient, reliable data exchange. The future enhancement on TCP are moving toward adaptive and intelligent control strategies that combine delay awareness, loss detection, and learning-based techniques. These approaches can predict congestion more effectively, balance throughput and fairness, and adjust to varying network conditions. As technologies such as 6G, edge computing, and autonomous communication systems expand, these smart mechanisms will play a vital role in achieving stable and efficient data transmission.

References

1. Al-Saadi, R., Armitage, G., But, J., Branch, P.: A survey of delay-based and hybrid TCP congestion control algorithms. IEEE Commun. Surveys Tutor. **21**(4), 3609–3638 (2019)
2. Biswal, S.P., Barik, N.R., Patel, S.: Comparative analysis of congestion window size and throughput for BBR versus elastic TCP. In: 2023 IEEE 15th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 14–18. IEEE (2023)
3. Lin, J., Cui, L., Zhang, Y., Tso, F.P., Guan, Q.: Extensive evaluation on the performance and behaviour of TCP congestion control protocols under varied network scenarios. Comput. Netw. **163**, 106872 (2019)
4. Biswal, S.P., Patel, S.: A bandwidth delay product based modified Veno for high-speed networks: BDP-Veno. J. Netw. Comput. Appl. **231**, 103983 (2024)
5. Zhang, J., Yao, Z., Tu, Y., Chen, Y. 'A survey of TCP congestion control algorithm. In: 2020 IEEE 5th International Conference on Signal and Image Processing (ICSIP), pp. 828–832, IEEE (2020)
6. Padhye, J., Firoiu, V., Towsley, D.F., Kurose, J.F.: Modeling TCP reno performance: a simple model and its empirical validation. IEEE/ACM Trans. Netw. **8**(2), 133–145 (2002)
7. Patel, S.: Comparative analysis of cumulative distribution function for TCP congestion window size and triple-duplicate period. Int. J. Commun Syst **32**(15), e4114 (2019)
8. Parvez, N., Mahanti, A., Williamson, C.: TCP NewReno: slow-but-steady or impatient? In: 2006 IEEE International Conference on Communications, vol. 2, pp. 716–722. IEEE (2006)
9. Xu, L., Harfoush, K., Rhee, I.: Binary increase congestion control (BIC) for fast long-distance networks. In: IEEE INFOCOM 2004, vol. 4, pp. 2514–2524. IEEE (2004)
10. Wang, J., Wen, J., Han, Y., Zhang, J., Li, C., Xiong, Z.: Cubic-fit: a high performance and TCP cubic friendly congestion control algorithm. IEEE Commun. Lett. **17**(8), 1664–1667 (2013)
11. Biswal, S.P., Gupta, S., Kumar, A., Patel, S.: Comparative analysis of compound TCP with various end-to-end congestion control mechanisms. In: 2022 4th International Conference on

Advances in Computing, Communication Control and Networking (ICAC3N), pp. 1762–1767. IEEE (2022)
12. Poorzare, R., Calveras, A., Abedidarabad, S.: An improvement over TCP Vegas to enhance its performance in optical burst switching networks. Opt. Rev. **28**(2), 215–226 (2021)
13. Wei, D.X., Jin, C., Low, S.H., Hegde, S.: Fast TCP: motivation, architecture, algorithms, performance. IEEE/ACM Trans. Netw. **14**(6), 1246–1259 (2006)
14. Tan, K., Song, J., Zhang, Q., Sridharan, M.: A compound TCP approach for high-speed and long distance networks. In: Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications, pp. 1–12. IEEE (2006)
15. Pokhrel, S.R., Williamson, C.: Modeling compound TCP over WiFi for IoT. IEEE/ACM Trans. Networking **26**(2), 864–878 (2018)
16. Biswal, S.P., Patel, S.: Optimization of Veno parameter based on stochastic approximation: OVeno. Simul. Model. Pract. Theory **142**, 103121 (2025)
17. Kanagarathinam, M.R., Singh, S., Sandeep, I., Kim, H., Maheshwari, M.K., Hwang, J., Roy, A., Saxena, N.: Nexgen D-TCP: Next generation dynamic TCP congestion control algorithm. IEEE Access **8**, 164482–164496 (2020)
18. Liu, S., Başar, T., Srikant, R.: TCP-illinois: a loss and delay-based congestion control algorithm for high-speed networks. In: Proceedings of the 1st International Conference on Performance Evaluation Methodologies and Tools, pp. 55–es (2006)
19. Hernawan, A.: Comparative performance testing of the impact of ACK loss in TCP Tahoe, TCP Reno, and TCP new Reno on the ns-2 simulator. J. Inf. Telecommun. Eng. **7**(1), 91–101 (2023)
20. Kim, B., Kim, D., Lee, J.: Lost retransmission detection for TCP sack. IEEE Commun. Lett. **8**(9), 600–602 (2004)
21. Floyd, S.: Highspeed TCP for large congestion windows. Technical Report (2003)
22. Kelly, T.: Scalable TCP: improving performance in highspeed wide area networks. ACM SIGCOMM computer communication Review **33**(2), 83–91 (2003)
23. Wang, Z., Crowcroft, J.: Eliminating periodic packet losses in the 4.3-Tahoe BSD TCP congestion control algorithm. ACM SIGCOMM Comput. Commun. Rev. **22**(2), 9–16 (1992)
24. Srijith, K., Jacob, L., Ananda, A.L.: TCP Vegas-a: improving the performance of TCP Vegas. Comput. Commun. **28**(4), 429–440 (2005)
25. King, R., Baraniuk, R., Riedi, R.: TCP-Africa: an adaptive and fair rapid increase rule for scalable TCP. In: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies., vol. 3, pp. 1838–1848. IEEE (2005)
26. Marfia, G., Palazzi, C., Pau, G., Gerla, M., Sanadidi, M., Roccetti, M.: TCP Libra: exploring RTT-fairness for TCP. In: International Conference on Research in Networking, pp. 1005–1013. Springer (2007)
27. Scholz, D., Jaeger, B., Schwaighofer, L., Raumer, D., Geyer, F., Carle, G.: Towards a deeper understanding of TCP BBR congestion control. In: 2018 IFIP Networking Conference (IFIP Networking) and Workshops, pp. 1–9. IEEE (2018)
28. Barré, S., Paasch, C., Bonaventure, O.: Multipath TCP: from theory to practice. In: International Conference on Research in Networking, pp. 444–457. Springer (2011)
29. Peng, Q., Walid, A., Hwang, J., Low, S.H.: Multipath TCP: analysis, design, and implementation. IEEE/ACM Trans. Netw. **24**(1), 596–609 (2014)
30. Li, W., Zhang, H., Gao, S., Xue, C., Wang, X., Lu, S.: SmartCC: a reinforcement learning approach for multipath TCP congestion control in heterogeneous networks. IEEE J. Sel. Areas Commun. **37**(11), 2621–2633 (2019)
31. Wischik, D., Raiciu, C., Greenhalgh, A., Handley, M.: Design, implementation and evaluation of congestion control for multipath {TCP}. In: 8th USENIX Symposium on Networked Systems Design and Implementation (NSDI 11) (2011)
32. Khalili, R., Gast, N., Popovic, M., Boudec, J.-Y.L.: Opportunistic linked-increases congestion control algorithm for MPTCP. In: Internet-Draft Draft-Khalili-MPTCP-Congestion-Control-02, Internet Engineering Task Force, Oct. 2013. Work in Progress

33. Cai, Y., Xiong, H., Yu, S., Chen, M., Zhou, X. : D-OLIA: the packet loss differentiation based opportunistic linked-increases algorithm for MPTCP in wireless heterogeneous network. In: 2021 31st International Telecommunication Networks and Applications Conference (ITNAC), pp. 78–85 (2021)
34. Walid, A., Peng, Q., Hwang, J., Low, S.H.: Balanced linked adaptation congestion control algorithm for MPTCP. In: Internet-Draft Draft-Walid-MPTCP-Congestion-Control-04, Internet Engineering Task Force, Jan. 2016. Work in Progress
35. Biswal, S.P., Chatterjee, A., Alanam, K., Patel, S.: Performance analysis of multipath TCP congestion control algorithms. In: 2025 IEEE 14th International Conference on Communication Systems and Network Technologies (CSNT) pp. 1205–1209 (2025)
36. Poorzare, R., Augé, A.C.: Challenges on the way of implementing TCP over 5g networks. IEEE Access **8**, 176393–176415 (2020)
37. Reddy, A.P.K., Kumari, M.S., Dhanwani, V., Bachkaniwala, A.K., Kumar, N., Vasudevan, K., Selvaganapathy, S., Devar, S.K., Rathod, P., James, V.B.: 5g new radio key performance indicators evaluation for IMT-2020 radio interface technology. IEEE Access **9**, 112290–112311 (2021)
38. Lorincz, J., Klarin, Z., Ožegović, J.: A comprehensive overview of TCP congestion control in 5g networks: research challenges and future perspectives. Sensors **21**(13), 4510 (2021)
39. Zhang, M., Polese, M., Mezzavilla, M., Zhu, J., Rangan, S., Panwar, S., Zorzi, M.: Will TCP work in mmWave 5g cellular networks? IEEE Commun. Mag. **57**(1), 65–71 (2019)
40. Mateo, P.J., Fiandrino, C., Widmer, J.: Analysis of TCP performance in 5g mm-wave mobile networks. In: ICC 2019-2019 IEEE International Conference on Communications (ICC), pp. 1–7. IEEE (2019)
41. Polese, M., Jana, R., Zorzi, M.: TCP and MP-TCP in 5g mmWave networks. IEEE Internet Comput. **21**(5), 12–19 (2017)
42. Poorzare, R., Augé, A.C.: FB-TCP: a 5g mmWave friendly TCP for urban deployments. IEEE Access **9**, 82812–82832 (2021)
43. Mezzavilla, M., Zhang, M., Polese, M., Ford, R., Dutta, S., Rangan, S., Zorzi, M.: End-to-end simulation of 5g mmWave networks. IEEE Commun. Surveys Tutor. **20**(3), 2237–2263 (2018)
44. Dehos, C., González, J.L., De Domenico, A., Ktenas, D., Dussopt, L.: Millimeter-wave access and backhauling: the solution to the exponential data traffic increase in 5g mobile communications systems? IEEE Commun. Mag. **52**(9), 88–95 (2014)
45. Mirza, M., Sommers, J., Barford, P., Zhu, X.: A machine learning approach to TCP throughput prediction. ACM SIGMETRICS Performance Evaluation Review **35**(1), 97–108 (2007)
46. Maliha, M., Habibi, G., Atiquzzaman, M.: A survey on congestion control and scheduling for multipath TCP: machine learning versus classical approaches. In: 2023 18th Conference on Computer Science and Intelligence Systems (FedCSIS), pp. 49–61. IEEE (2023)
47. Zhang, T., Mao, S.: Machine learning for end-to-end congestion control. IEEE Commun. Mag. **58**(6), 52–57 (2020)
48. Sarker, I.H.: Machine learning: algorithms, real-world applications and research directions. SN Comput. Sci. **2**(3), 160 (2021)
49. Shakya, A.K., Pillai, G., Chakrabarty, S.: Reinforcement learning algorithms: a brief survey. Expert Syst. Appl. **231**, 120495 (2023)
50. Winstein, K., Balakrishnan, H.: TCP ex machina: computer-generated congestion control. ACM SIGCOMM Comput. Commun. Rev. **43**(4), 123–134 (2013)
51. Dong, M., Li, Q., Zarchy, D., Godfrey, P.B., Schapira, M.: {PCC}: re-architecting congestion control for consistent high performance. In: 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15), pp. 395–408 (2015)
52. Anton, S.D., Kanoor, S., Fraunholz, D., Schotten, H.D.: Evaluation of machine learning-based anomaly detection algorithms on an industrial Modbus/TCP data set. In: Proceedings of the 13th International Conference on Availability, Reliability and Security, pp. 1–9 (2018)
53. Hagos, D.H., Engelstad, P.E., Yazidi, A., Kure, Ø.: General TCP state inference model from passive measurements using machine learning techniques. IEEE Access **6**, 28372–28387 (2018)

54. Welzl, M., Islam, S., Von Stephanides, M.: Real-time TCP packet loss prediction using machine learning. IEEE Access (2024)
55. Sivakumar, V., Delalleau, O., Rocktäschel, T., Miller, A.H., Küttler, H., Nardelli, N., Rabbat, M., Pineau, J., Riedel, S.: MVFST-RL: An asynchronous RL framework for congestion control with delayed actions (2019). arXiv:1910.04054
56. Jay, N., Rotman, N., Godfrey, B., Schapira, M., Tamar, A.: A deep reinforcement learning perspective on internet congestion control. In: International Conference on Machine Learning, pp. 3050–3059. PMLR (2019)
57. Yan, F.Y., Ma, J., Hill, G.D., Raghavan, D., Wahby, R.S., Levis, P., Winstein, K.: Pantheon: the training ground for internet congestion-control research. In: 2018 USENIX Annual Technical Conference (USENIX ATC 18), pp. 731–743 (2018)
58. Wei, W., Gu, H., Li, B.: Congestion control: a renaissance with machine learning. IEEE Netw. **35**(4), 262–269 (2021)

Chapter 3
Introduction to Multipath TCP Congestion Control Algorithms



Subhra Priyadarshini Biswal and Sanjeev Patel

Abstract The development of faster and more sophisticated communication networks has affected how TCP operates. Researchers have developed numerous new TCP congestion control mechanisms to improve performance and reliability in today's increasingly complex networking environments. The growth of high speed data rates and broadband technologies have introduced many new limitations to the traditional model of TCP due to: (1) dynamically changing traffic patterns; (2) large bandwidth-delay products and (3) extreme low-latency requirements. As a result, many TCP researchers have developed new methods to enhance TCP's capabilities and responsiveness, thus creating next generation transport layer protocols that will provide high speed and consistent operation in mixed environments. An example of a newly developed protocol is Multipath TCP, which expands the original TCP Protocol by allowing simultaneous transmission of data from a single source to a single destination over multiple routes or paths. Multipath TCP takes advantage of the ability to utilize multiple paths to achieve greater resource utilization, increased fault tolerance and improved overall throughput, without changing anything at the end-host applications. However, integrating multiple subflows introduces several new challenges related to congestion control, fairness, and coordination among the paths competing for shared bottlenecks. In order to handle these challenges, several MPTCP congestion control algorithms such as Linked Increase Algorithm (LIA), Opportunistic Linked Increase Algorithm (OLIA), and Balanced Linked Adaptation (BALIA) have been proposed with the aim of balancing efficiency, fairness, and stability across heterogeneous paths. This chapter provides a comprehensive overview of the evolution of TCP and the design principles of MPTCP congestion control mechanisms. The goal of this chapter is to present an insight into the optimization of MPTCP performance to support the design of next-generation communication systems that require efficient, reliable, and intelligent data transport.

S. P. Biswal (✉)
School of Computer Engineering, Kalinga Institute of Industrial Technology, Bhubaneswar, India
e-mail: subhrabiswal94@gmail.com

S. Patel
Department of Computer Science and Engineering, National Institute of Technology Rourkela, Rourkela, Odisha, India
e-mail: patels@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_3

Keywords TCP · Multipath TCP · Fairness · LIA · OLIA · BALIA

3.1 Introduction

Transmission Control Protocol (TCP) is a connection-oriented transport layer protocol that guarantees reliable data transmission. Most of the widely used applications, including Email, the World Wide Web (WWW), and file transfer services, are based on TCP for efficient bulk data transfer [1]. It is a connection-oriented protocol that guarantees reliability with in-order packet delivery [2]. Various analytical models have been proposed to characterize its performance based on round trip-time (RTT) and packet loss. The various congestion control, flow control, and retransmission behavior of TCP can be captured using different analytical models [3]. Researchers can also perform an analysis of performance across various network conditions (e.g., variations to the latency, bandwidth, and packet loss rate) and use that information to understand the potential impact of these dynamics on TCP performance in high-speed networks, in which the conventional TCP mechanisms may not make optimal use of the available capacity. This may lead to suboptimal throughput and unfair resource allocation between multiple flows.

Various issues can affect TCP's window-based congestion control on high-speed and long-haul networks, such as high bandwidth-delay product, packet loss due to transmission errors, out-of-order packet delivery, and a limitation in buffer capacity at network devices [4]. TCP can become under-utilized or cause increased latency from bufferbloat if it is not tuned/enhanced; therefore, modeling TCP in such a way can demonstrate performance bottlenecks and help to develop optimized congestion control algorithms [5]. Additionally, these models provide insights into the interaction of TCP with other kinds of traffic and network mechanisms, providing network engineers with the necessary information to enhance the efficiency, fairness, and stability of the network.

A number of congestion control mechanism variants have emerged to overcome the limitation of traditional TCP in heterogeneous and high-speed environments [6]. Approaches are further classified based on congestion levels, namely: TCP Reno, Newreno, CUBIC, and Bottleneck Bandwidth and Round-trip propagation time BBR amongst others [7]. These algorithms aim at enhancing throughput, fairness, and responsiveness across various network scenarios. Modern operating systems most commonly make use of TCP CUBIC and BBR. TCP CUBIC works using a cubic function that increases the congestion window size to efficiently probe the available bandwidth [8]. On the other hand, TCP BBR follows the model-based approach to keep an optimum trade-off between bandwidth estimation and delay control [9]. Even with such developments, the single-path TCP variants still cannot utilize multiple network paths effectively. They also fail to handle asymmetric link characteristics which are quite common in today's heterogeneous networks.

Addressing such challenges, Multipath TCP (MPTCP) was proposed as an extension to the traditional TCP approach [10]. It allows simultaneous transmission of

data across multiple network interfaces or paths [11]. MPTCP enhances network utilization by aggregating bandwidth from different links. It helps in improving overall network throughput, its robustness, and resilience against path failures. It supports multiple applications by managing multiple subflows within a single connection. Each sub-flow is an independent TCP connection, ensuring reliability and ordered delivery [10]. This also helps in load balancing between the paths dynamically, increasing efficiency, thereby improving tolerance against faults. This becomes highly effective in mobile and data center environments with multiple interfaces like Wi-Fi, LTE, or 5G.

Introducing more than one connection (via the introduction of subflows) creates new problems related to fairness and congestion management. As all subflows utilize the same connection, MPTCP cannot take up more bandwidth from the common connection (bottleneck) than will a single TCP connection over that common/transit facility. Therefore, several different types of congestion management algorithms can be developed for the purpose of enforcing fairness on Shared Bottlenecks between multiple MPTCP connections. The congestion management algorithms can be categorized into two groups based on their method of operation, i.e., Coupled Congestion Control Algorithms, and Uncoupled Congestion Control Algorithms. For example, the Coupled Congestion Control Algorithms include the Linked Increase Algorithm (LIA), the Opportunistic Linked Increase Algorithm (OLIA), and the Balanced Linked Adaptation (BALIA) Algorithm. The LIA algorithm always ensures that TCP remains Friendly towards other TCP connections. The OLIA algorithm attempts to balance both Throughput and Fairness between multiple Paths of transit. Finally, the BALIA algorithm improves Stability, Responsiveness, and Playing an Important Role in Both Increasing Throughput and Reducing Congestion to have a significant impact on MPTCP Connections Performance and Optimize the Overall Performance of All MPTCP Connections [12].

MPTCP efficiency has been enhanced in the last few years employing Hybrid and Learning-Based methods to control congestion, with an emphasis on developing Adaptive Decision-Making by assessing current (Real-Time) Network Parameters (Delay, Packet Loss, Available Bandwidth) using Hybrid and Learning-Based Algorithms. The use of Hybrid and Learning-Based Algorithms in Cross-Layer Optimization will further enhance the overall performance of MPTCP Networks with respect to Congestion, particularly with the introduction of Explicit Congestion Notification (ECN). With the increasing need for High-Speed, Low-Latency Communication in 5G and beyond, Intelligent, Hybrid and Learning Based Algorithms for Congestion Control are expected to be highly successful in improving Dynamic Adaptability and Stability in Complex Network Conditions.

3.2 Fundamentals of Multipath TCP

MPTCP works at transport layer of the network protocol stack as shown in Fig. 3.1. The connection establishment process in MPTCP is similar to that of traditional TCP,

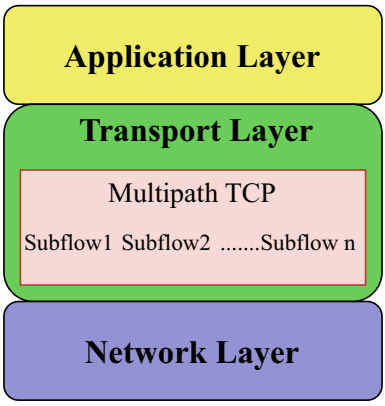


Fig. 3.1 MPTCP in the network architecture

as it also begins with the standard three-way handshake. Instead of using only SYN and ACK flag bits, MPTCP extends the traditional TCP handshake by embedding additional options within the TCP header through options such as $MP_{CAPABLE}$ [13]. This option confirms whether both communicating endpoints support MPTCP or not. This option enables both hosts to share unique identifiers or keys that are later used to verify and securely create extra subflows within the same MPTCP connection. The $MP_{CAPABLE}$ option is included in the SYN, SYN/ACK, and ACK packets during the first three-way handshake of an MPTCP connection. If the sender is uncertain whether the receiver has acknowledged the connection setup, it immediately retransmits the $ACK + MP_{CAPABLE}$ packet to ensure the successful establishment of the MPTCP connection.

During the three-way handshake, both endpoints exchange keys that serve as authentication material for establishing future subflows. In the case of creating additional subflows, both endpoints follow the same three-way handshake process used in a standard TCP connection. However, these packets also include the MP_{JOIN} option to associate the new subflow with the existing MPTCP connection [14]. When one endpoint intends to establish a new subflow with the other, it uses a token generated from the key exchanged during the initial handshake to identify the MPTCP connection that the new subflow will be associated with. To confirm authenticity, a Hash-based Message Authentication Code (HMAC) is generated using the keys from the $MP_{CAPABLE}$ handshake and the random values (nonces) exchanged in the MP_{JOIN} process. This mechanism securely associates the new subflow with the existing MPTCP session. In an MPTCP connection, the set of IP addresses associated with a host can change over time. MPTCP manages the addition and removal of these addresses through both explicit and implicit methods. The ADD_{ADDR} option is used for explicit address announcement, allowing a host to inform its peer about a new IP address that can be used to establish additional subflows.

To provide reliable and in ordered data delivery, MPTCP assigns a 64-bit Data Sequence Number (DSN) to all data exchanged within the connection. Each subflow maintains its own 32-bit TCP sequence number space, similar to standard TCP. A

special MPTCP mapping option links the subflow's sequence numbers to the overall DSN, allowing the receiver to correctly reconstruct the data in its original order. MPTCP also allows the priority of paths by the help of MP_{PRIO} signal. MPTCP also enables setting the priority of different paths using the MP_{PRIO} option. This signal allows a host to mark a particular subflow as either backup or regular. This helps the connection to decide which path to use for active data transmission and which to reserve for fail over or backup purposes. When a host wants to close only one subflow while keeping the MPTCP connection active, it follows the traditional TCP FIN/ACK flag bit procedure. On the other hand, when a host wants to indicate that it has completed sending all data for the entire MPTCP session, it sends a Data FIN within the DSS option to inform the other Host that no further data will be transmitted.

3.3 MPTCP Congestion Control Algorithms

MPTCP uses a congestion control approach to efficiently manage traffic from multiple concurrent paths while maintaining fairness with regular TCP flows. Unlike traditional TCP, which controls congestion over a single path, MPTCP coordinates several subflows that may have different capacities, delays, and levels of congestion. The main goal is to balance load across available paths to avoid the problems of overloading any single route, and ensure that the overall connection remains TCP-friendly.

MPCTP congestion control is categorized in to two types, such as coupled and uncoupled approaches. In the uncoupled approach, each subflow behaves like an independent TCP connection. These approaches manages the congestion based on the respective path conditions. As a result, it may give higher throughput by fully utilizing the corresponding path bandwidth. However, this may leads to the problem of unfairness. Some of the example of such approaches are Reno [15], Newreno [16], CUBIC [8], BIC [17], Vegas [18] etc.

The coupled approach, on the other hand, links the congestion control of all subflows within the same MPTCP session. By coordinating the adjustment of congestion windows, it promotes fairness, balances load across multiple paths, and prevents any single path from becoming congested. Examples of coupled algorithms include LIA, OLIA, and BALIA, which aim to optimize both performance and stability in multipath environments. These algorithms are explained below.

3.3.1 LIA

It is one of the traditional coupled congestion control methods in MPTCP. Its main goal is to ensure fairness with standard TCP connections while effectively utilizing multiple available paths. In LIA, all subflows within an MPTCP connection share

information about their congestion states with each other. The algorithm adjusts each subflow's congestion window in a coordinated manner. As a result, each and every MPTCP connection gets a fair bandwidth allocation. This helps maintain fairness when MPTCP and regular TCP flows coexist on shared links. The congestion window growth in LIA depends on both the individual path's performance and the collective behavior of all subflows. Each subflow increases its congestion window proportionally to its throughput and inversely to the level of congestion it experiences. It increase congestion window as shown in Eq. (3.1) [19].

$$\Delta w_r = \min\left(\frac{\alpha}{w_{\text{total}}}, \frac{1}{w_r}\right) \tag{3.1}$$

where, W_r is the congestion window size of subflow r, W_{total} is the total congestion window across all active subflows and α is the coupling constant that ensures TCP-friendliness and balances throughput across subflows. The computation of α is illustrated in Eq. (3.2).

$$\alpha = w_{\text{total}} \cdot \frac{\max\left(\frac{w_r}{RTT_r^2}\right)}{\left(\sum_k \frac{W_k}{RTT_k}\right)^2} \tag{3.2}$$

where RTT_r denotes the round-trip time of subflow r. The sum of total congestion window size and RTT is represented as w_k and RTT_k. When packet loss occurs on any subflow, the corresponding window size is reduced, ensuring that congestion on one path does not excessively impact others.

Overall, LIA provides a balance between throughput efficiency and TCP-friendliness. It allows MPTCP to take advantage of multiple network paths while avoiding aggressive behavior that could harm other TCP connections sharing the same network resources.

3.3.2 OLIA

The OLIA congestion control mechanism has been proposed to ensure efficient utilization of available multiple network paths by MPTCP and maintain fairness with the standard TCP flows [20]. In contrast to the single-path TCP, which sends data on only one route, MPTCP is able to perform concurrent transmissions of data on several paths. OLIA's basic objective is to balance these parallel transmissions by dynamically adjusting the congestion window in each path, depending on the performance characteristics such as delay, loss rate, and available bandwidth of that particular path. This ensures that all the network resources are utilized by the MPTCP sessions without overloading a particular link.

Traditional LIA strikes a balance between responsiveness and congestion fairness; this often requires compromising on either responsiveness or congestion fairness. In contrast, the design of OLIA does not require making such a trade-off: it allows responsiveness and optimal congestion balancing to be achieved simultaneously. One of the major contributions of OLIA is that it can effectively trade off throughput maximization against fairness [21]. The algorithm allocates different increase rates of the congestion window for each subflow depending on its congestion and efficiency. Higher priorities are given to paths that have lower congestion with shorter RTTs for transmitting data, while the growth of the congestion window will become slower on paths that are highly congested. This helps adaptively in dividing traffic among all paths more equitably, thus avoiding congestion collapse and enhancing overall stability [22].

OLIA also remedies some shortcomings of its predecessor, the LIA. LIA, while being TCP-friendly at all times, sometimes does not utilize the available capacity sufficiently because of its too-conservative window adjustments. This is improved in OLIA by the introduction of an opportunistic component that allows subflows with better network conditions to increase their sending rate more aggressively while still keeping the overall MPTCP connection fair against other competing flows. This leads to better link utilization and improved responsiveness in dynamic network conditions. The congestion window increase is fundamentally different between LIA and OLIA. Indeed, OLIA can be considered a modification of the TCP NewReno algorithm for efficient support of multiple concurrent paths within MPTCP. It extends the additive increase concept from TCP NewReno by making use of path-specific feedback, where each subflow is allowed to increase its congestion window based on its individual performance while preserving fairness across the set of all paths.

The additive increase of OLIA congestion control algorithm is illustrated in Eq. (3.3).

$$\Delta W_r = \frac{\frac{w_r}{RTT_r^2}}{\left(\sum_k \frac{w_k}{RTT_k}\right)^2} + \frac{\alpha_r}{w_r} \tag{3.3}$$

where, w_r is the congestion window size of subflow r, RTT_r denotes the round-trip time (RTT) of subflow r and α_r is the coupling factor for subflow r, which adjusts the increase rate based on the path's relative performance and congestion state. The coupling factor α_r is computed as illustrated in Eq. (3.4).

$$\alpha_r = \begin{cases} \frac{1/N_C}{|M|}, & \text{if } r \in \mathcal{M} \\ -\frac{1/N_C}{|N|}, & \text{if } r \in \mathcal{N} \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

where, N_C denotes number of available paths over MPTCP connections, M indicates that subflow r belongs to the set of well-utilized subflows and N indicates that subflow r belongs to the set of underutilized subflows.

In other words, OLIA enhances MPTCP performance by effectively balancing the data transmission across paths and adapting dynamically to fluctuating network conditions. Its design guarantees fairness toward traditional TCP flows, efficient use of available bandwidth, and robustness against fluctuating path delays and losses. Maintaining a balance between aggressiveness and fairness, OLIA has become one of the most effective and widely studied coupled congestion control algorithms for MPTCP environments.

3.3.3 BALIA

BALIA is the congestion control algorithm developed for MPTCP to better distribute and manage data over multiple network paths. This was designed to address some drawbacks of earlier schemes like LIA and OLIA in terms of efficiency, fairness, and responsiveness [23]. Balia adapts to fluctuations in network congestion so that the multiple flows connected to a MPTCP do not lose performance through lack of synchronisation with other connections.

Unlike LIA's much lower limit to the growth of the congestion window and OLIA, Balia has developed a balanced mathematical model that will react correctly to the congestion signal. The congestion window increases or decreases according to both the size of the congestion window and the round trip time for each flow. Faster connections will increase the overall throughput while slow or congested connections will decrease their sending rate on a temporary basis. It also treats TCP flows equally over multiple connections, meaning MPTCP will not take up all of the available bandwidth. The main intention of Balia is to create a structured approach to MPTCP that takes into account all of the important performance metrics of MPTCP, i.e. TCP responsiveness, TCP fairness, and stability in MPTCP. It introduces a new analytical design framework that enable researchers to understand how these properties interact among themselves. Within this framework, it becomes evident that a complete elimination of the trade-off between friendliness and responsiveness is impossible. Instead of neglecting this limitation, BALIA focuses on the maintenance of an optimal balance to ensure that the algorithm works efficiently while being as friendly as possible towards standard TCP flows. During the additive increase phase, the increase of the window size is performed as shown in Eq. (3.5).

$$\Delta w_r = \frac{\frac{w_r}{RTT_r}}{(\sum_k \frac{w_k}{RTT_k})^2} + \frac{\alpha_r + 1}{2} + \frac{\alpha_r + 4}{5} \tag{3.5}$$

where, α_r is the coupling factor which is illustrated in Eq. (3.6).

$$\alpha_r = \frac{max(\frac{w_k}{RTT_k})}{\frac{w_r}{RTT_r}} \tag{3.6}$$

BALIA strikes a balance between these two by carefully adjusting each subflow's congestion window based on network feedback, such as delay and packet loss. It makes sure that the subflows increase their transmission rates when bandwidth is available but decrease it upon detection of congestion. MPTCP connections operate with this mechanism to be fair in sharing network resources with traditional TCP flows but maintain a quick response to changes in network conditions. This method increases an organization's responsiveness to changes, while continuing to operate in a consistent manner across systems. The BALIA algorithm is also important for its ability to perform consistently, regardless of the type of network. The BALIA algorithm uses an algorithm to adjust to perform consistently on both wired and wireless networks and is able to adapt its performance to perform consistently across different types of networks, even when there is a discrepancy in the amount of time required for data to travel from one end of the network to the other (round trip time) and/or how much bandwidth is available on the given type of network (wired vs wireless). The BALIA algorithm decreases performance fluctuations due to changes in the amount of congestion that is present on the network by controlling how fast the size of the congestion window can be changed, and therefore does not lower performance abruptly because of large changes in the size of the congestion window. Therefore, due to these attributes, the BALIA algorithm works best in dynamic environments, such as heterogeneous networks and mobile networks.

The stability of the BALIA algorithm is derived from its design; it synchronises the control decisions that each active subflow makes in regards to congestion, while also maintaining a balance between all subflows. By carefully tuning the rate at which each subflow decreases and increases its congestion window, the BALIA algorithm can prevent excessive oscillation of congestion window sizes across all active subflows. In addition, the BALIA algorithm is able to provide an effective solution for congested-control in MPTCP in a wide range of network environments, including hybrid wired/wireless networks and heterogeneous mobile networks. Therefore, it will provide a good data transmission solution across all available paths without creating excess load on one of those paths. Its design enables the simultaneous achievement of fairness, stability, and responsiveness-qualities that earlier algorithms could not achieve together. By striking a fine balance between load distribution and congestion management, BALIA boosts the performance of multipath communication systems deployed in mobile, cloud, and data center networks.

3.4 Research Advancements in MPTCP

Several researchers have worked on the development of MPTCP congestion control mechanisms, which have been guided toward achieving performance, efficiency, and

fairness along multiple network paths. Most of the early works focused on developing mechanisms that could maintain fair coexistence between MPTCP and traditional single-path TCP connections in order to prevent resource dominance. Later works focused on better adaptability and responsiveness toward fluctuating network conditions. Various algorithms have been proposed for enhancing throughput, ensuring low latency, and maintaining stability during data transmission in heterogeneous environments. Some of the key and most recent works related to MPTCP are discussed in the subsequent sections.

Frommgen et al. [24] introduced a novel transport-layer scheme, which reduces latency in multipath scenarios using redundant packet transmissions across parallel sub-flows. The authors claim that for latency-sensitive applications, the traditional objective of maximizing throughput has to be complemented by strategies that minimize delay and packet re-transmission time. Their approach opportunistically sends duplicate packets across multiple paths, thereby reducing the time until a given packet is successfully received while also mitigating head-of-line blocking at the receiver side. They experimentally show that their redundancy-based approach may achieve much lower latency compared to standard Multipath TCP, especially when individual sub-flows in heterogeneous path environments are subject to variable delays or losses. This work emphasizes how multipath congestion control and scheduling mechanisms may be optimized not only for throughput but also for stringent real-time delay requirements.

Chao et al. [25] presented an overview of how MPTCP can be applied in vehicular networks, where the rapid mobility and frequent change of topology make it challenging to maintain a stable and efficient communication. The authors discuss that MPTCP is considered a strong candidate to enhance both reliability and throughput for V2X communications by virtue of its capability to transmit data simultaneously over multiple interfaces. The authors acknowledge that variations in dynamic routes, inconsistency in latency, and interruptions to calls created by handovers will prevent MPTCP from being used directly under these conditions. This review has therefore highlighted a growing need for the creation of an adaptive scheduling and path management strategy for effectively addressing all network fluctuations. This necessarily indicates future research efforts aimed at combining MPTCP with cross-layer optimization techniques and context-aware routing methods, with the goal of improving the performance and robustness of MPTCP over vehicular communication systems.

Aljubayri et al. [26] introduced possible solutions for minimizing transmission delays in MultiPath TCP (MPTCP) when utilized in an IoT scenario. The researchers state that in traditional MPTCP, latency occurs when the network load is not balanced on different links that are used simultaneously to send multiple packets. To solve these problems, they suggest an enhancement to the standard scheduling and congestion control methods that are used when sending multiple packets simultaneously.The authors propose a method by which the activity schedule of the existing congestion-control mechanisms on each link can be simultaneously adjusted to adapt to changes in the network environment, using real-time feedback. This new approach will utilize all available link resources more effectively, thereby reducing overall delay times and increasing responsiveness to user requests. Simulations demonstrate that their new

proposed methodology outperforms the current standard MultiPath Transmission Control Protocol (MPTCP) in terms of both overall throughput and latency performance. As such, their method is particularly well-suited for Internet of Things (IoT) applications, where delay times are crucial, including examples of smart healthcare, autonomous vehicles, and robotics/automated industrial control systems.

The research of Siddiqi et al. [27] consists of a collection of studies utilized to integrate artificial intelligence (AI) and multi-path transmission control protocol (MPTCP) in order to enhance the capacity of MPTCP for providing more intelligent and efficacious traffic control services. They demonstrated that traditional MPTCP algorithms could not compete effectively with the rapidly changing environments being created by today's modern networks due to the rapid changes created by changing paths, varying congestion levels, and changing available bandwidth. These challenges can be addressed through the application of AI techniques, for example, through the use of machine learning, undecided decision, and deep learning technologies for forecasting, optimal utilization of bandwidth, and making intelligence decisions in real time about congestion control. Authors classified the different approaches of AI-enabled MPTCP and ranked them in terms of how well they met the original goals established at the time of design and the effectiveness with which they achieved their outcomes. Furthermore, this work presents many future research opportunities that could potentially lead to the creation of self-optimizing or contextually aware intelligent MPTCP that has the capability of performing smart traffic control functions, reducing latency, and improving the overall effectiveness of communication in very complex heterogeneous network environments.

Silva et al. [28] followed a study on the implementation of MPTCP to optimize the usage of excessive amounts of data from IoT devices. They have been challenged to find ways to manage the large amount of data generated from IoT devices and therefore have found ways to use MPTCP to do this. They created a technique for distributing IoT traffic over the different networks connecting the device and used a dynamic method to allocate which network to use based on real-time conditions such as the quality of that network, how congested that network is, and how efficiently the device is operating that will offer an advantage in the use of multiple paths for data transmission. The authors also show how MPTCP will improve reliability and reduce latency while maintaining a level of balance between the multiple networks; as a result, it would improve the overall effectiveness of each network. The authors demonstrate that the implementation of MPTCP with IoT devices would provide an opportunity for better overall use of network resources and greater overall scalability and a greater degree of stability within a heterogeneous network scenario.

An intelligent real-time control mechanism to support MPTCP's ability to dynamically adapt to changes in network environments is proposed through the use of a learning-based architecture [29]. The authors studied how MPTCP could more efficiently adapt to changing network conditions than traditional methodology, which typically results in MPTCP being suboptimal. This methodology gives rise to the introduction of the use of reinforcement learning as the basis for MPTCP learning and modifying its behaviour in real-time. By continuously monitoring network feedback (e.g. delay, bandwidth, packet loss), MPTCP optimizes its path selection and con-

gestion management in a real-time manner. From experimental results, this adaptive method improves MPTCP's throughput and minimises its latency while maintaining fairness among all data paths. The authors' findings suggest that implementing an intelligent learning architecture within MPTCP could significantly enhance its ability to adapt and operate efficiently within a greater number of complex network scenarios.

Pokhrel et al. [30] presented the development of an advanced MPTCP mechanism to fulfill the severe demands of connected autonomous vehicles. Their methodology concentrates on attaining ultra-reliable and low-latency communication through effective handling of multiple network paths. With adaptive path selection and optimization techniques, the model ensures data delivery stability under fast-changing vehicular environments. The proposed method minimizes transmission delay and packet loss while maintaining a high level of reliability for real-time operations such as vehicular safety messaging and autonomous coordination.

Han et al. [31] developed an enhanced mechanism of MPTCP by integrating the BBR algorithm along with an adaptive packet scheduling strategy. The proposed approach focuses on improving throughput while maintaining the stability of a connection against dynamic and lossy network conditions. Through continuous analysis, the path characteristics of an adaptive scheduler intelligently distribute packets among available paths to prevent congestion buildup and minimize retransmissions. Such synergy between BBR's bandwidth estimation and real-time scheduling adjustments enables the efficient utilization of multiple paths for smooth data delivery and robust performance, even under fluctuating network scenarios.

Deguchi [32] studied the performance of various MPTCP scheduling algorithms over a wide range of network scenarios to explore each of their strengths and weaknesses. The work focused on the impact of network factors such as delay, bandwidth differences, and packet loss on scheduler efficiency. Based on this, strategies for optimizing path utilization while ensuring stability in data flows were suggested. These findings highlight that an appropriately designed scheduling mechanism plays a crucial role in load-balancing multiple paths for increasing throughput and ensuring reliable performance in heterogeneous networks.

Xing et al. [33] proposed an MPQUIC scheduling mechanism that is stream-aware for HTTP traffic in mobile networks. Its scheduler allocates different priorities according to the urgency and service requirements of various streams, ensuring that critical data is transmitted as soon as possible. By taking into account path characteristics, available bandwidth, and latency, delays can be minimized and the efficiency of an overall network improved. The results of this paper indicate that adopting stream-level awareness in multipath scheduling improves throughput and reliability in mobile network scenarios. Table 3.1 shows the summary of recent developements on MPTCP.

Table 3.1 Summary of recent developments in MPTCP research

Authors	Key contribution/focus
Frommgen et al. [24]	Proposed ReMP-TCP, a multipath TCP variant that minimizes end-to-end delay through improved packet scheduling and efficient path utilization for latency-sensitive flows
Chao et al. [25]	Conducted a comprehensive review of MPTCP applications in vehicular networks, addressing issues related to mobility management, path diversity, and reliability
Aljubayri et al. [26]	Developed a delay-optimized MPTCP scheme for IoT environments that enhances data transmission efficiency using adaptive scheduling mechanisms
Siddiqi et al. [27]	Explored AI-based approaches for traffic management in MPTCP, emphasizing adaptive congestion control and intelligent routing decisions
Silva et al. [28]	Proposed an MPTCP-based offloading framework for IoT systems that enhances throughput and reduces network congestion through efficient data distribution
He et al. [29]	Introduced a learning-driven, real-time transmission control method that dynamically tunes congestion parameters in MPTCP networks for better responsiveness
Pokhrel et al. [30]	Designed a highly reliable, low-latency MPTCP protocol suited for connected and autonomous vehicles, achieving robust communication performance
Maliha et al. [34]	Compared traditional and learning-based MPTCP congestion control mechanisms, identifying strengths and research gaps across both approaches
Deguchi [32]	Analyzed the fairness of MPTCP congestion control, focusing on how buffer size and propagation delay influence throughput balance among subflows
Kubde [35]	Evaluated multiple MPTCP schedulers under varying network conditions and suggested strategies to improve path utilization and overall flow stability
Han et al. [31]	Integrated BBR with adaptive packet scheduling to enhance throughput and stability of MPTCP in lossy or dynamic network environments

3.5 Conclusion

In this chapter, we examined the developments and recent advancements of MPTCP's congestion control mechanisms, scheduler designs, and the use of MPTCP in modern networks (e.g., IoT, vehicular communications, and mobile systems). The following congestion control methods were analysed: LIA, OLIA, BALIA and AI-assisted methods for balancing throughput, fairness and responsiveness of multiple network paths. Recent enhancements include adaptive packet scheduling, stream-aware techniques, and reinforcement-learning-based strategies that have shown great potential

for improving latency, path utilisation and overall network stability. All of these reflect the ability of intelligent scheduling and dynamic adaptation to effectively operate across heterogeneous and lossy networks. MPTCP remains a viable approach to efficiently utilise multiple network interfaces based on the studies reviewed in this chapter. Future work is expected to focus on integrating AI-based optimisation, cross-layer coordination, and ultra reliable low latency communication techniques into MPTCP for improved performance to support a variety of new and emerging network applications.

References

1. Widmer, J., Denda, R., Mauve, M.: A survey on TCP-friendly congestion control. IEEE Netw. **15**(3), 28–37 (2001)
2. Biswal, S.P., Patel, S.: Optimization of veno parameter based on stochastic approximation: Oveno. Simul. Model. Pract. Theory **142**, 103121 (2025)
3. Paxson, V.: Empirically derived analytic models of wide-area TCP connections. IEEE/ACM Trans. Netw. **2**(4), 316–336 (2002)
4. Biswal, S.P., Patel, S.: A bandwidth delay product based modified veno for high-speed networks: Bdp-veno. J. Netw. Comput. Appl. **231**, 103983 (2024)
5. Biswal, S.P., Gupta, S., Kumar, A., Patel, S.: Comparative analysis of compound TCP with various end-to-end congestion control mechanisms. In: Proceedings of the 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), pp. 1762–1767. IEEE (2022)
6. Prakash, M., Abdrabou, A., Zhuang, W.: An experimental study on multipath TCP congestion control with heterogeneous radio access technologies. IEEE Access **7**, 25563–25574 (2019)
7. Biswal, S.P., Barik, N.R., Patel, S.: Comparative analysis of congestion window size and throughput for BBR versus elastic TCP. In: Proceedings of the 2023 IEEE 15th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 14–18. IEEE (2023)
8. Ha, S., Rhee, I., Xu, L.: Cubic: a new TCP-friendly high-speed TCP variant. ACM SIGOPS Oper. Syst. Rev. **42**(5), 64–74 (2008)
9. Scholz, D., Jaeger, B., Schwaighofer, L., Raumer, D., Geyer, F., Carle, G.: Towards a deeper understanding of TCP BBR congestion control. In: Proceedings of the 2018 IFIP Networking Conference (IFIP Networking) and Workshops, pp. 1–9. IEEE (2018)
10. Barré, S., Paasch, C., Bonaventure, O.: Multipath TCP: from theory to practice. In: International Conference on Research in Networking, pp. 444–457. Springer (2011)
11. Biswal, S.P., Chatterjee, A., Alanam, K., Patel, S.: Performance analysis of multipath TCP congestion control algorithms. In: Proceedings of the 2025 IEEE 14th International Conference on Communication Systems and Network Technologies (CSNT), pp. 1205–1209. IEEE (2025)
12. Peng, Q., Walid, A., Hwang, J., Low, S.H.: Multipath TCP: analysis, design, and implementation. IEEE/ACM Trans. Netw. **24**(1), 596–609 (2014)
13. Ford, A., Raiciu, C., Handley, M., Bonaventure, O.: TCP extensions for multipath operation with multiple addresses. Technical Report (2013)
14. Wischik, D., Raiciu, C., Greenhalgh, A., Handley, M.: Design, implementation and evaluation of congestion control for multipath {TCP}. In: Proceedings of the 8th USENIX Symposium on Networked Systems Design and Implementation (NSDI 11) (2011)
15. Padhye, J., Firoiu, V., Towsley, D.F., Kurose, J.F.: Modeling TCP reno performance: a simple model and its empirical validation. IEEE/ACM Trans. Netw. **8**(2), 133–145 (2002)
16. Parvez, N., Mahanti, A., Williamson, C.: An analytic throughput model for TCP Newreno. IEEE/ACM Trans. Netw. **18**(2), 448–461 (2009)

17. Xu, L., Harfoush, K., Rhee, I.: Binary increase congestion control (BIC) for fast long-distance networks. In: IEEE INFOCOM 2004, vol. 4, pp. 2514–2524. IEEE (2004)
18. Low, S.H., Peterson, L.L., Wang, L.: Understanding TCP vegas: a duality model. J. ACM **49**(2), 207–235 (2002)
19. Raiciu, C., Handley, M.J., Wischik, D.: Coupled Congestion Control for Multipath Transport Protocols. RFC 6356 (2011)
20. Nguyen, K., Kibria, M.G., Ishizu, K., Kojima, F.: A study on performance evaluation of multipath TCP implementations. In: Proceedings of the 8th International Symposium on Information and Communication Technology, pp. 242–248 (2017)
21. Lubna, T., Mahmud, I., Kim, G.-H., Cho, Y.-Z.: D-olia: a hybrid mptcp congestion control algorithm with network delay estimation. Sensors **21**(17), 5764 (2021)
22. Khalili, R., Gast, N., Popovic, M., Boudec, J.Y.L.: Opportunistic Linked-Increases Congestion Control Algorithm for MPTCP. Internet-Draft draft-khalili-mptcp-congestion-control-05, Internet Engineering Task Force (2014)
23. Walid, A., Peng, Q., Hwang, J., Low, S.H.: Balanced Linked Adaptation Congestion Control Algorithm for MPTCP. Internet-Draft Draft-Walid-mptcp-Congestion-Control-04, Internet Engineering Task Force (2016)
24. Frommgen, A., Erbshäußer, T., Buchmann, A., Zimmermann, T., Wehrle, K.: Remp TCP: low latency multipath TCP. In: Proceedings of the 2016 IEEE International Conference on Communications (ICC), pp. 1–7 (2016)
25. Chao, L., Wu, C., Yoshinaga, T., Bao, W., Ji, Y.: A brief review of multipath TCP for vehicular networks. Sensors **21**(8), 2793 (2021)
26. Aljubayri, M., Peng, T., Shikh-Bahaei, M.: Reduce delay of multipath TCP in IoT networks. Wireless Netw. **27**(6), 4189–4198 (2021)
27. Siddiqi, S.J., Naeem, F., Khan, S., Khan, K.S., Tariq, M.: Towards AI-enabled traffic management in multipath TCP: a survey. Comput. Commun. **181**, 412–427 (2022)
28. Silva, C.F., Ferlin, S., Alay, O., Brunstrom, A., Kimura, B.Y.: Iot traffic offloading with multipath TCP. IEEE Commun. Mag. **59**(4), 51–57 (2021)
29. He, B., Wang, J., Qi, Q., Sun, H., Liao, J., Lu, L., Han, Z.: Learning-based real-time transmission control for multi-path TCP networks. IEEE Trans. Cognit. Commun. Netw. **9**(5), 1353–1369 (2023)
30. Pokhrel, S.R., Kumar, N., Walid, A.: Towards ultra reliable low latency multipath TCP for connected autonomous vehicles. IEEE Trans. Veh. Technol. **70**(8), 8175–8185 (2021)
31. Han, J., Xing, Y., Liu, J., Xue, K., Li, J., Wei, D.S., Sun, Q., Lu, J.: From an in-depth understanding of multipath TCP enhancement schemes to an adaptive control framework in wireless networks. IEEE Trans. Netw. Serv. Manag. **21**, 4563 (2025)
32. Deguchi, Y., Kobayashi, A., Tarutani, Y., Fukushima, Y., Yokohira, T.: Throughput fairness in congestion control of multipath TCP. In: Proceedings of the 2022 13th International Conference on Information and Communication Technology Convergence (ICTC), pp. 123–126. IEEE (2022)
33. Xing, Y., Xue, K., Zhang, Y., Han, J., Li, J., Wei, D.S., Li, R., Sun, Q., Lu, J.: A stream-aware mpquic scheduler for http traffic in mobile networks. IEEE Trans. Wireless Commun. **22**(4), 2775–2788 (2022)
34. Maliha, M., Habibi, G., Atiquzzaman, M.: A survey on congestion control and scheduling for multipath TCP: machine learning vs classical approaches. In: Proceedings of the 2023 18th Conference on Computer Science and Intelligence Systems (FedCSIS), pp. 49–61. IEEE (2023)
35. Kubde, V.S., Sawarkar, S.: Evaluation of mptcp schedulers in diverse scenarios. Eur. J. Electr. Eng. Comput. Sci. **5**(2), 50–54 (2021)

Chapter 4
Introduction to VANET: Architecture, Issues, and Challenges



Aditya Dev Mishra, Arshad Husain, and Hemant Kumar Saini

Abstract Establishing communication among vehicles and vehicles to infrastructure is one of the important aspects for providing services for humans, such as road safety and healthcare. Vehicular Ad Hoc Networks (VANETs) play a crucial role in achieving such services and establishing this infrastructure. VANETs are commonly used in Intelligent Transportation Systems (ITS), which incorporate the application and services related to traffic management, road safety, time optimization, commercial economics, and emergency services. Therefore, VANETs is one of the major research and trending areas among researchers. The major characteristics of vehicle communication and vehicle-to-infrastructure communication are mobility, dynamic topology, scalability, and varied communication environments, which lead to a major challenge and provide immense research opportunities. In this chapter, the authors have presented a comprehensive impression and outline of VANETs, which majorly include architecture and challenges. The chapter also focuses on critical issues, challenges related to performance measures, and concerns related to privacy and security. The authors have included an overview of the current state of the art and offer possible paths for establishing secure and efficient communications among vehicles. Finally, in this chapter, the authors have discussed the aspects of challenges for the implementation and give future recommendations related to the integration of VANETs with existing popular advanced technologies such as artificial Intelligence, edge computing, communication generation (5G/6G).

Keywords Vehicular ad hoc networks · Challenges · Intelligent transportation systems

A. D. Mishra (✉) · A. Husain · H. K. Saini
School of Computer Science Engineering and Technology, Bennett University, Greater Noida, India
e-mail: aditya2aug@gmail.com

A. Husain
e-mail: arshad.husain18@gmail.com

H. K. Saini
e-mail: drhksainimalaysia@ieee.org

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_4

4.1 Introduction

These days. Researchers need to pay more attention to Vehicular Ad Hoc Networks (VANETs) and how they might be improved as the number of vehicles and their demand grow quickly. VANETs are a type of Mobile Ad Hoc Network (MANET) that lets vehicles talk to each other (V2V: Vehicle-to-Vehicle) and to infrastructure (V2I: Vehicle-to-Infrastructure). This communication is vital for making roads safer, cutting down on traffic jams, and letting consumers use infotainment services. In the future, VANETs will be the most important aspect of smart city infrastructure and intelligent transportation systems. The problems include nodes moving around all the time, topologies changing quickly, and the need to focus on multiple communication conditions to make VANE Ts powerful, scalable, and safe. Researchers are working hard on VANET architecture [1], protocols, and security frameworks since there arae still a lot of difficulties that need to be fixed before they can be used in real life. These challenges include latency in communication, privacy protection, and security concerns in automotive situations. Mchergui et al. [2] investigate the function of AI in VANETs and delineate the advantages and disadvantages of AI-driven methodologies, such as machine learning and deep learning, as prospective research avenues within the vehicular context. Saad et al. [3] looked examined 68 studies about blockchain-based vehicular networks for the Internet of Vehicles (IoV) to find a vacuum in the research that could be filled in the future. Sohail et al. [4] examined the routing protocols employed in VANETs, considering several criteria including principles, architecture, and application domains, while proposing future trends in the design of these protocols.

The motivation for this study arises from the necessity to furnish a thorough comprehension of VANET architecture and to underscore the principal issues and challenges that impede its extensive implementation. This work seeks to guide researchers and practitioners in tackling unresolved issues and enhancing the design of dependable, efficient, and secure vehicular communication systems by synthesizing insights from recent advancements.

The main contribution of this chapter is as follows:

- To examine the overview of VANETs architecture, key issues, and applications in recent years.
- To examine the key challenges, such as security and privacy issues, performance challenges, and open research problems, in the presented study.

The structure of the presented chapter is as follows. Section 4.1 of the presented chapter included the introduction part, advancement in vehicular communication, and the role of VANETs in intelligent transportation systems. Section 4.2 described the components and applications of VANETs. Section 4.3 of the presented chapter discusses the architecture of VANETs, while Sect. 4.4 explains the key issues in the VANET system. Sections 4.5 and 4.6 discuss the challenges associated with security, privacy, and performance, respectively, in vehicular environments. Some of the prime

open research challenges are discussed in Sect. 4.7. At the end, Sect. 4.8 presented the conclusion and future extension of the presented study.

4.1.1 Advancement in Vehicular Communication Systems

The evolution of vehicular communication systems has grown rapidly over the last few decades as the technology has moved from wireless networks and GPS to edge computing and artificial intelligence. Initially, sharing of the real-time data within the vehicular environment is a tough task as it depends on sensors, radars, and GPS systems. It also affects the roadside safety, traffic management, and roadside infrastructure in intelligent transportation systems. Now, with the integration of federated learning, edge computing, and AI into the vehicular communication system, the security, privacy, and trust. The three phases of VANETs are depicted in Fig. 4.1. The first phase examined the challenges of collision avoidance, emergency services, and lane change assistance and overcame them. In the second phase, challenges such as large-scale traffic management, reliability, and cooperative driving have been addressed. The current phase addresses the challenges associated with security and privacy by incorporating federated learning and blockchain into vehicular environments.

The evolution of vehicular communication systems progresses from an isolated onboard system to a fully connected, intelligent, and autonomous system. This growth shows the critical role of VANETs in shaping the future of intelligent transportation systems and enhancing convenience for everyday users. Dwivedi et al. [5] explore the advantages of blockchain-based systems in VANETs, including opportunities and challenges, with the integration in different domains such as healthcare, supply chain, etc. [6] explore the integration of blockchain techniques and federated learning in an IoT-based vehicular network. De Souza et al. [7] explore the lack of standardization

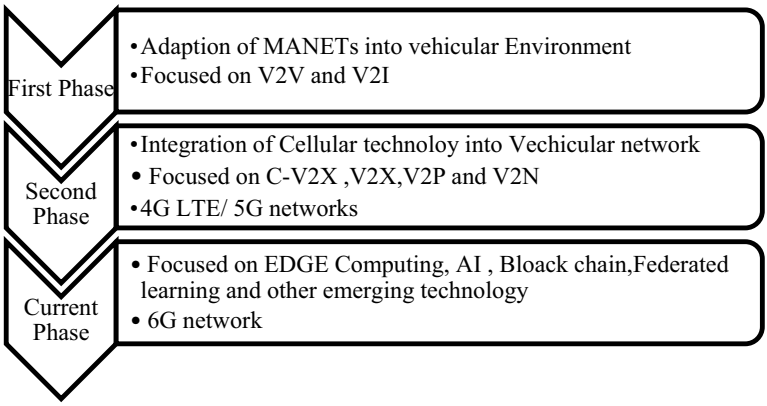


Fig. 4.1 Phases of VANETs

in simulation technology and the integration of emerging technology, such as cloud computing, in urban areas for the advancement of VANETs.

4.1.2 Significance of VANET in Intelligent Transportation Systems

VANETs are a big part of Intelligent Transportation Systems (ITS) because they let vehicles talk to each other (V2V), to infrastructure (V2I), and to everything (V2X). Vehicular networks transfer data fast and with little delay, which is important for avoiding accidents, sending emergency alerts, and driving together. VANETs also make scalability better by being able to handle dynamic topologies in places with a lot of movement and making sure that different types of communication technologies can join without any problems. When these technologies are added to ITS, it becomes more reliable, safer for drivers, and more efficient on the road.

4.2 Conceptual Overview of VANETs

This part gives an overview of VANETs, covering their features, parts, ways of communicating, and uses.

4.2.1 Definition and Characteristics

Vehicular Ad Hoc Networks are a type of Mobile Ad Hoc Network that is made to help smart transportation systems by letting cars talk to each other (V2V), to infrastructure (V2I), and to everything else (V2X). It makes life better for people by making traffic management more efficient, making roads safer, and offering infotainment services. Figure 4.2 shows the main features of VANETs. These important differences between VANETs and regular wireless networks create both opportunities and problems for research and development.

4.2.2 Components of VANET

Figure 4.3 shows the five main parts of VANETs: the Roadside Unit, the On-Board Unit, the Application Unit, the Trusted Authority, and the Communication channels. The On-Board Unit (vehicles), the Roadside Unit (infrastructure), and the backend system, which includes cloud computing and communication management, make up

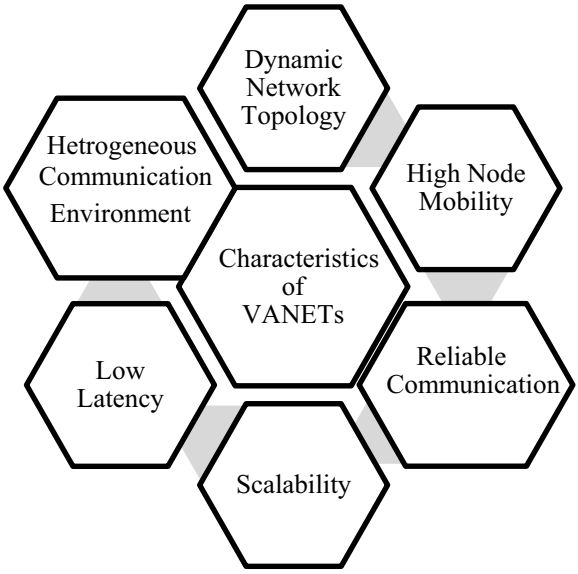


Fig. 4.2 Key characteristics of VANETs

the key parts of VANET. The On-board unit (OBU) is put in cars so they may talk to each other wirelessly (V2V) and to roadside infrastructure (V2I). This part also lets you run safety apps like warnings for changing lanes and avoiding collisions. The roadside unit (RSU) oversees physical infrastructure and also links cars to the backend systems. This part also included services like GPS updates, roadside hazard alerts, and entertainment services.

The backend infrastructure has a traffic management system that works with cloud servers and can also work with smart transportation systems. In this way, all parts are important for making communication quick and easy, keeping people safe, and controlling traffic in smart transportation systems.

4.2.3 Communication Paradigms (V2V, V2I, V2X)

VANETs typically offer three types of communication models, i.e. Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Everything (V2X). In the V2V model, as the name suggests, it has been used to establish the connection among vehicles; it does not include the infrastructure aspect [8]. The primary focus of this model is road safety by allowing a supportive consciousness of the communicating vehicle's drivers. The second model of communication is V2I, where communication is established between the vehicle and the infrastructure, such as base stations and traffic signals. The objective of this communication is traffic management, parking

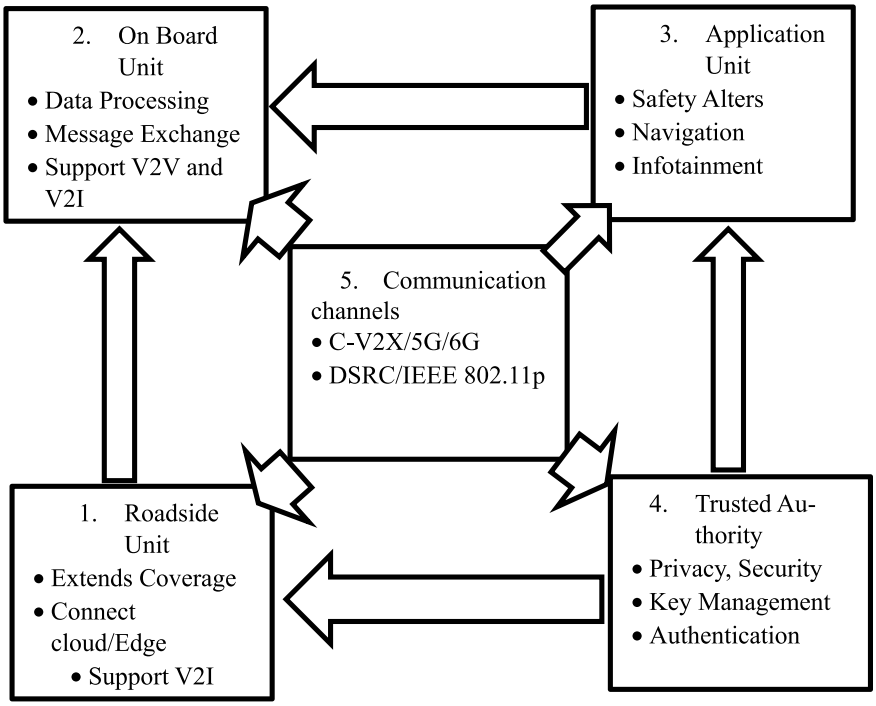


Fig. 4.3 Components of vehicular ad hoc networks

assistance, and other concerns related to infrastructure facilities. The final communication model is Vehicle-to-Everything (V2X), which consists of all types of communications such as V2V, V2I, Vehicle-to-Pedestrian (V2P), and Vehicle-to-Network (V2N). The objective of V2P is to provide pedestrian safety-related services, whereas cloud-based navigation and concerned information is provided by V2N. Ficzere et al. [9] present the insights of cellular vehicle-to-everything (C-V2X) and highlight the influence of 5G/6D in VANETs. Yusuf et al. [10] examine the utilization of V2X and discuss the recent development in advanced driver assistance systems for autonomous vehicles.

4.2.4 Applications of VANET

As shown in Fig. 4.4, the main uses of VANET can be grouped into traffic management, safety applications, and infotainment services. Autonomous vehicles are the most important use of VANETs today. They make it possible for reliable communication and coordination to make self-driving operations safer and more efficient.

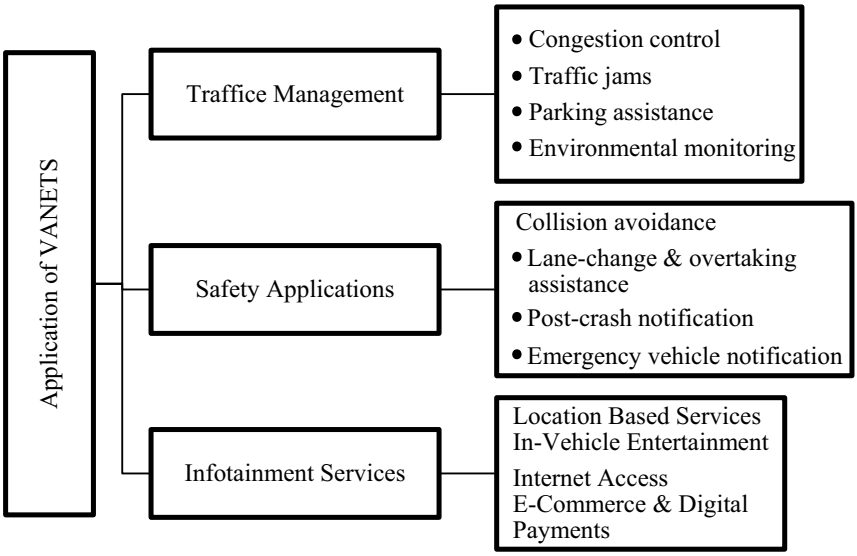


Fig. 4.4 Application of VANETs

4.3 VANET Architecture

Vehicular Ad Hoc Networks (VANETs) are the main part of current Intelligent Transportation Systems (ITS). The architecture of VANET outlines how vehicles, infrastructure, and other things work together to make sure that communication is reliable for safety, traffic control, and entertainment services. The design connects mobile nodes (vehicles with On-Board Units, OBUs), stationary infrastructure (Roadside Units, RSUs), and back-end servers (cloud/edge systems) in a general way, as depicted in Fig. 4.5. A well-structured architecture is important for vehicle scenarios that need high mobility, changing topologies, and different types of communication [8].

4.3.1 Network Topology

VANET network topology is inherently dynamic because of rapid vehicle mobility. Its structure can be categorized into three primary forms:

- **Pure Vehicular Ad Hoc Topology (V2V)**: Vehicles form a self-organizing network without relying on infrastructure. Communication is direct and decentralized, which is efficient for short-range, safety–critical applications like collision avoidance or cooperative driving.

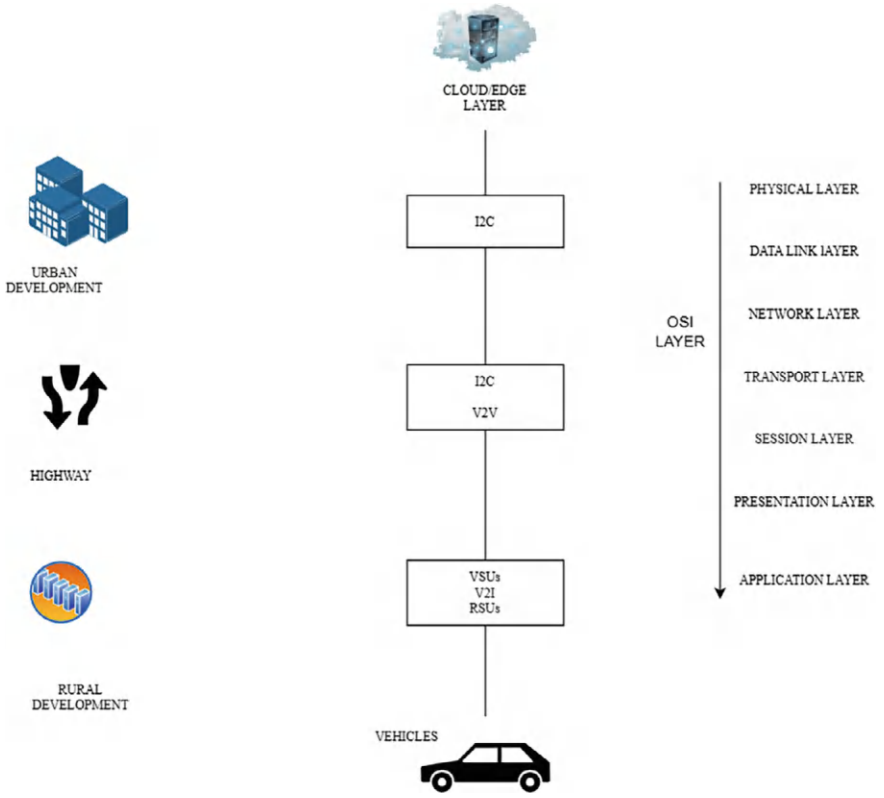


Fig. 4.5 VANET architecture

- **Infrastructure-Based Topology (V2I)**: Vehicles connect to RSUs deployed along the roads, highways, or intersections. RSUs act as gateways to the internet or cloud servers, providing traffic updates, infotainment services, and long-range communication support.
- **Hybrid Topology (V2X)**: A combination of V2V and V2I models, allowing vehicles to switch dynamically between ad hoc and infrastructure modes. This topology is more resilient to connectivity loss and supports advanced ITS applications, including navigation assistance and large-scale traffic management.

The network topology is often sparse in rural areas and dense in urban regions, making adaptive and scalable routing a necessity.

4.3.2 Communication Models

VANET communication models define entities that interact with each other as follows:

- **Vehicle-to-Vehicle (V2V)**: Direct exchange of information between vehicles using wireless technologies. It is crucial for safety applications such as lane change warnings, forward collision alerts, and cooperative adaptive cruise control [8].
- **Vehicle-to-Infrastructure (V2I)**: Communication between vehicles and RSUs. This model supports real-time traffic information, road hazard notifications, and internet access.
- **Vehicle-to-Pedestrian (V2P)**: Extends communication to vulnerable road users like pedestrians and cyclists using smartphones or wearable devices, enhancing road safety.
- **Vehicle-to-Network (V2N)**: Connects vehicles to broader communication networks through cellular infrastructure, enabling cloud services, remote diagnostics, and entertainment applications.
- **Vehicle-to-Everything (V2X)**: An umbrella model combining V2V, V2I, V2P, and V2N. It ensures holistic integration of all entities in the vehicular ecosystem to create a safer and smarter transportation environment.

4.3.3 Protocol Stack and Standards (IEEE 802.11p, DSRC, C-V2X)

The VANET protocol stack uses ideas from the OSI model, but also includes requirements that are specific to vehicles:

1. Physical and Data Link Layer:
   - **IEEE 802.11p (WAVE)**: A modified Wi-Fi standard that works well for vehicle communication and has low latency and short-range connectivity.
   - **Dedicated Short Range Communication (DSRC)**: A U.S. standard that works in the 5.9 GHz band and is made just for sending safety alerts to cars.
   - **Cellular-V2X (C-V2X)**: Built on LTE and 5G technologies it offers better coverage, scalability, and support for ultra-reliable low-latency communication (URLLC).
2. **Network Layer:** Uses position-based or topology-based techniques to handle routing [11] and addressing. GPSR (Greedy Perimeter Stateless Routing) and AODV variations that are best for vehicles are two of the most common protocols.
3. **Transport Layer:** Ensures that data is sent reliably or in real time, depending on the type of application (for example, TCP for infotainment and UDP for safety warnings).
4. **Application Layer:** Supports both safety–critical apps (like collision warnings and traffic alerts) and non-safety services (like streaming multimedia and ads).

5. **Security Layer (Cross-Cutting):** Uses PKI, pseudonym schemes, and blockchain-enabled solutions to make sure that authentication, privacy, and protection against bad attacks are all in place.

4.3.4 Infrastructure and Deployment Scenarios

The way VANETs are used depends on the type of route and the needs of the application:

- **Urban Deployment**: Putting a lot of RSUs near crossroads, traffic lights, and public locations makes sure that coverage is strong. These deployments help in managing traffic jams, smart parking, and coordinating public transportation.
- **Highway Deployment**: RSUs are put up at regular intervals along roadways. Lane departure alerts, accident notifications, and cooperative driving at high speeds are some of the safety features that are most important.
- **Rural Deployment**: Deployment in rural areas is limited because there aren't many vehicles on the road. V2V relays are more important for communication, while RSUs are only put at certain places, like curves, crossroads, or places where accidents are likely to happen.
- **Edge/Cloud-Integrated Deployment**: RSUs are connected to edge servers or cloud infrastructure [12] to give users real-time analytics, AI-based decision-making, and traffic optimization on a wide scale.
- **5G/6G Enabled Deployment**: Future deployments will use cellular networks for ultra-low latency and huge connectivity. This will make it possible for self-driving cars to work together and interact with smart city ecosystems.

4.4 Key Issues in VANET

Intelligent Transportation Systems (ITS) are built on Vehicular Ad Hoc Networks (VANETs), which let vehicles and infrastructure talk to each other in real time. But putting them into use in real life is hard because of several technological and operational problems that affect stability, dependability, and efficiency. The following are the biggest problems.

4.4.1 High Mobility and Dynamic Topology

Vehicles in VANETs move quickly, and they often go in opposite directions, which modifies the network's topology all the time. Connections between nodes don't last long, which causes frequent disconnections and unreliable communication pathways. This quick movement makes it harder to keep dependable routes, causes packet

loss, and slows down data delivery. Also, frequent changes in topology make typical MANET routing algorithms less successful, so adaptive solutions that are specifically made for automotive contexts are needed.

4.4.2 Scalability and Network Congestion

VANETs need to work in both sparse rural networks and busy urban traffic situations. In places with a lot of people, sending messages at the same time can cause channel congestion, collisions, and broadcast storms, which slow down the network. On the other hand, it is hard to keep connections in low-density areas because there aren't many nodes to relay data. The integration of different communication technologies (such as DSRC, LTE, and 5G/6G) makes scalability problems even worse. To avoid performance problems, these technologies need good bandwidth allocation and congestion control algorithms.

4.4.3 Routing Challenges

One of the main challenges with VANETs is that they have a very dynamictopology and a wide range of deployment circumstances, which makes routing difficult. Routing protocols that are based on position, topology, or a combination of the two sometimes have problems when there is a lot of traffic (which causes interference and overhead) or when there isn't much traffic (which causes significant delays and gaps in connectivity). VANETs need context-aware routing to serve safety–critical apps where timely delivery of packets is important, unlike MANETs. So, it is still hard to come up with routing strategies that strike a good balance between scalability, adaptability, and low latency.

4.4.4 Quality of Service (QoS) Issues

Managing QoS in VANETs is hard since different applications need different things. Safety applications, such as collision alerts, need very low latency and great reliability. Infotainment services, on the other hand, need larger throughput but can handle delays. When these kinds of apps are used together in the same space, it makes it harder to fairly divide up resources and set priorities. Also, things outside of the system, like ambient interference, channel fading, and the hardware limits of On-Board Units (OBUs) and Roadside Units (RSUs), make it even harder to guarantee QoS. One of the biggest problems in VANETs is making sure that service quality stays the same in changing and diverse environments.

4.5 Security and Privacy Challenges in VANETs

The security and privacy challenges in VANETs are critical as they affect the entire Intelligent Transportation Systems. The VANET's performance can be degraded due to a malicious attack on the network, such as denial-of-service, phishing, jamming, spoofing, and disrupting both computation and communication. Table 4.1 depicts the security and privacy challenges and their description in vehicular Ad Hoc network systems.

Mundhe et al. [13] suggested that many existing authentication schemes require a trusted authority with opaque functionality and also advise on the significance of computation and communication overhead for authentication. They also focus on the development of lightweight and efficient privacy-preserving schemes in VANETs. Nandy et al. [14] focused on the enhancement of security in the VANET system by enlightening challenges, existing solutions, and future directions in intrusion detection systems.

Table 4.1 Security challenges and their description

Challenges	Description
Threats and vulnerabilities	Due to decentralized architecture, high node mobility, and an open wireless communication medium, VANETs face challenges such as denial-of-service attacks, spoofing, jamming, and Eavesdropping
Authentication	In a vehicular environment, authentication is achieved to identify authorized vehicles, nodes using public key infrastructure techniques, pseudonyms, symmetric keys, and hash-based schemes, but challenges such as frequent handoffs due to high mobility, balancing security and low latency for safety–critical applications, and trade-offs between privacy and traceability remain significant
Trust management	To examine the credibility of the nodes and transmitted information to protect from malicious codes, but challenges such as maintaining trust history, dealing with collusion attacks, and fast decision-making with limited resources remain significant
Privacy preservation	Due to the fast message exchange between vehicles and roadside units (RSUs), challenges such as low latency requirement, scalability, resource constraints, and trade-off between privacy and traceability remain significant
Intrusion detection	ML and DL techniques are used for monitoring, detecting, and mitigating malicious behaviour in the vehicular environment, but challenges such as false alarms, missed attacks, dynamic topology, and real-time detection remain significant

4.6 Performance Challenges in VANETs

The VANET-based network faced performance-related issues due to dynamic network topology, quality of service, security, safety, and other parameters, as mentioned in Table 4.2. These parameters may serve as a research direction for improving VANET services.

4.7 Open Research Challenges

In recent years, VANETs have emerged as an effective and efficient way of enhancing intelligent transportation systems, bringing significant advancements to facilitate people. However, researchers still need to focus on some of the grand challenges in this domain, including but not limited to.

- **Integration with Next Generation Wireless Network**: The integration of VANETs with 5G/6G and beyond leads to better solutions for problems like scalability, interoperability, reliability, traffic management, and improving the smart transportation system.
- **Integration with Advanced Techniques**: The integration of VANETs with advanced techniques, such as blockchain and edge computing, improves data security, privacy, and decentralized decision-making. However, challenges such as high computational overhead, latency constraints, and seamless interoperability with the existing vehicular network remain significant. The researcher working in the area of VANETs must focus on the optimization of the blockchain protocol and edge resource management to ensure real-time, secure, and scalable VANET operations.
- **Energy Efficiency and Sustainability**: The design of an energy-efficient and sustainable vehicular network is a major concern for reducing power consumption and its impact on the environment. The key challenges, such as efficient management of resources for both RSUs and OBUs, support sustainable data processing under high mobility, and remain significant for the research community. The advancement of the VANETs includes green networking techniques, renewable energy [11], and intelligent resource allocation to achieve an eco-friendly and scalable VANET system.
- **Standardization and Interoperability**: In a heterogeneous VANET environment, standardization and interoperability are the prime concerns to establish an effective and seamless communication environment among vehicles and across different vehicular networks. The open research challenges include coordination among the communication protocols, enabling cross-vendor compatibility, and supporting multi-domain V2X integration. The extension should be focused on universal standards, a flexible framework, and an adaptable interface to ensure reliable, scalable, and interoperable VANET deployment.

Table 4.2 Performance challenges and their description

Challenges	Description
Reliability, high mobility and dynamic topology [15]	The high speed of the vehicle causes frequent changes in network topology, leading to link disruptions and reducing the reliability of data delivery and routing
Scalability and bandwidth management [16]	The presence of a large number of vehicles in urban areas makes it challenging to achieve efficiency in terms of routing, connectivity, and bandwidth on a large scale
Quality of service (QoS) [17]	Balancing QoS across diverse applications is difficult, as safety applications require low latency and high reliability, while non-safety applications require high throughput
Routing protocol challenges [18]	A position-based and a hybrid protocol face challenges in dense and sparse traffic, and also traditional MANET routing protocols are not suitable due to high mobility
Slow connectivity, signal propagation issues [19]	In rural or low-density areas, limited connectivity may cause packet losses and communication delays
Channel congestion and interference [17]	Channel access and minimizing interference are big issues in dense urban environments due to collisions, spectrum scarcity, and broadcast storms
Heterogeneous communication environments [20]	Integration of multiple communication technologies, such as Wi-Fi, DSRC, 5G, and LTE, ensures interoperability but increases the complexity of the network
Energy and resource limitation [21]	Due to the limited capability of Roadside Units (RSUs) and On-Board Units (OBUs) in VANETs, as complex computation tasks and large-scale communication demands can overwhelm these resources
Environmental factors [22]	Due to high-rise buildings and weather conditions, Vehicular networks' loss of signal strength reduces communication reliability, while high packet error rates degrade overall network performance
Latency [8]	Maintaining low latency in VANETs is a grand challenge as a quick response is required in case of an emergency or collision
Fairness and load balancing [21]	To ensure fairness in bandwidth allocation and effective load balancing is a great challenge in VANETs, as the traffic load may overload some channels and roadside units while leaving others underutilized

4.8 Conclusion and Future Directions

This study primarily provides an overview of Vehicular Ad Hoc Networks and highlights key challenges in security and privacy, as well as performance issues such as dynamic topology, high mobility, scalability, and traceability. In this way, this study provides a potential research opportunity for the researcher based on the existing literature review. The importance of VANETs in establishing smart cities can't be ignored as the number of vehicles increases exponentially in today's world. The extension of this study may be in several directions, including the integration VANET with edge computing, IOT, 5G/6G in the advancement of the intelligent transportation system. Another foreseen work of ensemble ML models and deep reinforcement learning into the VANETs for improving security and privacy by anomaly detection, traffic management ensuring reliability, low-latency communication, security, and safety.

References

1. Verma, K., Hasbullah, H., Saini, H.K.: Reference broadcast synchronization-based prevention to DoS attacks in VANET. In: Proceedings of the 7th International Conference on Contemporary Computing (IC3), pp. 270–275. IEEE, Noida (2014). https://doi.org/10.1109/IC3.2014.6897185
2. Mchergui, A., Moulahi, T., Zeadally, S.: Survey on artificial intelligence (AI) techniques for vehicular ad-hoc networks (VANETs). Veh. Commun. **34**, 100403 (2022)
3. Saad, M., Khan, M.K., Ahmad, M.B.: Blockchain-enabled vehicular ad hoc networks: a systematic literature review. Sustainability **14**(7), 3919 (2022)
4. Sohail, M., Latif, Z., Javed, S., Biswas, S., Ajmal, S., Iqbal, U., Raza, M., Khan, A.U.: Routing protocols in vehicular ad hoc networks (VANETs): a comprehensive survey. Internet Things **23**, 100837 (2023)
5. Dwivedi, S.K., Amin, R., Das, A.K., Leung, M.T., Choo, K.K.R., Vollala, S.: Blockchain-based vehicular ad-hoc networks: a comprehensive survey. Ad Hoc Netw. **137**, 102980 (2022)
6. Javed, A.R., Hassan, M.A., Shahzad, F., Ahmed, W., Singh, S., Baker, T., Gadekallu, T.R.: Integration of blockchain technology and federated learning in vehicular (IoT) networks: a comprehensive survey. Sensors **22**(12), 4394 (2022)
7. de Souza, J.A.R., Cavalcanti, E.R., Bezerra, T.S., Gomes, R.C.D.M.: VANETs' research overview updated: past, present and future. J. Braz. Comput. Soc. **30**(1), 380–393 (2024)
8. Saini, H.K., Jain, K.: Impact of adaptive velocity-based Hello intervals in UAS to sustainability: enhancing network performance through adaptive velocity-based hello intervals. In: Proceedings of the 15th International Conference on Contemporary Computing (IC3–2023), pp. 674–682. ACM, New York (2023). https://doi.org/10.1145/3607947.3608082
9. Ficzere, D., Varga, P., Wippelhauser, A., Hejazi, H., Csernyava, O., Kovács, A., Hegedűs, C.: Large-scale cellular vehicle-to-everything deployments based on 5G—critical challenges, solutions, and vision towards 6G: a survey. Sensors **23**(16), 7031 (2023)
10. Yusuf, S.A., Khan, A., Souissi, R.: Vehicle-to-everything (V2X) in the autonomous vehicles domain: a technical review of communication, sensor, and AI technologies for road user safety. Transp. Res. Interdiscip. Perspect. **23**, 100980 (2024)
11. Saini, H.K.: GeNp-ODHR: Green energic and smart network performance estimation-based optimized deep Hello routing for flying ad hoc network. SAE Int. J. CAV **8**(3), 319–335 (2025). https://doi.org/10.4271/12-08-03-0022

12. Saini, H.K., Jain, K.: Implementation of DEWMA-based hello packet for AODV to improve the performance of FANET with 3D-GMM. SN Comput. Sci. **5**, 321 (2024). https://doi.org/10.1007/s42979-024-02673-z
13. Mundhe, P., Verma, S., Venkatesan, S.J.C.: A comprehensive survey on authentication and privacy-preserving schemes in VANETs. Comput. Sci. Rev. **41**, 100411 (2021)
14. Nandy, T., Noor, R.M., Kolandaisamy, R., Idris, M.Y.I., Bhattacharyya, S.: A review of security attacks and intrusion detection in vehicular networks. J. King Saud Univ. Comput. Inf. Sci. **36**(2), 101945 (2024)
15. Tan, S., Zong, X., Xiao, F.: Dynamic topology adaptability of adaptive multimodal transmission strategy based on environment perception in multi-hop routing of 5G vehicle network. Discov. Internet Things **5**, 36 (2025). https://doi.org/10.1007/s43926-025-00138-0
16. Zhou, Z., Roncoli, C.: A scalable vehicle assignment and routing strategy for real time on-demand ridesharing considering endogenous congestion. Transp. Res. C Emerg. Technol. **139**, 103658 (2022)
17. Abosaif, A.N., Hamza, H.S.: Quality of service-aware service selection algorithms for the internet of things environment: a review paper. Array **8**, 100041 (2020). https://doi.org/10.1016/j.array.2020.100041
18. Bilal, S.M., Bernardos, C.J., Guerrero, C.: Position-based routing in vehicular networks: a survey. J. Netw. Comput. Appl. **36**(2), 685–697 (2013). https://doi.org/10.1016/j.jnca.2012.12.023
19. Deutsch, A., et al.: High-speed signal propagation on lossy transmission lines. IBM J. Res. Develop. **34**(4), 601–615 (1990). https://doi.org/10.1147/rd.344.0601
20. Dutta, A., Campoverde, L.M.S., Tropea, M., De Rango, F.: A comprehensive review of recent developments in VANET for traffic, safety and remote monitoring applications. J. Netw. Syst. Manag. **32**(4), 73 (2024)
21. Wu, Y., Wu, J., Chen, L., Yan, J., Han, Y.: Load balance guaranteed vehicle-to-vehicle computation offloading for min–max fairness in VANETs. IEEE Trans. Intell. Transp. Syst. **23**(8), 11994–12013 (2022). https://doi.org/10.1109/TITS.2021.3109154
22. Zou, X., Sun, L., Lan, T., Fan, C., Liu, S., Zhao, H., Qiu, J.: The effects of weather factors on road traffic casualties: analysis on provincial panel data of China from 2006 to 2021. Heliyon **10**(17), e36788 (2024). https://doi.org/10.1016/j.heliyon.2024.e36788

Chapter 5
Analysis of Hybrid FL Models for Intrusion Detection in Decentralized Software-Defined Networks



Aparna Kumari, Pooja Chaturvedi, and Deepika Bishnoi

Abstract The central control, programming capability and adaptive traffic management offered by SDN has transformed the traditional network architectures. Despite the advantages offered by centralized SDN organization, it also exposes the network to advanced security challenges such as Distributed Denial of Service (DDoS), spoofing and zero-day attacks. The centralized data collection and analysis provided by the traditional intrusion detection systems increases the risk related to single point of failure, scalability and privacy issues. These concerns can be addressed by incorporating the FL paradigm which allows the distributed processing of network traffic data without the need of centralized. In this chapter a hybrid FL model combining the five supervised machine learning models based on Logistic Regression, KNN, Random Forest, MLP and XGBoost is proposed for the efficient and robust intrusion detection for a distributed environment found in SDN. The proposed approach includes an essential preprocessing step such as class balancing using the SMOTE method. The implementation result show that the hybrid model based on tree based and neural network classifiers such as XGBoost, MLP and Random Forest achieves superior performance.

Keywords Software defined networking · Intrusion detection · Federated learning · Denial of service attack · Security

A. Kumari · P. Chaturvedi (✉) · D. Bishnoi
Institute of Technology, Nirma University, Ahmedabad, Gujarat, India
e-mail: pooja.chaturvedi@nirmauni.ac.in

A. Kumari
e-mail: aparna.kumari@nirmauni.ac.in

D. Bishnoi
e-mail: deepika.bishnoi@nirmauni.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_5

5.1 Introduction

Network management using the Software-Defined Networking (SDN) is a revolutionary phenomenon. In this, a single controller is in charge of the network's operation by separating the data plane from the control plane. This organization can be considered analogous to having a central control center for a city's traffic light management and it provides easier and flexible network management [1]. The centralized organization has a limitation too, that it is the major target for attack and in the case of a controller failure, the entire network can collapse [2]. The distributed denial of service attack, which disrupts the service of the controller is often considered as biggest threat in SDN environment. SDN is often used in critical infrastructures like smart devices or data centers, these attack presents an increasing risk and therefore inclusion of intrusion detection is a crucial step to have the networks operating seamlessly [3].

In the traditional intrusion detection systems used in SDN, the task of data collection and analysis often depends on a central hub. This approach is well suitable in few cases but the scenarios in which there is a decentralized organization of SDN privacy of data, speed and scalability are major concern. In such FL based model training can be found as effective. The FL algorithm requires the devices to train the detection models locally and it share only learned information and not the raw traffic data needs to be shared [4]. In these cases, each device provides a local security guard, which collaborate with each other without exposing sensitive network details, hence it is well suited for distributed SDN environments. The behavior of the FL algorithms is different in terms of network conditions and speed [5]. The objective of intrusion detection systems is to determine the most suitable approach for mitigating the threats in SDN while adopting against the real-life challenges such as bandwidth constraint and sneaky adversaries.

5.1.1 Motivation

The FL based approaches for intrusion detection seems promising in the SDN environment, but there exists a significant research gap in the field. Most of the approaches proposed in the literature have considered the homogeneous SDN environment and it includes limited attack scenarios and ignores the inherent complexities cite [6, 7]. The significant work needs to be done in the area of benchmarking of the FL based algorithms in the realistic DoS attack scenarios such as stealthy attack or DoS attacks. The trade-offs between accuracy of detection, efficient communication and robustness against adversarial manipulations in SDN contexts which leaves uncertainty about the practical viability of FL-based intrusion detection systems need to be addressed [8]. This chapter aims to address these gaps by extensively evaluating FL algorithms in diverse and real-world-inspired SDN organizations.

5.1.2 Chapter Organization

The chapter is organized as follows: related work in Sect. 5.2, methodology in Sect. 5.3, federated learning-based framework in Sect. 5.4, implementation details and the discussion of results is presented in Sect. 5.5 and chapter concludes with Sect. 5.6 with conclusion and future work.

5.2 Related Work

Intrusion detection in SDN has attracted great research focus and the recent work done in the field shows the transition from centralized SDN organization to distributed federated learning approaches. The earlier approaches used for Intrusion detection in SDN environment were mostly centralized one and uses the statistical or signature-based analysis for the classification of network traffic. Although such intrusion detection system offers high detection accuracy in controlled network environment, but it also increased the latency and is prone to the risk of single point of failure, hence cannot be deployed in the real-world environment. The machine learning and deep learning approaches improved the detection accuracy due to ability of identification of complex spatial and temporal attack features. The supervised machine learning models have a limitation in terms of requirement of the large-scale annotated dataset and it requires the repository of the raw network traffic on a central point which induces the privacy and compatibility restrictions- particularly in the multi-organizational or cross domain SDN emulations. This arises a need for the shift towards the privacy preserving learning approaches. In [9], authors have FL based multiclass IDS for SDN to increase the confidentiality of data, demonstrate the improved scalability but it is limited by the interpretability of the model specific feature. In [10], the deep FL based IDS is proposed for the IOT enabled SDN based on IPFS for enabling the communication exchange in a secure manner by reducing the risk involved in communication and it also achieves high detection accuracy. In [11], authors have proposed XfedHunter algorithm for the detection of advanced threats in SDN based on graph neural networks and focuses on transparency over the throughput maximization. In [12], continual FL based IDS is proposed and it supports incremental learning of increasing attack-based signatures and can adopt to temporal distribution variation. In [13], authors have proposed F-NIDS approach to reduce the communication cost and converge is improved in the cases of irregular client applications. The summary of the work done in the field of intrusion detection is as shown in the Table 5.1.

5.3 Methodology

The methodology is represented in the Algorithm 1.

Table 5.1 Summary of literature review

Sr. no	Refernece	Architecture and approach	FL type and aggregation	Dataset	Performance	Advantages and constraints
1	Raza et al. [9]	FL-based multiclass classification for SDN	Centralized FL, multiclass output	Realistic cybersecurity dataset	Compared against state-of-art; improved privacy and scalability	Avoids centralizing data, scalability enhanced security but model specifics and metrics not disclosed
2	IPFS-FL NIDS on multi-controller SDN (2025) [10]	Deep learning NIDS over SDN-IoT with IPFS for model sharing	Likely centralized FL with IPFS	Custom "IoT-SDN" dataset	Accuracy: 99.89%, loss: 0.005	Very high accuracy, encrypted parameter sharing; custom dataset—generalizability may need validation on standard benchmarks
3	XFedHunter (2023) [11]	Explainable FL framework for APT detection in SDN	FL with GNN + local models; explainable	NF-ToN-IoT and DARPA TCE3 datasets	Not quantified in summary; emphasizes trust and accountability	Handles APT threats, improves model interpretability, tested on recognized datasets; performance metrics not reported
4	Continual Federated Learning (CFL 2025) [12]	SDN-based cloud IDS using continual FL	Continual FL across cloud-SDN architecture	SDN-based cloud environment	Not specified	Supports evolving threats over time; built for SDN cloud lacks disclosed results or dataset details
5	F-NIDS (2023) [13]	NIDS using federated AI and asynchronous communication	Federated AI + asynchronous communication	Not specified	Not specified	Reduces communication overhead; flexible coordination; lacking performance data and specifics
6	General FL-IDS insights [12]	Framework-level comparisons across domains (incl. SDN)	Centralized, decentralized, hybrid	IoT, CPS, SDN, heterogeneous domains	Varies by system; metrics differ widely	Highlights FL architectures, challenges (non-IID data, aggregation, resource constraints), and evaluation metrics consistency issues across studies

5.3.1 Dataset Overview

In this chapter we have utilized the dataset available in [6]. The dataset was generated by creating the emulation environment created using Mininet. The several SDN topologies were created using open flow switches and host. The switches were able to communicate via the SDN controller implemented using Ryu. The simulation was run over ~250 min and data were generated in the form of network traffic. The generated traffic were sampled in the monitoring interval of 30 s and the statistics were collected from the switches and ports. Through the simulation few features were extracted directly and few were derived from the existing features. The examples of direct features are switch id, packet count, byte count, duration and port statistics and derived features are packet rate, total duration, byte per flow, throughput in kbps. The dataset consists of 104,345 rows and 23 columns. The label column is considered as the target column which is used to classify the traffic as normal or malicious.

5.3.2 Data Cleaning and Preprocessing

After collecting the dataset, the several preprocessing methods were applied. The missing values present in the rx_kbps and tot_kbps were imputed with the mean values. The z-score method is used to remove the outliers in the dataset. The numerical and categorical features were determined. The one hot encoding method is applied on the categorial attributes such as src, dest and Protocol.

5.3.3 Data Visualization

The count of instances in every class were plotted and it is seen from the plot that the dataset is skewed as can be seen in the Fig. 5.1.

5.3.4 Data Splitting and Balancing

In the next step the dataset is divided into 70% training and 30% testing along with the stratify parameter to preserve the class ratio. SMOTE oversampling method for the minority class is applied on the training dataset to make the dataset balanced. After that StandardScaler is applied to normalize the features to zero mean and unit variance. This method is crucial to achieve the convergence of the models such as Logistic Regression, MLP and KNN.

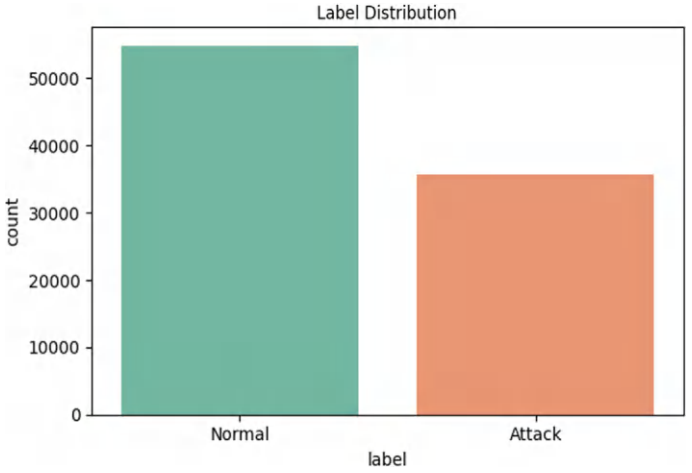


Fig. 5.1 Class wise distribution of traffic

5.3.5 Model Development Using FL Framework

For the implementation of FL models in combination with several supervised learning models, the training dataset is partitioned into 3 clients which are executed locally.

5.3.6 ML Models Used

Five supervised machine learning models were employed for training within the federated learning (FL) framework. Logistic Regression was used as a baseline linear classifier to establish reference performance. Random Forest, a tree-based ensemble method, was selected due to its robustness against noise and its ability to handle complex feature interactions. K-Nearest Neighbors (KNN) is a distance-based classifier that assigns class labels based on majority voting among the nearest neighbors in the feature space. XGBoost, a boosting-based ensemble model, was included because of its strong performance on structured datasets and its capability to capture complex decision boundaries. Multilayer Perceptron (MLP), a neural network–based model, was utilized to identify nonlinear patterns present in the dataset.

The FL model training was simulated over five communication rounds. In each round, every client trained a local model using its respective partition of the training dataset. The locally trained models then generated predictions on a shared global test set. Unlike the conventional FedAvg approach, which aggregates model parameters, the proposed framework performs aggregation at the prediction level by averaging the outputs generated by each client. The final prediction of the FL-based hybrid model is

obtained by averaging the predictions across all clients and across all communication rounds.

5.3.7 Metrics and Performance Evaluation

The models are evaluated on the basis of following metrices- Accuracy, Precision, Recall, F1 Score, AUC. The confusion matrix is also plotted for each model to determine the rate of true positive and false positive.

5.4 Federated Learning Framework

Algorithm 1
Algorithm: Federated Learning–Based Intrusion Detection
Input: Raw SDN dataset D
Output: Global intrusion detection model M_global with performance metrics Step 1: Data Preprocessing - Load dataset D - Handle missing values using median imputation - Remove outliers using Z-score filtering (|z| >= 3) - Encode categorical features (src, dst, protocol) using one-hot encoding Step 2: Dataset Preparation - Split dataset D into training set D_train and test set D_test - Apply SMOTE on D_train for class balancing - Normalize features using z-score standardization Step 3: Federated Data Partitioning

- Partition D_train into N subsets (D1, D2, …, DN)

- Assign each subset Dc to client c

Step 4: Federated Learning Procedure

- Initialize global model M_global with random weights

- For each round r = 1 to R do

* Server broadcasts M_global to all clients

* For each client c = 1 to N in parallel

- Receive global model M_global

- Train local model Mc on client dataset Dc

- Send updated local parameters θc to server

* Server aggregates updates using weighted averaging:

$$\theta_global = (1/N) * \Sigma\ \theta c \quad \text{for } c = 1 \text{ to } N$$

Update M_global ← θ_global

Step 5: Evaluation

- Test M_global on D_test

- Compute Accuracy, Precision, Recall, F1-score, and AUC

- Plot confusion matrix and ROC curve

Step 6: Comparison and Analysis

- Repeat procedure for Logistic Regression, Random Forest, KNN, XGBoost, and MLP

- Compare results and identify best-performing model under FL setup

5.5 Results and Analysis

The Hybrid FL models using Logistic Regression, Random Forest, KNN, XGBoost and MLP were implemented on the dataset. All the simulations were done using the python programming language and models were implemented using the inbuilt libraries such as pandas, numpy, matplotlib and sklearn on a 64-bit system with 11th Gen Intel ® Core ™ i5-1135G7 processor, frequency 2.4 GHz and 8 GB RAM. The results of the implemented models are as shown in the Figs. 5.2 and 5.3.

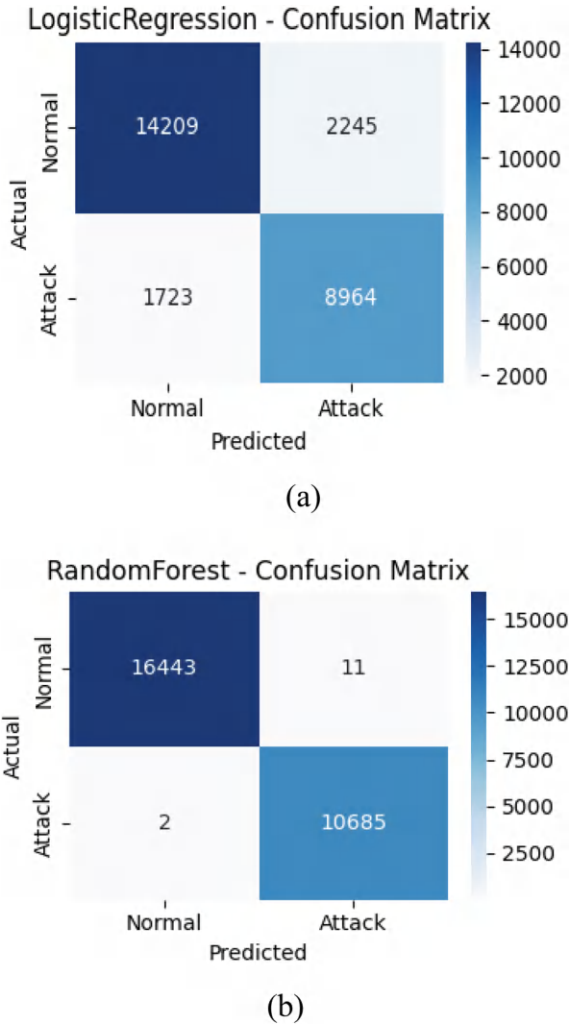
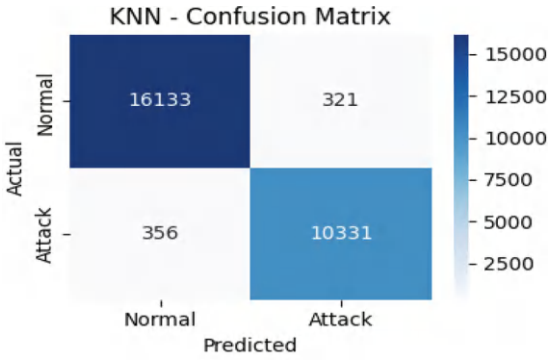
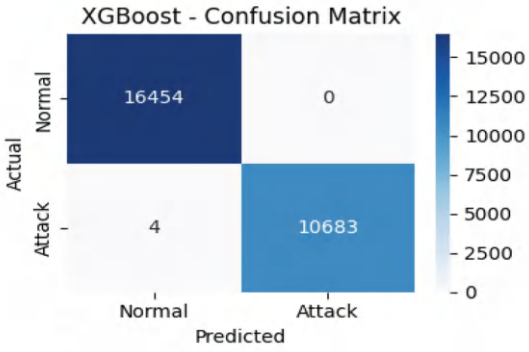


Fig. 5.2 Confusion matrix for the models. **a** Logistic regression, **b** random forest, **c** KNN, **d** XGBoost, **e** MLP

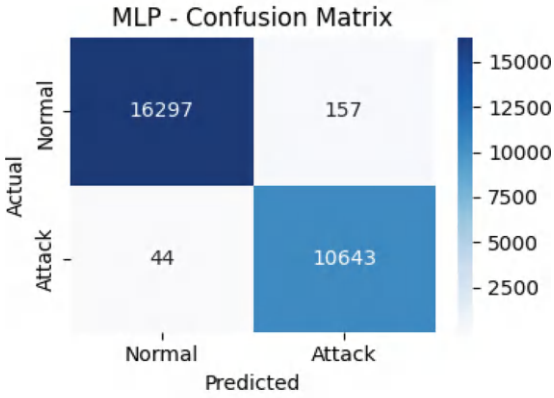
Fig. 5.2 (continued)



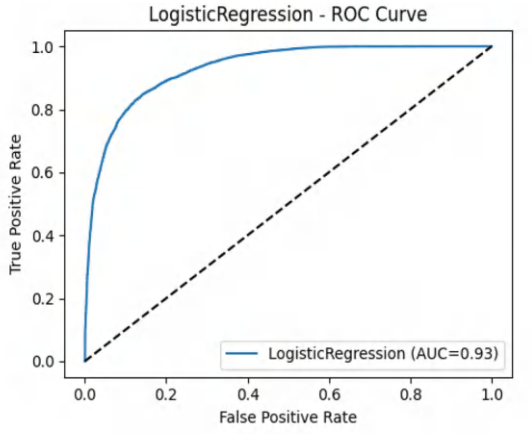
(c)



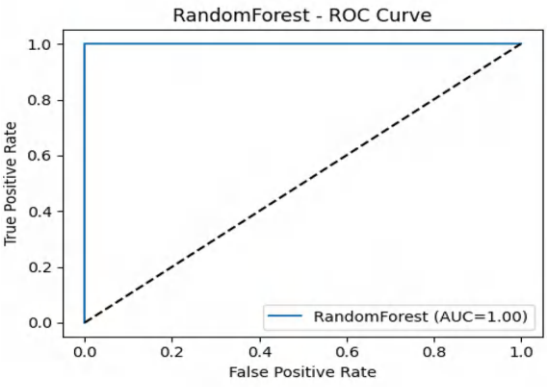
(d)



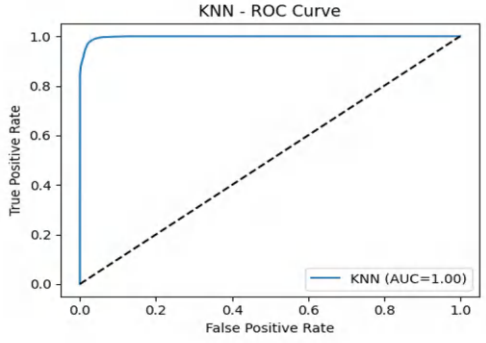
(e)



(a)



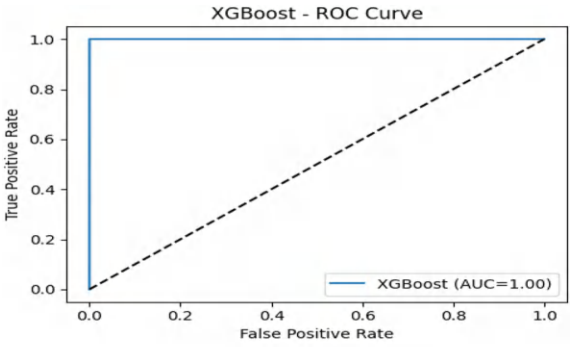
(b)



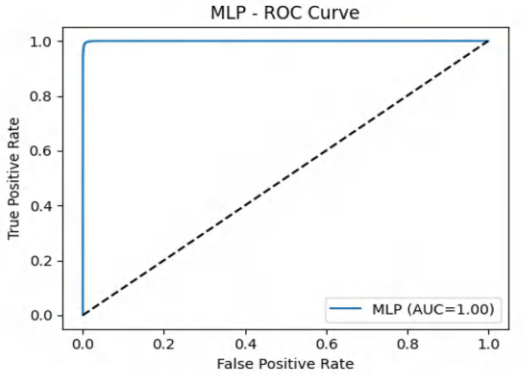
(c)

Fig. 5.3 ROC curves for the models. **a** Logistic regression, **b** random forest, **c** KNN, **d** XGBoost, **e** MLP

Fig. 5.3 (continued)



(d)



(e)

The summary of the all the performance metrices in shown in the Table 5.2 for the different models. From the results it can be seen that Random Forest, XGBoost and MLP achieves the best accuracy in the Federated conditions.

The performance comparison of the all the models on the basis of different performance metrics is as shown in the Fig. 5.4. From the results it can be seen by integrating the FL model with Random Forest, XGBoost and MLP achieves the best accuracy

Table 5.2 Performance comparison of models

Model	Accuracy	Precision	Recall	F1 score	AUC
Logistic regression	0.85	0.79	0.83	0.81	0.93
Random forest	0.99	0.99	0.99	0.99	1.00
KNN	0.97	0.96	0.96	0.96	0.99
XGBoost	0.99	1.00	0.99	0.99	1.00
MLP	0.99	0.98	0.99	0.99	0.99

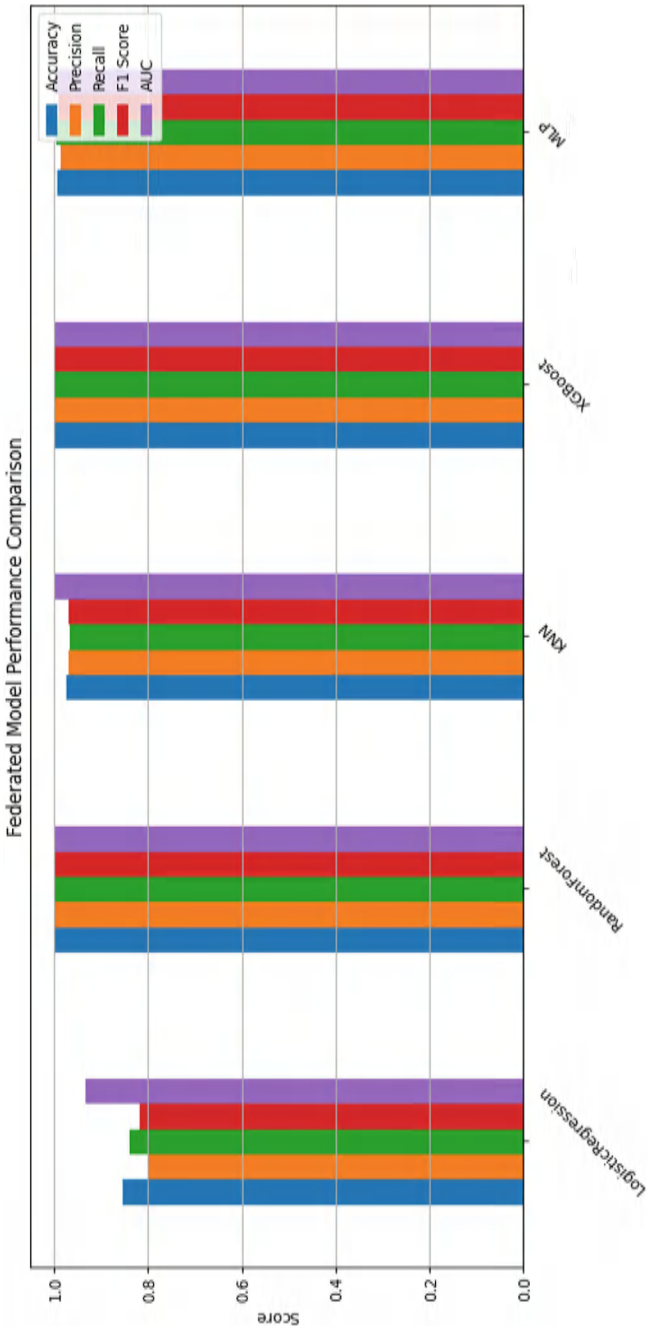


Fig. 5.4 Performance metrics comparison of the considered models

in fewer number of iterations. The logistic regression gives the worst performance under the federated environment.

5.6 Conclusion and Future Scope

The chapter proposes a robust, privacy-preserving intrusion detection framework for SDN using FL and a multi-model ensemble of supervised machine learning classifiers. We have also applied SMOTE to address the common challenge in intrusion detection as class imbalance. The implementation results show that the hybrid federated and ensemble not only exhibits high detection performance but also outperforms the individual classifiers in terms of metrics such as Accuracy, Precision, Recall, F1 Score, and AUC. The future work of this work could be the integration of the proposed approach with real time SDN environment using stream based data collection and incremental update in existing models. The proposed FL based framework could be extended to heterogenous environment consisting of IOT, Cloud, Edge Networks) to handle diverse traffic data features and distributions. The feature selection approach can be integrated to evaluate the robustness of model in detection of intrusion.

References

1. Fedorchenko, E., Novikova, E., Shulepov, A.: Comparative review of the intrusion detection systems based on federated learning: advantages and open challenges. Algorithms **15**(7), 247 (2022). https://doi.org/10.3390/a15070247
2. Buyuktanir, B., Altinkaya, Ş, Baydogmus, G.K., Yildiz, K.: Federated learning in intrusion detection: advancements, applications, and future directions. Clust. Comput. **28**, 473 (2025). https://doi.org/10.1007/s10586-025-05325-w
3. Restuccia, F., D'Oro, S., Al-Shawabkeh, M.: Securing software-defined networking: challenges and opportunities. IEEE Commun. Surv. Tutor. **21**(4), 3469–3499 (2019)
4. Ahmad, I., Namal, S., Ylianttila, M., Gurtov, A.: Security in software defined networks: a survey. IEEE Commun. Surv. Tutor. **17**(4), 2317–2342 (2015)
5. Bhattacharjee, S., Talukder, A., Al-Shaer, E.: Lightweight intrusion detection for software-defined networking. IEEE Trans. Netw. Serv. Manag. **15**(3), 1056–1069 (2018)
6. Yan, Q., Yu, F.R.: Distributed denial of service attacks in software-defined networking with cloud computing. IEEE Commun. Mag. **53**(4), 52–59 (2016)
7. McMahan, H.B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, vol. 54, pp. 1273–1282 (2017)
8. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. Proceed. Mach. Learn. Syst. **2**, 429–450 (2020)
9. Raza, M., Saeed, M.J., Riaz, M.B., Sattar, M.A.: Federated learning for privacy-preserving intrusion detection in software-defined networks. IEEE Access **12**, 69551–69567 (2024). https://doi.org/10.1109/ACCESS.2024.3395997
10. Dhirar, H., Hamad, A.H.: Federated deep learning intrusion detection system on software defined-network based internet of things. IAES Int. J. Artif. Intell. **14**(4), 3109–3120 (2025)

11. Huynh, T.T., Ngo, D.H.S., Phan, T.D., Nghi, H.K., Ngo-Khanh, K., Pham, V.-H.: *XFedHunter: An Explainable Federated Learning Framework for Advanced Persistent Threat Detection in SDN* (2023). https://doi.org/10.48550/arXiv.2309.08485
12. Chetouane, A., Karoui, K.: New continual federated learning system for intrusion detection in SDN-based edge computing. Concurr. Comput. Pract. Exper. **37**(2), e8332 (2025). https://doi.org/10.1002/cpe.8332
13. Oliveira, J.A., Gonçalves, V.P., Meneguette, R.I., Sousa, R.T., Guidoni, D.L., Oliveira, J.C.M., Filho, G.P.R.: F-NIDS: a network intrusion detection system based on federated learning. Comput. Netw. **236**, 110010 (2023). https://doi.org/10.1016/j.comnet.2023.110010

Chapter 6
Performance Study of TCP in Vehicular Ad Hoc Networks



Prangya Priyadarshini, Putcha Sai Kiran, and Arun Kumar

Abstract Vehicular Ad Hoc Networks (VANETs) are a foundational technology for Intelligent Transportation Systems (ITS), enabling a new generation of safety, efficiency, and infotainment applications. While many of these applications depend on reliable, connection-oriented data delivery, the de facto standard, the Transmission Control Protocol (TCP), performs poorly in the vehicular domain. This performance degradation stems from a fundamental mismatch: TCP's congestion control algorithms were designed for stable, wired networks and incorrectly interpret the frequent, non-congestive packet losses caused by high mobility, wireless channel errors, and rapid topological changes as signs of network congestion. This chapter provides a comprehensive survey and performance analysis of TCP in VANETs. It systematically examines the performance of key TCP variants, from classic loss-based algorithms like Reno and Cubic to more adaptive, bandwidth-aware solutions like Westwood. Furthermore, the chapter investigates advanced adaptation strategies, including cross-layer designs that use feedback from lower protocol layers, the potential of Multipath TCP (MPTCP) to enhance connection resilience, and the integration with emerging paradigms such as edge computing and Named Data Networking (NDN). Synthesizing findings from recent empirical studies, it is concluded that while no single solution is universally optimal, context-aware protocols that can differentiate the root cause of packet loss consistently outperform their oblivious counterparts. Finally, the chapter identifies critical open research challenges and highlights future directions, emphasizing the need for AI/ML-driven optimization, security-aware transport mechanisms, and standardized benchmarking frameworks to guide the development of robust transport protocols for the next generation of connected and autonomous vehicles.

P. Priyadarshini · P. S. Kiran · A. Kumar (✉)
Department of Computer Science and Engineering, NIT Rourkela, Rourkela, India
e-mail: kumararun@nitrkl.ac.in

P. Priyadarshini
e-mail: 522cs3005@nitrkl.ac.in

P. S. Kiran
e-mail: 524cs1008@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_6

Keywords VANET · TCP · Cross-layer protocols · Named data networking (NDN) · Multipath TCP (MPTCP)

6.1 Introduction

The paradigm shift toward intelligent and autonomous transportation is radically altering current mobility. The Intelligent Transportation System (ITS) is at the heart of this revolution, a comprehensive architecture of advanced communication, sensor, and control technologies meant to improve road safety, traffic efficiency, and driving experience [1]. Vehicular Ad Hoc Networks (VANETs) are the fundamental communication backbone of ITS, allowing for a dispersed, self-organizing network between vehicles and fixedÂ infrastructure [2]. A VANET's architecture generally consists of On-Board Units (OBUs) in vehicles and Roadside Units (RSUs) distributed along highways, allowing for multiple modes of communication, including Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and, by extension, Vehicle-to-Everything (V2X) [3]. This complex communication web serves a wide range of applications, from safety-critical alerts concerning crashes and road dangers to non-safety services like real-time traffic updates, infotainment streaming, and large-scale software upgrades.

The performance of these applicationsÂ is heavily reliant on the underlying communication protocol stack. While safety messages frequently rely on lightweight, broadcast-based protocols, a large and rising number of non-safety applications require the dependable, connection-oriented data delivery provided by transport layer protocols [4]. The Transmission Control Protocol (TCP) has beenÂ the foundation for reliableÂ data transport across the Internet for decades . Its complex congestion management, flow control, and in-order delivery methods have made it the go-to solution for applications requiringÂ data integrity. As a result, TCP is an ideal choice for allowing services such as HD map downloads for self-driving cars, Firmware-Over-The-Air (FOTA) upgrades, and high-quality video streaming in the vehicle.

However, TCP's fundamental design principles contradict VANET's inherent characteristics. TCP's congestion control methods are based on the underlying assumption that packet loss is a direct signal of network congestion [5]. This assumption is valid in stable wired networks. In VANETs, it is consistently broken. The automotive environment is distinguished by high node mobility, frequent and unpredictable topological changes, intermittent connection, and variable wireless link quality as a result of signal fading, interference, and barriers [6]. In such a context, physical and link-layer dynamics are more likely to cause packet loss than router buffer overflows. When typical TCP variations like Reno or Cubic face non-congestive packet loss, they falsely activate congestion avoidance techniques, severely limiting their transmission window. This causes a series of performance difficulties, including significant throughput loss, increased end-to-end delay, and false retransmissions, eventually weakening the Quality of Service (QoS) essential for automotive applications [7].

This basic contradiction highlights the critical need to investigate, adapt, and improve TCP for reliable operation in VANETs. With the introduction of 5G/6G and Cellular-V2X (C-V2X) technologies, the potential data rates and connection possibilities for automobiles are rapidly increasing [8]. While these developments appear to relieve certain bandwidth and latency limits, they do not address the transport layer issues posed by high-speed mobility and ad hoc V2V connection instability. Ensuring that the transport protocol can intelligently discern between different sources of packet loss and adjust its behavior accordingly is critical to realizing the full potential of next-generation ITS. Recent research has centered on proposing unique TCP variations, cross-layer designs that exploit information from lower levels, and investigating alternatives such as Multipath TCP (MPTCP) to provide connection resilience [7]. A detailed knowledge of these dynamics is required for developing strong communication systems capable of meeting the complex needs of connected and autonomous vehicles.

6.2 Fundamentals of TCP

The Transmission Control Protocol (TCP) is a core protocol in the Internet suite that is designed to enable dependable, orderly, and error-free data transfer between applications. Its sturdy architecture has facilitated theÂ growth of the internet, but its operational assumptions are being challenged by contemporary, dynamic network topologies such as VANETs [9]. Understanding its underlying design and operations is critical for determining its performance limits in such contexts.

6.2.1 Brief Review of TCP Protocol Architecture

TCP establishes a logical, end-to-end connection between two hosts using Layer 4 (the Transport Layer) of the OSI and Internet protocol stacks. This stateful, connection-oriented design provides data integrity regardless of the underlying network's reliability [10]. A TCP session starts with a three-way handshake (SYN, SYN-ACK, and ACK) to synchronize sequence numbers and establish a connection. Once established, data is sent in a stream of bytes that TCP divides into packets. The connection is ended using a four-way handshake to ensure that both sides have finished transferring data.

TCP's resilience is based on three important principles that allow it to adapt to changing network conditions. It assigns sequence numbers to each byte of data received, allowing the receiver to reassemble the stream in the proper order and detect missing parts. The receiver uses acknowledgments (ACKs) to confirm data receipt. If an ACK for a certain segment is not received within a predefined Retransmission Timeout (RTO) period, the sender considers the segment lost and resends it [11].

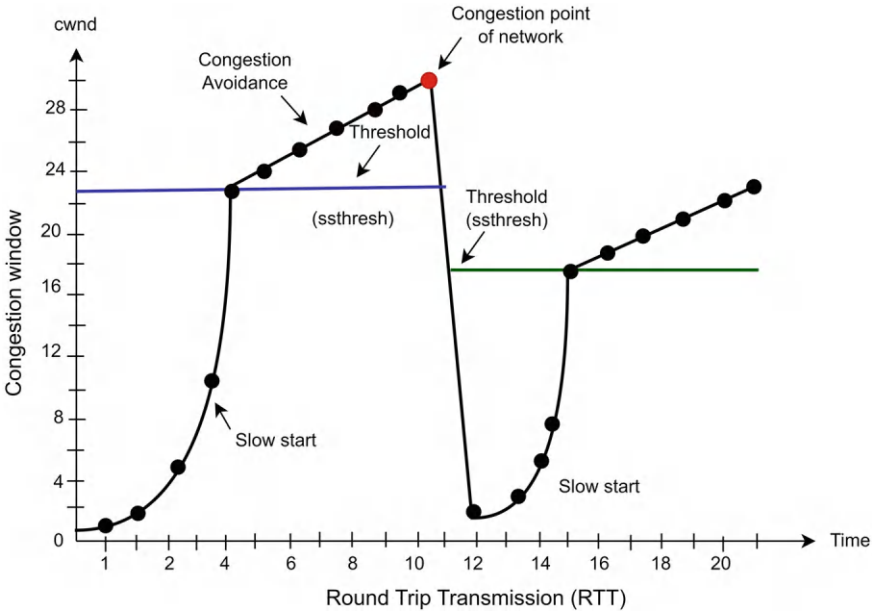


Fig. 6.1 Evolution of the TCP congestion window under AIMD congestion control [12]

This mix of sequencing, acknowledgments, and timeouts is the foundation of TCP's ability to provide consistent delivery over unreliable IP networks.

6.2.2 Congestion Control, Flow Control, and Reliability Mechanisms

The efficacy of TCP is regulated by three interrelated mechanisms: flow control, congestion control, and dependability.

- Flow Control: It is a receiver-centric method that prevents a fast sender from overloading a sluggish receiver. The TCP header contains the "receive window" (rwnd) field, which is used by the receiver to advertise available buffer space. The sender must guarantee that the quantity of unacknowledged data in flight does not exceed this limit, aligning its sending pace with the receiver's processing capabilities [13].
- Reliability: This is ensured by checksums, sequence numbers, and retransmission mechanisms. This approach is inherently associated with congestion control, as the initiation of retransmission (e.g., a timeout or multiple ACKs) serves as the principal indicator for identifying possible network congestion.

- Congestion Control: TCP congestion control is a sender-focused strategy designed to prevent overwhelming the network path during a connection; the sender maintains a congestion window (cwnd) that restricts the amount of data allowed in flight at any given time, representing the sender's estimate of the network's current capacity. The cwnd is continuously adjusted according to network feedback, primarily using the Additive Increase, Multiplicative Decrease (AIMD) algorithm, which promotes fairness and stability among competing flows. During the Congestion Avoidance phase, the cwnd grows linearly for each Round-Trip Time (RTT) under lossless conditions, but is reduced multiplicatively—typically halved—when packet loss signals congestion. The process begins with a Slow Start phase for new connections, exponentially increasing the cwnd until the onset of packet loss or a predefined threshold is reached. Fast Retransmit and Fast Recovery mechanisms allow the sender to promptly detect and correct localized packet losses (typically via triple duplicate ACKs), thus mitigating the need for slower RTO. Figure 6.1 illustrates typical TCP congestion window dynamics, from exponential growth in Slow Start, through linear Congestion Avoidance, to the sharp reduction on detecting congestion, and subsequent recovery. Such behavior exemplifies TCP's core strategy for network-friendly rate adaptation.

While these techniques are extremely effective for wired networks, they are commonly misactivated in dynamic situations such as VANETs. High bit error rates and mobility-induced link disconnections result in non-congestive packet loss, which TCP consistently misinterprets as an indication of network congestion. This causes an unnecessary and aggressive reduction in the cwnd, resulting in substantial performance loss and underutilization of available bandwidth [14].

6.2.3 Common TCP Variants

TCP has evolved into multiple variants, each seeking to enhance the congestion management algorithm for specific network conditions. Figure 6.2 illustrates this classification along with representative protocols.

- TCP Reno and New Reno: These are conventional loss-based variations that strictly adhere to the AIMD concept. Reno enters Fast Recovery after getting three duplicate ACKs but quits after receiving a fresh ACK, which might be inefficient if numerous packets were lost. New Reno improves on this by remaining in Fast Recovery until all outstanding data at the beginning of the recovery phase is acknowledged, making it more resilient to numerous packet losses [15]. Their biggest weakness is their inability to differentiate between congestive and non-congestive loss.
- TCP Vegas: Unlike loss-based techniques, Vegas is a proactive, delay-based variation. It calculates the predicted throughput and compares it to the measured throughput by tracking RTT. If the real rate differs considerably from the predicted rate, Vegas detects incipient congestion and changes the window size before

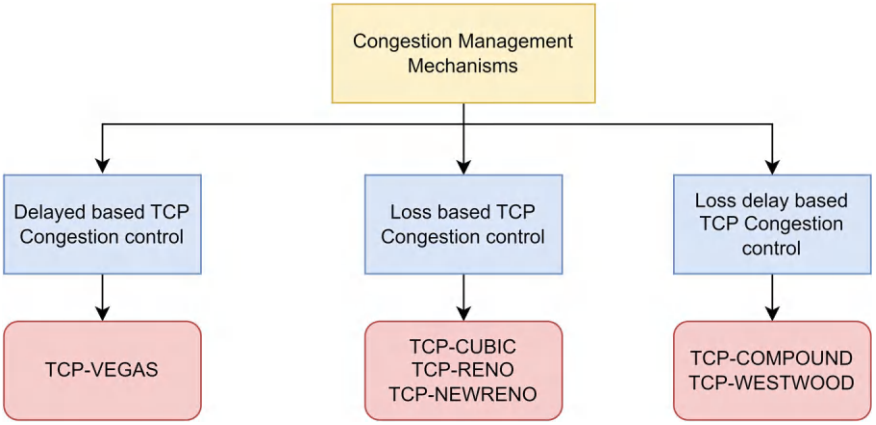


Fig. 6.2 Taxonomy of standard TCP congestion control mechanisms based on congestion detection signals

packet loss occurs [16]. While it is successful at avoiding congestion, it can be less aggressive and may cedeÂ bandwidth to other loss-based flows.

- TCP Cubic: Cubic is the default TCP in contemporary Linux kernels, and it is designed for high-speed, high-latency networks (hBDP). It uses a cubic function for window growth, allowing for quick expansion to fill available bandwidth following a congestion event. However, because it is still a loss-based algorithm, it shares Reno's basic weakness in situations with high non-congestive packet loss [17].
- TCP Compound: Compound TCP, created by Microsoft and the default on Windows systems, is a hybrid form that combines the standard loss-based AIMD technique with a delay-based component, similar in theory to Vegas. This enables it to be both aggressive in seeking bandwidth and sensitive to growing queueing delays, resulting in a balanced performance profile [18].
- TCP Westwood: This variation is especially intended to increase performance in wireless networks. Its main innovation is a sender-side bandwidth estimation system. After a packet loss, instead of immediately reducing the cwnd and ssthresh, Westwood sets these windows to a more acceptable value based on its estimation of available bandwidth. This makes the recovery procedure more intelligent and less punitive in the face of random, non-congestive wireless losses, making it a frequent benchmark for VANET studies [19].

The shift from basic loss-based algorithms to more complicated bandwidth- and delay-aware variations shows an ongoing effort to adapt TCP to the increasingly diverse and difficult circumstances of modern communication networks.

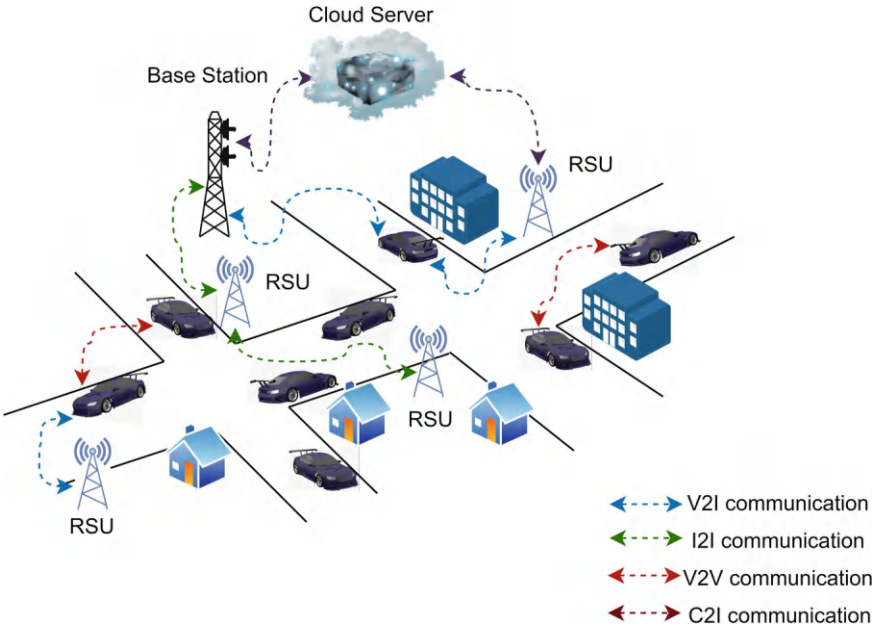


Fig. 6.3 VANET architecture and communication model

6.3 VANET Architecture and Communication Model

VANETs are the essential communication backbone for modern ITS, allowing crucial information to be exchanged in order to improve safety, efficiency and comfort [20]. These networks' architecture is naturally diverse and multi-tiered, with a focus on operating in extremely dynamic and demanding contexts. Understanding this architecture and underlying communication models is critical for developing and testing efficient transport layer protocols.

6.3.1 System Components

The VANET ecosystem is composed of several key entities that interact to facilitate seamless data flow as shown in Fig. 6.3. This architecture is increasingly evolving into a hierarchical model that integrates ad hoc communication with robust infrastructure support [21].

- Vehicles (On-Board Units—OBUs): Each participating vehicle is outfitted with an OBU, a complex communication device that has processing units, memory, sensors, and one or more wireless transceivers. The OBU is the network's principal mobile node, in charge of gathering data (such as speed, position, and hazardous

circumstances), processing it locally, and broadcasting or unicasting it to other entities.

- Roadside Units (RSUs): These are fixed communication nodes placed strategically along roadways, such as junctions, traffic signals, and highways. RSUs provide as a link between mobile ad hoc networks and fixed backhaul infrastructures. Their functions are diverse, including extending the transmission range of V2V communications, providing internet gateway access, disseminating local traffic or weather information, and acting as edge computer nodes for low-latency data processing and content caching.
- Base Stations and Access Points: In modern hybrid VANETs, cellular base stations (e.g., 4G LTE eNodeBs or 5G NR gNodeBs) are essential. They offer wide-area connection for Vehicle-to-Network (V2N) communication, allowing vehicles to stay connected even when they are out of range of RSU. This is critical for applications that need ongoing cloud connectivity, such as real-time navigation and telematics.
- Control Centers and Cloud Infrastructure: At the top of the hierarchy are centralized or distributed control centers, as well as cloud platforms. These organizations gather and consolidate massive volumes of data from vehicles and infrastructure to do large-scale traffic analysis, fleet management, and predictive modeling. They provide crucial updates, like as High-Definition (HD) maps or security fixes, back to the vehicle network.

6.3.2 Types of Communication

VANETs support several distinct communication modes, collectively referred to as Vehicle-to-Everything (V2X). The selection of technology to allow these modes is an important architectural decision, with two key standards dominating the landscape: IEEE 802.11p (part of the DSRC suite) and Cellular-V2X (C-V2X), which includes LTE-V2X and its development, 5G New Radio (NR)-V2X [22].

- Vehicle-to-Vehicle (V2V) Communication: This mode involves direct, ad hoc communication between vehicles without reliance on fixed infrastructure. V2V is critical for safety applications that need extremely low latency, including as collision alerts, emergency electronic brake lights, and cooperative maneuvering. Both DSRC and C-V2X (via their PC5/sidelink interfaces) offer direct V2V connections. However, C-V2X has shownÂ greater reliability and scalability, particularly in intensive traffic conditions [23].
- Vehicle-to-Infrastructure (V2I) Communication: V2I allows vehicles to communicate with RSUs and cellular base stations. This mode is required for traffic control (e.g., traffic signal phase and timing), internet connectivity, and local data services. For example, an RSU can give intersection safety information or offload massive information transfers, therefore lowering cellular network congestion.

- Vehicle-to-Everything (V2X) Communication: V2X is a broad word that includes V2V, V2I, and developing modes such as Vehicle-to-Pedestrian (V2P) and Vehicle-to-Network (V2N). 5G NR-V2X is intended to be the unifying technology for these many use cases, providing separate service classes such as Ultra-Reliable Low-Latency Communication (URLLC) for self-driving and improved Mobile Broadband (eMBB) for infotainment [24]. V2X's adaptability enables a wide range of applications, from safety warnings to collaborative data exchange for improved situational awareness.

6.3.3 Unique Features and Constraints of VANET Environments

The VANET environment has particular challenges that make it apart from conventional Mobile Ad Hoc Networks (MANETs) and classic wireless networks. These limitations have a direct and significant influence on the performance of transport layer protocols such as TCP.

The VANET environment is distinguished by a unique combination of high-speed yet constrained mobility and harsh, dynamic channel conditions that collectively undermine the performance of standard communication protocols. Unlike the random movement depicted in MANETs, vehicle mobility follows a known trajectory yet moves incredibly quickly, resulting in extremely short connection lifetimes and rapid, continuous changes in network structure. A steady multi-hop path can evaporate in seconds, resulting in many route breakages [25]. Compounding the problem, the wireless channel is notoriously unreliable; high vehicle speeds cause considerable Doppler shifts that impede signal reception, while urban environments present severe multipath fading and Non-Line-of-Sight (NLOS) circumstances owing to barriers. Furthermore, high node density can cause significant contention and interference at the MAC layer [26]. These physical and link-layer issues translate directly into transport-layer barriers, resulting in irregular, non-congestive packet loss, excessive jitter, and fluctuating RTTs. Standard TCP, unable to distinguish between the causes of these occurrences, misinterprets them as indications of network congestion. This misconception activates its aggressive avoidance algorithms, limiting throughput and resulting in low Quality of Service (QoS), which is undesirable for many real-time and data-intensive automotive applications. As a result, building robust transport protocols needs a thorough knowledge of this complicated cross-layer interaction, in which physical-layer dynamics drive transport-layer performance.

6.4 TCP Challenges in VANET

Despite its widespread use and stability in wired networks, the TCP encounters a slew of performance-degrading issues when applied in VANETs. Its basic design assumption, which equates packet loss with network congestion, is fundamentally incompatible with the dynamic and error-prone nature of the automotive environment. This section thoroughly investigates the main problems, detailing how mobility, wireless channel impairments, and network factors interact to cause substantial TCP performance constraints.

6.4.1 Mobility and Topology Dynamics

The defining characteristic of VANETs—high yet constrained vehicular mobility—is the primary antagonist to TCP's assumption of a stable end-to-end path. While vehicle traffic follows predictable route layouts, high relative speeds result in ephemeral connection lifetimes and a constantly changing network architecture. This fast churn directly compromises the key congestion management and dependability mechanisms of TCP. In multi-hop V2V communication, the stability of a TCP connection is determined by the stability of each intermediary link. A single link failure can sever the whole path, which often occursÂ on high-speed highways or at complicated urban intersections [27].

This route instability causes a number of major concerns. First, the time it takes for an underlying ad hoc routing protocol (such as AODV) to identify a connection loss and build a new route frequently surpasses TCP's RTO period. As a result, TCP suffers an erroneous timeout, assumes severe congestion, and significantly decreases its congestion window (cwnd), frequently returning to the slow-start phase [28]. This leads to protracted periods of transmission idleness and significant underutilization of the available channel. Second, frequent re-routing can result in considerable packet reordering, which may activate TCP's Fast Retransmit mechanism needlessly by creating repeated acknowledgments, resulting to incorrect cwnd reductions [29]. As a result, the TCP end-to-end dependability model is unsuitable for the VANET topology's hop-by-hop instability.

6.4.2 Wireless Channel Conditions and Error Characteristics

The vehicular wireless channel is notoriously harsh, and it accounts for a significant portion of non-congestive packet loss. TCP is fundamentally incapable of distinguishing between packet loss induced by network congestion and loss caused by the poor qualities of the wireless medium [30]. Important channel-induced issues include:

- Fading and Shadowing: The presence of buildings, trees, and other vehicles produces a complicated propagation environment with multipath fading and shadowing. This results in a high Bit Error Rate (BER) and intermittent packet corruption or loss, especially in congested urban Non-Line-of-Sight (NLOS) environments.
- Doppler Shift: High vehicle speeds create considerable Doppler changes in carrier frequency, which can decrease receiver performance and increase packet error rates, particularly on highways.

When TCP encounters packet loss from these phenomena, it incorrectly invokes its congestion control algorithms. Studies comparing urban and highway contexts consistently reveal that misunderstanding is a major driver of poor TCP performance [31]. While RSU-assisted networks can alleviate these impacts by decreasing the wireless path and ensuring a more robust link, solely ad hoc V2V connections are still very susceptible. Without a cross-layer method to notify TCP of the underlying channel condition, it will continue to perform suboptimally, aggressively limiting its transmitting rate in reaction to ambient variables rather than real network saturation.

6.4.3 Network Density, Transmission Power, and Node Distribution

The spatial distribution and density of vehicles provide a highly dynamic network environment, which has a direct influence on TCP performance. Simulation-based research, utilizing tools like NS-3 and SUMO, consistently indicate two extreme but typical scenarios:

- Dense Network: In urban traffic jams or at busy intersections, excessive vehicle density causes strong conflict at the Medium Access Control (MAC) level. This causes frequent packet collisions and buffer overflows at the link layer, resulting in packet loss to TCP. TCP misinterprets contention-based loss as congestion, resulting in low performance when the network is highly connected [32]. Transmission power exacerbates this problem; a longer transmission range increases the likelihood of connection while simultaneously expanding the interference domain, resulting in more collisions.
- Sparse Network: Low vehicle density on rural roads or during off-peak hours causes the network to become fragmented or partitioned. Maintaining a steady, end-to-end multi-hop route becomes challenging or impossible. In such cases, TCP connections commonly fail owing to chronic route unavailability, making reliable data transmission impossible without infrastructure support.

This illustrates a critical trade-off where TCP performance is poor at both extremes of network density, highlighting its lack of adaptability to the fluctuating spatial characteristics inherent to VANETs.

6.4.4 TCP Performance Bottlenecks

The issues outlined above lead to numerous well-documented TCP performance bottlenecks. The overall issue is a misconception of non-congestion loss, which sets off a vicious loop of bogus retransmissions and excessive cwnd decreases, resulting in unstable and substantially reduced throughput. Specific bottlenecks include:

- High Packet Loss and Low Throughput: The combination of mobility-induced route failures and wireless channel defects causes a high packet loss rate, forcing TCP to spend the majority of its time in congestion avoidance or recovery modes, seldom attaining an ideal sending rate.
- Inflated End-to-End Delay and Jitter: Frequent route modifications and link-layer retransmissions cause considerable and highly variable delays, rendering RTT calculation unreliable and resulting in either premature or delayed retransmissions.
- Inefficiency of TCP Variants: According to comparative evaluations, conventional loss-based variations such as TCP Reno and Cubic perform particularly badly due to their severe window reduction strategies [33]. While bandwidth-estimation versions such as TCP Westwood increase robustness by employing a more sophisticated post-recovery window configuration, they still face the core problem of loss distinction. Delay-based variations, such as TCP Vegas, can be more reliable, but they are frequently overly cautious, failing to maximize available bandwidth use.

Finally, TCP's inability to modify its behavior based on the underlying cause of packet loss remains the primary outstanding difficulty. Without major change or cross-layer support, TCP will remain a performance constraint for dependable data applications in VANETs.

6.5 TCP Variants and Adaptations for VANET

The intrinsic mismatch between TCP's congestion-centric architecture and the numerous sources of packet loss in VANETs has sparked substantial study into protocol adjustments. Standard TCP variations are not designed to deal with the high mobility, intermittent connection, and wireless channel problems that are common in automobile contexts. As a result, a wide range of solutions have been proposed, ranging from minor improvements to existing TCP algorithms to comprehensive cross-layer designs. This section compares different techniques, analyzes their effectiveness using important performance measures, and synthesizes the findings to suggest future study possibilities.

6.5.1 Comparative Study of Key TCP Variants

Numerous studies have benchmarked the performance of common TCP variants in simulated VANET scenarios, revealing distinct performance profiles.

- **TCP Reno and New Reno**: Reno and New Reno, the fundamental loss-based variations, use the Additive Increase, Multiplicative Decrease (AIMD) algorithm to reduce the congestion window (cwnd) by half when packet loss is detected. VANETs' performance is regularly rated low [34]. They are unable to discern between non-congestive losses and genuine congestion, resulting in frequent and needless cwnd decreases, significant throughput degradation, and underutilization of the available wireless channels. When compared to the intermittent, high-volume packet loss that is frequent in vehicle networks, New Reno's better recovery method provides only minimal gains.
- **TCP Vegas**: This delay-based variation proactively avoids congestion by monitoring changes in RTT. It maintains a small packet queue in the network and reduces its sending pace if RTT exceeds a particular threshold. While this method can decrease buffer bloat and packet loss in stable networks, its efficacy on VANETs is mixed. The significant RTT variance created by dynamic multi-hop routing might confound its estimate algorithm, and it is frequently discovered to be overly cautious, unable to compete successfully for bandwidth against more aggressive loss-based flows.
- **TCP Cubic**: Cubic, the current default in Linux kernels, uses a cubic function for window expansion, making it extremely scalable and aggressive in high Bandwidth-Delay Product (BDP) networks. However, because it is primarily a loss-based algorithm, it has the same basic constraint as Reno in VANETs. Comparative tests repeatedly reveal that, while Cubic can achieve high throughput in stable V2I networks, its performance suffers significantly under high mobility and wireless error rates, since it aggressively closes its window in reaction to non-congestive events [17].
- **TCP Westwood**: This variation improves TCP New Reno by including a sender-side bandwidth estimate method. Instead of naively halving the cwnd in the case of a loss, Westwood configures the control windows depending on its current estimate of available bandwidth. This "smarter" recovery eliminates a significant fall in transmission rate owing to random wireless losses. As a result, TCP Westwood is frequently claimed to outperform Reno, New Reno, and Cubic in VANETs, providing more throughput and faster recovery from non-congestive loss events [35]. Its key constraint is the accuracy of its bandwidth estimation in the presence of quickly changing channel circumstances.

6.5.2 Cross-Layer Approaches and Hybrid Methods

Recognizing that TCP lacks visibility into the lower layers of the protocol stack, cross-layer designs aim to provide TCP with explicit feedback about the network state. This allows TCP to make more informed decisions, fundamentally addressing the problem of loss differentiation.

The primary premise of cross-layer solutions is for the network, MAC, or physical layers to alert the transport layer of non-congestive events. Techniques such as TCP-ELFN (Explicit Link Failure Notification) use input from the routing protocol (e.g., AODV). When a route break occurs, the routing agent transmits an ELFN message to the TCP sender. When the sender receives this signal, it freezes its cwnd and retransmission timers and periodically retransmits the lost packet until a new route is established. This prevents TCP from improperly initiating congestion control [36]. Other systems use MAC-layer information about collisions or physical-layer measures such as Signal-to-Noise Ratio (SNR) to determine network quality and influence TCP behavior. CL-TCP, a recent example, employs channel status information to adaptively change TCP's cwnd, resulting in considerable throughput increases over regular variations [14].

Although theoretically robust, cross-layer designs provide substantial practical challenges. They contradict the OSI model's rigorous layering rules, necessitating changes to the operating system kernel and protocol stack. This makes them difficult to standardize and implement globally. Their efficacy is also heavily dependent on the accuracy and timeliness of feedback from lower tiers. RSU support can help such systems by serving as a solid coordination point, although pure V2V implementations remain difficult.

6.5.3 QoS Evaluation Metrics and Comparative Table

The performance of TCP variations and adaptations in VANETs is measured using a defined set of QoS metrics. These include throughput (goodput), end-to-end latency, Packet Delivery Ratio (PDR), jitter, and connection reliability. The table below highlights the frequently stated performance characteristics of main TCP variations in comparison to these measures in a typical VANET context (Table 6.1).

6.6 Advanced Topics

While modifications to TCP's core congestion control algorithms represent a foundational step, the evolution of vehicular networks necessitates exploring more advanced and holistic solutions. The future of reliable transport in VANETs is intrinsically linked to addressing security vulnerabilities, leveraging multi-path communication,

Table 6.1 Comparative performance of TCP variants in VANETs

TCP variant	Congestion principle	Throughput under loss	Delay sensitivity	PDR in high mobility	Overall suitability
Reno/new reno	Loss-based (AIMD)	Low	High	Low	Poor
Cubic	Loss-based (cubic)	Low to moderate	High	Low	Poor to fair
Vegas	Delay-based	Moderate	Low (proactive)	Moderate	Fair (too passive)
Westwood	Loss-based (BWE)	High	Moderate	Moderate to high	Good
Cross-layer	Feedback-based	Very high	Low (adaptive)	High	Excellent (complex)

BWE: bandwidth estimation

and integrating with next-generation networking paradigms. This section explores these advanced topics, providing a forward-looking perspective on the challenges and opportunities shaping the next generation of vehicular transport protocols.

6.6.1 Security Concerns and Impact on Transport Protocol Performance

The open and broadcast nature of the wireless medium makes VANETs susceptible to a wide range of cyberattacks, which can directly cripple the performance of transport protocols like TCP. Malicious actors can exploit vulnerabilities at the network and transport layers to induce artificial performance degradation, effectively turning TCP's own mechanisms against it [37].

Key vulnerabilities and their impact on TCP include:

- **Denial-of-Service (DoS) and Congestion Attacks**: Malicious nodes can flood a channel with bogus packets, creating artificial congestion. Standard TCP variants, unable to distinguish this from legitimate network saturation, will aggressively reduce their congestion windows, leading to a drastic drop in throughput for all nearby legitimate connections [38].
- **Blackhole and Grayhole Attacks**: A compromised or malicious node participating in the routing process can selectively or completely drop packets passing through it. To a TCP sender, this appears as an extremely lossy or broken link, triggering repeated timeouts and retransmissions, ultimately leading to connection failure and severe throughput degradation [39].
- **Spoofing and Session Hijacking**: Attackers can spoof IP addresses to inject malicious packets, such as TCP RST (reset) segments, to prematurely terminate established connections. This disrupts application data flow and requires a new, time-consuming connection setup process.

Mitigating these threats requires a security-aware, cross-layer approach. While secure routing protocols are the first line of defense, recent research explores transport-layer enhancements and Intrusion Detection Systems (IDS). For example, cross-layer IDS can monitor MAC-layer behavior (e.g., unusually high packet drop rates from a specific neighbor) and correlate it with routing and transport layer performance to detect and isolate malicious nodes. Developing lightweight, authenticated TCP extensions that can validate control segments without imposing significant overhead is an active area of research.

6.6.2 Multipath TCP (MPTCP) and Alternatives

Single-path TCP is inherently fragile in VANETs, as a single link failure can terminate the entire connection. Multipath TCP (MPTCP) addresses this by allowing a single transport connection to operate over multiple paths simultaneously, making it a natural fit for the heterogeneous connectivity of modern vehicles [40].

In a vehicular context, a vehicle might simultaneously have access to a direct V2V link (via IEEE 802.11p or C-V2X Sidelink), a V2I link to a nearby RSU, and a cellular V2N link (via 5G). MPTCP can leverage these paths to provide:

- **Fault Tolerance and Resilience**: If a volatile V2V path breaks due to mobility, the MPTCP connection can seamlessly continue over the more stable cellular link, preventing a session interruption. This "make-before-break" handover capability is critical for application continuity.
- **Load Balancing and Throughput Aggregation**: MPTCP can distribute data across multiple paths, potentially aggregating their bandwidth to achieve higher throughput than any single path could provide.

However, MPTCP faces challenges in VANETs, including managing highly asymmetric paths (e.g., a low-latency V2V link vs. a high-latency cellular link), the overhead of subflow management in a rapidly changing environment, and the difficulty of accurate link quality estimation for its packet schedulers [41].

As alternatives, protocols like **QUIC** (Quick UDP Internet Connections) are gaining traction. Built on UDP, QUIC eliminates head-of-line blocking and features a connection migration mechanism based on a connection ID, which allows it to maintain a session even as the client's IP address changes—a perfect feature for mobile vehicles switching between Wi-Fi and cellular networks [42]. **SCTP** (Stream Control Transmission Protocol) also offers multi-homing and multi-streaming capabilities, providing similar resilience benefits to MPTCP.

6.6.3 *Integration with Emerging Paradigms*

The evolution of TCP in VANETs is increasingly intertwined with broader shifts in network architecture.

- **Named Data Networking (NDN)**: As an Information-Centric Networking (ICN) architecture, NDN shifts the paradigm from host-centric (IP address) to data-centric (content name) communication. In VANETs, this is highly advantageous. A vehicle requests content by name (e.g., "/intersection/main-street/traffic.mp4"), and any nearby node (vehicle or RSU) holding the data can respond. This eliminates the need for a stable end-to-end TCP connection to a specific server, leveraging in-network caching and making content retrieval inherently resilient to mobility and intermittent connectivity.
- **Edge and Fog Computing**: Deploying computational resources at the network edge (e.g., in RSUs or 5G base stations) can significantly enhance TCP performance. Edge nodes can act as intelligent proxies, terminating the TCP connection from the core network and managing the volatile wireless link to the vehicle using a specialized, mobility-aware transport protocol. This split-TCP approach isolates the wired and wireless segments, improving latency and reliability. Furthermore, the edge node's localized view of network conditions allows it to provide more accurate and timely congestion feedback to vehicles [43].
- **AI-Driven TCP Optimization**: Machine Learning (ML) and Artificial Intelligence (AI) are poised to revolutionize transport protocol design. An AI agent on a vehicle could use inputs like GPS data, speed, and surrounding traffic density to predict future link quality or impending disconnections. This predictive capability would allow TCP to proactively adjust its congestion window or prepare for a path handover before packet loss occurs. Federated learning approaches can enable vehicles to collaboratively train a shared prediction model without compromising their private data, leading to a more robust and intelligent network-wide congestion control system.

The future of reliable transport in VANETs lies beyond incremental improvements to legacy TCP. A holistic approach that integrates robust security, leverages multi-path capabilities, and capitalizes on architectural innovations like edge computing and AI will be essential to meeting the stringent performance demands of next-generation intelligent transportation systems.

6.7 Open Research Challenges

Despite significant advancements in adapting transport protocols for vehicular environments, the journey towards achieving seamless, reliable, and efficient data communication is far from complete. The unique and extreme dynamics of VANETs continue to pose fundamental challenges that current solutions only partially address.

This section synthesizes the most pressing open research challenges and outlines promising directions for future investigation, bridging the gap between current capabilities and the stringent requirements of future connected and autonomous vehicle applications.

6.7.1 Scalability, Reliability, and Adaptability in Highly Dynamic Environments

The core unresolved challenge lies in designing a transport protocol that can concurrently manage scalability, reliability, and adaptability across the full spectrum of vehicular network conditions.

- **Scalability**: TCP's performance does not scale linearly with network density. In sparse networks, the challenge is maintaining connectivity, leading to frequent timeouts and connection failures [44]. Conversely, in dense urban traffic, the network becomes contention-limited. The sheer volume of control packets, including TCP acknowledgments and routing updates, can lead to MAC layer saturation and broadcast storms, creating congestion that throttles legitimate data flows. A scalable transport protocol must be able to gracefully handle both network fragmentation and high-density contention without collapsing [45].
- **Reliability**: Ensuring end-to-end reliability remains an elusive goal. The traditional TCP model of retransmitting lost packets is inefficient in the face of frequent, mobility-induced link disconnections. A lost ACK or data segment might be due to a transient route failure, not permanent loss. A key research gap is the development of a "disconnection-tolerant" TCP that can intelligently pause and resume a connection, distinguishing between temporary unavailability and catastrophic link failure [7].
- **Adaptability**: A one-size-fits-all congestion control algorithm is demonstrably ineffective. The optimal strategy for a high-speed highway scenario (dominated by mobility-induced loss) is different from that of a congested urban intersection (dominated by contention-induced loss). The central challenge is to create a truly adaptive TCP that can dynamically sense its environment—using metrics like vehicle speed, neighbor density, RSU proximity, and underlying routing behavior—and adjust its congestion control, retransmission, and flow management policies in real time [46].

6.7.2 Handover and Connectivity Issues

Seamless connectivity is a prerequisite for many emerging vehicular applications, yet it is profoundly difficult to maintain with standard TCP.

- **Handover Management**: Maintaining a TCP session during handovers—whether horizontal (between RSUs) or vertical (between different network interfaces like DSRC, C-V2X, and Wi-Fi)—is a critical challenge. Handover latency, even if brief, often exceeds TCP's RTO, causing spurious timeouts and unnecessary 'cwnd' reductions. While solutions like Multipath TCP (MPTCP) offer a theoretical framework for "make-before-break" handovers, optimizing their schedulers and subflow management for the rapid and asymmetric path changes in VANETs remains an open problem [47].
- **Impact of Route Re-establishment**: In multi-hop V2V networks, TCP's performance is intrinsically tied to the underlying routing protocol. The delay incurred by a reactive routing protocol (e.g., AODV) to discover a new path after a link break is a major source of TCP performance degradation. Research is needed on cross-layer mechanisms that can better synchronize the state of the transport and network layers, allowing TCP to patiently wait for a new route without prematurely terminating the connection [48].

6.7.3 Prospects for Future TCP Designs in Vehicular Networks

Addressing the aforementioned challenges requires moving beyond incremental improvements and embracing novel architectural and algorithmic paradigms.

- **AI/ML-Driven Transport Protocols**: Artificial Intelligence and Machine Learning represent the most promising frontier for creating adaptive protocols. An ML model could be trained to predict link instability or congestion based on real-time vehicular data (GPS, speed, surrounding vehicle density). This would enable a proactive TCP that could, for instance, reduce its sending rate before contention occurs or preemptively shift data to a more stable path before a link breaks [49, 50].
- **Edge-Native Transport Solutions**: The integration with Mobile Edge Computing (MEC) opens up new possibilities. Instead of a monolithic end-to-end TCP, a split-protocol architecture could be employed where a specialized, mobility-aware transport protocol manages the volatile Vehicle-to-Edge link, while standard TCP handles the stable connection from the edge to the core network. Designing and standardizing these edge-native protocols is a key research direction [51].
- **Standardized Benchmarking Frameworks**: A significant impediment to progress is the lack of standardized evaluation methodologies. Different studies use varying mobility models, network simulators, traffic densities, and performance metrics, making it difficult to compare the true efficacy of proposed solutions. The community urgently needs to develop a common set of benchmark scenarios (e.g., "Standard Urban Grid," "Standard Highway Stretch"), realistic vehicular datasets, and evaluation frameworks to ensure that future research is both reproducible and comparable [52].

The most critical research gaps revolve around creating transport protocols that are context-aware, predictive, and resilient to intermittent connectivity. Future work must focus on the tight integration of AI, edge computing, and multi-path capabilities, underpinned by a community-driven effort to standardize performance evaluation for a more rigorous and accelerated innovation cycle.

6.8 Conclusions

This survey has systematically demonstrated the fundamental incompatibility between TCP's congestion-centric design and the dynamic, error-prone nature of VANETs. The analysis of empirical studies consistently shows that traditional loss-based variants like TCP Reno and Cubic perform poorly, as they invariably misinterpret non-congestive packet losses due to mobility and wireless channel errors as signs of network congestion. In contrast, bandwidth-estimation variants such as TCP Westwood offer substantial performance gains and represent a more robust, readily deployable end-to-end solution. However, the most significant improvements are achieved through cross-layer adaptations that provide TCP with explicit feedback about link failures, and advanced paradigms like MPTCP that enhance connection resilience.

Building on these findings, several critical research avenues remain open and demand focused attention. First, the challenge of achieving seamless scalability and adaptability in environments that swing between extreme sparsity and high-density contention remains largely unsolved. Second, the design of security-aware transport protocols that can withstand attacks like DoS or blackhole without compromising performance is a nascent but vital field. Third, the lack of standardized benchmarking frameworks and realistic, large-scale vehicular datasets impedes reproducible and comparable research—a gap the community must address.

Future research should prioritize the integration of AI/ML for predictive congestion and link quality estimation, the development of hybrid transport protocols that use both cross-layer feedback and multi-path capabilities, and the design of edge-native transport solutions that capitalize on the low-latency infrastructure of 5G/6G networks.

References

1. Rodr, J., Wei, Z., Wang, J., Gutierrez, C.A., Correia, L.M., et al.: 6g-enabled vehicle-to-everything communications: current research trends and open challenges. IEEE Open J. Vehic. Technol. **21**, 653 (2025)
2. Al-Maliki, H., Al-Asadi, H.A.A., Abduljabbar, Z.A., Nyangaresi, V.O.: Reliable vehicular ad hoc networks for intelligent transportation systems based on the snake optimization algorithm. Eng. Technol. Appl. Sci. Res. **14**(6), 18631–18639 (2024)
3. Karne, R., Sreeja, T.: Review on vanet architecture and applications. Turk. J. Comput. Math. Educ. **12**(4), 1745–1749 (2021)

4. Mohamed, T.M., Ahmed, I.Z., Sadek, R.A.: Efficient vanet safety message delivery and authenticity with privacy preservation. PeerJ Comput. Sci. **7**, e519 (2021)
5. Allman, M., Paxson, V., Blanton, E.: RFC 5681—TCP Congestion Control, RFC 5681, IETF (2009). URL https://datatracker.ietf.org/doc/html/rfc5681
6. Achichi, B., Semchedine, F., Derdouri, L.: Hybrid approach for congestion control in vanets. In: Proceedings of the 7th International Conference on Software Engineering and New Technologies, pp. 1–4 (2018)
7. Kumar, K., Arora, S.K., Gaba, G.S.: Review and analysis of TCP performance in vanet. Wireless Person. Commun. **120**(1), 1027–1043 (2021). https://doi.org/10.1007/s11277-021-08448-z
8. Noor-A-Rahim, M., Liu, Z., Lee, H., Khyam, M.O., He, J., Pesch, D., Moessner, K., Saad, W., Poor, H.V.: 6g for vehicle-to-everything (v2x) communications: enabling technologies, challenges, and opportunities. Proc. IEEE **110**(6), 712–734 (2022)
9. Cunha, F., Villas, L., Boukerche, A., Maia, G., Viana, A., Mini, R.A., Loureiro, A.A.: Data communication in vanets: protocols, applications and challenges. Ad Hoc Netw. **44**, 90–103 (2016)
10. Wang, K., Guo, Y., Guo, Z.: Highly concurrent TCP session connection management system on fpga chip. Micromachines **14**(2), 385 (2023)
11. Al-Maaitah, N.O.: Tcp at-er: Tcp adaptive timeout based-on early retransmission over manet. J. Theor. Appl. Inform. Technol. **97**(24), 1154 (2019)
12. Abed, G.A., Ismail, M., Jumari, K.: A survey on performance of congestion control mechanisms for standard tcp versions. Aust. J. Basic Appl. Sci. **5**(12), 1345–1352 (2011)
13. TCP flow control and the sliding window. Technical report, IBM (1995). https://www.ibm.com/docs/en/storage-protect/8.1.25?topic=tuning-tcp-flow-control
14. Lorincz, J., Klarin, Z., Ozegoviá, J.: A comprehensive overview of tcp congestion control in 5g networks: research challenges and future perspectives. Sensors **21**(13), 4510 (2021)
15. Floyd, S., Henderson, T.: RFC 2582: The NewReno Modification to TCP's Fast Recovery Algorithm, RFC 2582, IETF (1999). https://www.rfc-editor.org/rfc/rfc2582.html
16. Brakmo, L.S., O'malley, S.W., Peterson, L.L.: Tcp vegas: new techniques for congestion detection and avoidance. In: Proceedings of the Conference on Communications Architectures, Protocols and Applications, pp. 24–35 (1994)
17. Ha, S., Rhee, I., Xu, L.: Cubic: a new tcp-friendly high-speed tcp variant. ACM SIGOPS Oper. Syst. Rev. **42**(5), 64–74 (2008)
18. Rao, A., Warrier, A., Raina, G.: Stability analysis of compound tcp with adaptive virtual queues. In: Proceedings of the 25th Chinese Control and Decision Conference (CCDC), pp. 2148–2154. IEEE (2013)
19. Xu, K., Tian, Y., Ansari, N.: Tcp-jersey for wireless IP communications. IEEE J. Sel. Areas Commun. **22**(4), 747–756 (2004)
20. Priyadarshini, P., Hota, L., Kumar, A., Chong, P.H.J.: A cross-layer congestion-aware routing protocol for uav-aided vanets with enhanced line-of-sight communication. In: International Conference on Smart Grid Inspired Future Technologies, pp. 131–144. Springer (2024)
21. De Castro, C., Raffaelli, C., Andrisano, O.: A dynamic hierarchical vanet architecture for named data networking applications. In: IEEE International Conference on Communications (ICC), pp. 3659–3665 (2015). https://doi.org/10.1109/ICC.2015.7248893
22. Yusuf, S.A., Khan, A., Souissi, R.: Vehicle-to-everything (v2x) in the autonomous vehicles domain: a technical review of communication, sensor, and AI technologies for road user safety. Transp. Res. Interdiscip. Perspect. **23**, 100980 (2024)
23. Wang, J., Topilin, I., Feofilova, A., Shao, M., Wang, Y.: Cooperative intelligent transport systems: the impact of c-v2x communication technologies on road safety and traffic efficiency. Sensors **25**(7), 2132 (2025)
24. Yin, H., Zhang, L., Roy, S.: Multiplexing urllc traffic within embb services in 5g nr: fair scheduling. IEEE Trans. Commun. **69**(2), 1080–1093 (2020)
25. Sharef, B.T., Alsaqour, R.A., Ismail, M.: Comparative study of variant position-based vanet routing protocols. Proced. Technol. **11**, 532–539 (2013)

26. Bejaoui, T.: QoS-oriented high dynamic resource allocation in vehicular communication networks. Sci. World J. **2014**(1), 718698 (2014)
27. Na, W., Lakew, D.S., Lee, J., Cho, S.: Congestion control versus link failure: TCP behavior in mmwave connected vehicular networks. Fut. Gener. Comput. Syst. **101**, 1213–1222 (2019)
28. Paxson, V., Allman, M., Chu, H.J., Sargent, M.: Rfc 6298—computing tcp's retransmission timer, RFC 6298, Internet Engineering Task Force (IETF) (2011). https://datatracker.ietf.org/doc/html/rfc6298
29. Blanton, E., Allman, M.: On making tcp more robust to packet reordering. ACM SIGCOMM Comput. Commun. Rev. **32**(1), 20–30 (2002)
30. Kulkarni, P., Sooriyabandara, M., Li, L.: Improving tcp performance in wireless networks by classifying causes of packet losses. IEEE Wireless Commun. Netw. **2009**, 1–6 (2009). https://doi.org/10.1109/WCNC.2009.4917752
31. Dalal, K., Chaudhary, P., Dahiya, P.: Performance evaluation of tcp and udp protocols in vanet scenarios using nctuns-6.0 simulation tool. Int. J. Comput. Appl. **36**(6), 6–9 (2011)
32. Zolfagharipour, L., Kadhim, M.H.: A technique for efficiently controlling centralized data congestion in vehicular ad hoc networks. Int. J. Comput. Netw. Appl. **12**(2), 267–277 (2025)
33. Biswal, S.P., Gupta, S., Kumar, A., Patel, S.: Comparative analysis of compound tcp with various end-to-end congestion control mechanisms. In: Proceedings of the 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), pp. 1762–1767. IEEE (2022)
34. Jamal, H., Sultan, K.: Performance analysis of tcp congestion control algorithms. Int. J. Comput. Commun. **2**(1), 30–38 (2008)
35. Gerla, M., Ng, B.K., Sanadidi, M., Valla, M., Wang, R.: Tcp westwood with adaptive bandwidth estimation to improve efficiency/friendliness tradeoffs. Comput. Commun. **27**(1), 41–58 (2004)
36. Prajapati, H.B., Bhatt, B.S., Dabhi, V.K.: An elfn-based adaptive probing strategy to improve tcp performance in ad hoc networks. Int. Confer. Comput. Commun. Eng. **2008**, 570–573 (2008). https://doi.org/10.1109/ICCCE.2008.4580668
37. Feng, X., Li, Q., Sun, K., Xu, K., Wu, J.: Exploiting cross-layer vulnerabilities: off-path attacks on the TCP/IP protocol suite. Commun. ACM **68**(3), 48–59 (2025)
38. Alenezi, M., Reed, M.J.: Denial of service detection through tcp congestion window analysis. World Cong. Internet Sec. **2013**, 145–150 (2013). https://doi.org/10.1109/WorldCIS.2013.6751036
39. Gurung, S., Chauhan, S.: Performance analysis of black-hole attack mitigation protocols under gray-hole attacks in manet. Wireless Netw. **25**(3), 975–988 (2019)
40. Liu, C., Qiu, Y., Chen, M., Cai, Y.: A vehicular network multipath management algorithm based on vehicular traffic. Vehic. Commun. **53**, 100906 (2025)
41. Zhang, T., Zhao, S., Cheng, B.: Multipath routing and mptcp-based data delivery over manets. IEEE Access **8**, 32652–32673 (2020)
42. Amponis, G., Lagkas, T., Argyriou, V., Radoglou-Grammatikis, P., Kyranou, K., Makris, I., Sarigiannidis, P.: Channel-aware quic control for enhanced cam communications in c-v2x deployments over aerial base stations. IEEE Trans. Veh. Technol. **73**(7), 9320–9333 (2024). https://doi.org/10.1109/TVT.2024.3393614
43. Peyman, M., Fluechter, T., Panadero, J., Serrat, C., Xhafa, F., Juan, A.A.: Optimization of vehicular networks in smart cities: from agile optimization to learn heuristics and simheuristics. Sensors **23**(1), 499 (2023)
44. Chen, A., Khorashadi, B., Ghosal, D., Chuah, C.-N.: Impact of transmission power on tcp performance in vehicular ad hoc networks. In: Proceedings of the 2007 Fourth Annual Conference on Wireless on Demand Network Systems and Services, pp. 65–71. IEEE (2007)
45. Chao, L., Wu, C., Yoshinaga, T., Bao, W., Ji, Y.: A brief review of multipath tcp for vehicular networks. Sensors **21**(8), 2793 (2021)
46. Mishra, T.K., Sahoo, K.S., Bilal, M., Shah, S.C., Mishra, M.K.: Adaptive congestion control mechanism to enhance tcp performance in cooperative IoV. IEEE Access **11**, 9000–9013 (2023)
47. Hapanchak, V., Costa, A.: A cross-layer approach for mptcp path management in heterogeneous vehicular networks. J. Commun. Softw. Syst. **19**(1), 103–113 (2023)

48. Jarupan, B., Ekici, E.: A survey of cross-layer design for vanets. Ad Hoc Netw. **9**(5), 966–983 (2011)
49. Jiang, H., Li, Q., Jiang, Y., Shen, G., Sinnott, R., Tian, C., Xu, M.: When machine learning meets congestion control: a survey and comparison. Comput. Netw. **192**, 108033 (2021)
50. Bhatia, J., Dave, R., Bhayani, H., Tanwar, S., Nayyar, A.: Sdn-based real-time urban traffic analysis in VANET environment. Comput. Commun. **149**, 162–175 (2020)
51. Sayegh, R., Marouane, H., Hoteit, S., Dandoush, A.: Mobility aware task migration in vehicular edge computing networks. In: Proceedings of the 36th International Tele traffic Congress "Towards Sustainable Digital Infrastructures" (ITC 36) (2025)
52. Petrov, T., Sevcik, L., Pocta, P., Dado, M.: A performance benchmark for dedicated short-range communications and lte-based cellular-v2x in the context of vehicle-to-infrastructure communication and urban scenarios. Sensors **21**(15), 5095 (2021)

Chapter 7
Towards Intelligent and Secure Vehicular Ad Hoc Networks: A Systematic Review of Post-2020 Research



Kaushik Adhikary, Bidisha Bhabani, and Tanaya Das

Abstract Vehicular Ad Hoc Networks (VANETs) is evolving as a core technology helping the Intelligent Transportation System (ITS) by providing a communication channel between vehicles and roadside infrastructure. VANETs are developing towards intelligent, safe, and autonomous vehicular ecosystems by combining Artificial Intelligence (AI), edge computing and blockchain. This chapter gives an insight of the recent papers published between 2020 and 2025. This survey focusses on advancements in routing, security, intrusion detection, and communication technologies like DSRC, C-V2X and emerging 6G frameworks. The research analysis focusses on significant progress in adaptive and Software Defined Network (SDN) based routing, lightweight authentication and Machine Learning based intrusion detection mechanisms. Still, there are many challenges that are yet to be resolved in achieving large-scale real-world communication, maintaining privacy, security, and ensuring interoperability across different networks. The chapter further discusses current research trends, identifies gaps such as congestion control and cross layer optimization, and also proposes potential direction towards collated learning, decentralized security models, and hybrid cloud architectures. The study overall stresses the need for VANETs to be scalable, explainable, and privacy preserving frameworks that can adapt to the dynamic nature of vehicular environments and support the vision of a safe, efficient and fully connected transportation systems.

Keywords Vehicular ad hoc networks (VANETs) · Intelligent transportation systems (ITS) · Artificial intelligence (AI) · Edge computing · Blockchain security · Software-defined networking (SDN)

K. Adhikary · B. Bhabani (✉) · T. Das
Department of Computer Science and Engineering, JIS University, Agarpara, Kolkata, India
e-mail: bidisha.bhabani@jisuniversity.ac.in

K. Adhikary
e-mail: kaushik.adhikary@jisuniversity.ac.in

T. Das
e-mail: tanaya.das@jisuniversity.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_7

7.1 Introduction

In recent times, with the quick growth of wireless communication technologies and Intelligent Transportation Systems (ITS), Vehicular Adhoc Networks (VANETs) is being developed. VANETs, is a specialized version of Mobile Adhoc Networks (MANETs) [1, 2]. VANETs helps in smooth communication among Vehicles (V2V) and between vehicles and roadside infrastructure (V2I). This network model helps in exchanging real time data for applications like accident avoidance, traffic jams, navigation, and infotainments. As the auto sector moves towards independent and connected vehicles, VANETs plays a crucial role in making the future of smart transportation systems [3, 4].

The infrastructure of VANET consists of vehicles which have On-board Units (OBUs) and fixed nodes called as Roadside Units (RSUs). The vehicles exchange information using wireless technologies such as Dedicated Short-Range Communication (DSRC), IEEE 802.11p or any emerging 5G/6G standards. The VANETs have unique characteristics like high mobility, rapidly changing topology and changing communication densities which are dependent on traffic conditions. The main challenges that a VANET face are having a steady communication, reducing network congestion, maintaining data security, and to get a low latent information distribution [5, 6].

The main objectives of VANETs are to increase road safety and transportation efficiency. One of the main safety applications of VANET is to notify the drivers about the traffic jams, accidents, dangerous road conditions on the way, and thereby reducing and taking preventive measures against accidents. Also, adding the comforting and commercial applications include route optimization, internet access, multimedia applications, and electronic toll collections. Due to the merging of Artificial Intelligence (AI) and edge computing, VANETs enable an intelligent communication infrastructure which can take real-time decisions based on data analytics and predictive modelling.

The architecture of VANET consists of vehicle-to-vehicle communication, vehicle to roadside communication and roadside to roadside communication [7, 8]. The overall coordination of all these nodes helps in creating a decentralized and centralized structures. The decentralized networks depend on peer-peer information exchange among vehicles, while the centralized systems use RSUs or cloud servers for data collections and coordination. This helps in creating a flexible architecture that allows VANETs to adjust to different circumstances from urban traffic to highway environments.

VANETs have tremendous potential to be the next generation automative technology. But still, they face many technical and practical challenges [9, 10]. As the vehicles have random speed, they lead to regular disruptions and fluctuating link quality which makes the routing and data delivery very complex. The wireless nature of VANETs also makes it vulnerable to security and privacy issues as any malicious nodes can insert fake information or can listen to sensitive data. Upscaling and interoperability among heterogeneous vehicles and communication standards also

remain key research areas [11]. To tackle these issues, it is required to develop a healthy routing protocol, adequate defense mechanism and adaptive communication frameworks designed to dynamic vehicular environments.

In short, VANETs have the capability to transform and make a leap in vehicular communication and ITS. By combining the features of real-time, secure and reliable exchange of information, they have the capability to change the nature of vehicles communicating with each other and with the infrastructure. With the progress in the world of vehicle communications and having the goal of an autonomous and connected mobility, upgrading VANET technology can help to play a crucial role in achieving a safe, efficient and smarter transportation infrastructure.

7.2 VANET Architecture

The ongoing need of people for safety, mobility, security, and multimedia applications on-board has led to the growth of vehicles from some plain mechanical locomotives to ITS. As a result, there is a greater need for VANETs in infrastructure and vehicle communications [12]. It is anticipated that VANET, a kind of MANET, will be the best technology for ITS. These networks are often created for a brief period of time by cars with wireless communication capabilities. These cars are equipped with OBU. These units are electronic gadgets that, as illustrated in Fig. 7.1, can communicate with RSUs (V2I communication) and other vehicles (V2V communication) that have OBUs via short-range direct communications. The communications that can occur between vehicles are known as V2V communications. The V2I communications are the communications that can take place between Vehicles and the RSUs. The RSUs play an important role of infrastructure-based communication. The real-time situation of the condition of a road is collected from vehicles, road side CCTVs, sensors, and RSUs. OBU analyzes all of the data gathered from different sources before showing the users the finished product. A variety of real-time applications that meet distinct demands might be developed as a result of the diverse kinds of information produced by diverse sources. So, new sorts of ITS related initiatives are continuously popping up and are being encouraged to be done by various ITS stakeholders such as governments, organizations, and vehicle manufacturers. The Indian government has acknowledged the necessity of developing creative transportation networks as a result of the country's digital revolution [13].

Contemporary vehicles are fitted with numerous amenities like remote locking, high-speed alerts, rear parking sensors, airbags, etc. However, these functionalities are offered without considering their security. Ensuring human safety is the primary goal of VANET development. These networks are only variations of MANETs, with common characteristics like infotainments, predictable vehicle motions, and limitless battery life. The applications which give safety-related information should be secured from hostile vehicles as any compromise in the safety and security of these applications can lead to the lethal effect of living beings who are communicating on the track. Lack of security protections in VANETs can result in a variety of attacks,

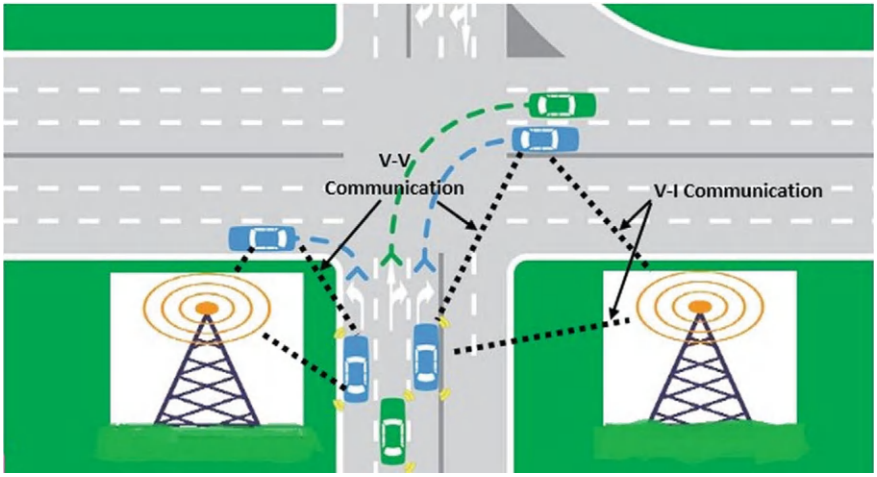


Fig. 7.1 Communications carried out in a VANET setup [14]

including denial-of-service (DoS) attacks, message suppression, and the dissemination of false alarms, all of which can ultimately cause accidents. The DoS attack is known as a Distributed DoS (DDoS) attack when it is executed by several vehicles dispersed over the network.

7.3 Issues and Challenges in Vehicular Ad Hoc Networks (VANETs)

7.3.1 High Mobility and Dynamic Topology

VANETs operate in highly mobile environments where vehicles frequently change position and velocity. This leads to rapid topology changes and frequent link disconnections, making routing and communication maintenance difficult [15]. Designing routing protocols capable of adapting to such dynamic conditions remains a fundamental research challenge.

7.3.2 Routing and Data Dissemination

Due to varying traffic densities and mobility patterns, efficient routing and data dissemination in VANETs are complex tasks. Traditional MANET protocols often fail in vehicular environments because of frequent topology changes and intermittent

connectivity [16]. Adaptive and location-based routing protocols have been proposed, yet balancing delay, reliability, and overhead continues to be a major issue.

7.3.3 Security and Privacy

Security is one of the most critical challenges in VANETs. Threats such as message falsification, Sybil attacks, denial-of-service (DoS), and eavesdropping can disrupt network reliability [17]. Furthermore, ensuring data authenticity while preserving user privacy is difficult, as driver information and location details are sensitive. Privacy-preserving authentication schemes are thus necessary to maintain trust without compromising confidentiality.

7.3.4 Scalability

As the vehicles add to the network, the communication channels cannot handle a huge flow of data which results in packet loss and increase in latency [18]. So, efficient bandwidth management and congestion control mechanisms are essential for maintaining service quality.

7.3.5 Quality of Service (QoS)

Critical applications related to safety in VANETs demand low latency and high reliability. However, maintaining QoS under varying conditions such as network congestion, signal interference, and high-speed mobility is extremely challenging [19]. Ensuring prioritized message delivery and fair resource allocation remains an open area of research.

7.3.6 Interoperability and Standardization

Due to different manufacturers using various communication technologies and standards (e.g., IEEE 802.11p, LTE-V2X, 5G NR-V2X), it leads to different interoperability problems. The absence of centralized global standards prevents smooth vehicular communication [20]. Standardization is therefore a critical factor for ensuring compatibility and reliable message exchange.

7.3.7 Infrastructure Dependency

VANETs are designed to be operatable in an ad hoc manner but many advanced applications depend on RSUs, cloud servers, or edge computing nodes. The less deployment of such infrastructure, especially in rural or underdeveloped areas restricts the performance of VANET systems [21].

7.3.8 Privacy–Accountability Trade-Off

Maintaining user anonymity while ensuring accountability for malicious actions is a complex ethical and technical problem. Complete anonymity can encourage misbehavior, whereas full traceability compromises privacy. Hybrid identity management systems and blockchain-based trust models are emerging solutions [22].

7.3.9 Energy and Resource Constraints

Vehicles have high energy compared to mobile devices. But computationally intensive security and routing algorithms increase the power and bandwidth consumptions. So, an efficient use of onboard resources and optimization of computational overhead are essential for practical deployment [23].

7.3.10 Legal and Ethical Concerns

There are many legal and ethical aspects to be addressed such as responsibility in case of system failures, and misuse of location information present many non-technical challenges. So, developing regulatory models to protect users while promoting innovations is required for public trust and adoption [24].

7.4 Literature Survey

The development of VANETs has been mainly influenced by rapid evolution of wireless communication, AI, and edge computing after 2020. The current research are being directed towards network performance, routing, security, scalability and integration with emerging technologies like blockchain, 5G/6G and SDN. This section reviews various research analyzing the key techniques, strengths, and limitations that have shaped recent VANET developments.

Dutta et al. [3] gave a detailed review of modern VANET developments comprising of traffic management, safety and remote monitoring applications. Their survey summarized the clustering and energy aware routing approaches tailored for V2X systems, emphasizing the necessity of integrating ML and edge computing for smart transportation. The authors pointed out the absence of large-scale real-world validation and cross layer optimization frameworks.

Setia et al. [25] proposed a ML-based DDoS detection technique for cloud computing in VANET. They achieved high accuracy compared to the traditional threshold-based models. They observed the need for better generalization across different dataset.

Hussein et al. [11] explored SDN-based VANET routing architectures. They focussed on global control and dynamic reconfiguration. Their paper stressed the need for real time traffic awareness and scalability though the challenges in controller placement and fault tolerance still existed.

Lee and Atkison [4] classified VANET applications across safety, infotainment and autonomous driving domains. They concluded that effective application level QoS depends on corresponding network layer optimization.

Grover [26] studied blockchain mechanisms for authentication and trust management. They observed data integrity improvement also noted the performance of trade-offs due to blockchain delay.

Mundhe et al. [27] gave an extensive survey on authentication and privacy preserving mechanisms comparing various cryptographic schemes and identifying the necessity for hybrid approaches with privacy guarantees.

Setitra and Fan [2] proposed an optimized TabNet based DDoS detection framework inside SDN enabled VANETs. They achieved robust classification but requiring validation using real vehicle data [2].

Chen et al. [28] designed a hybrid intrusion detection system (VAN-IDS) combining packet level and physical context features, yielding low detection latency but have less robustness testing.

Rabah and Idoudi [29] also designed an ML-based intrusion detection system which is validated through simulations; however, model performance degraded under varying mobility patterns. This revealed drift issues [29].

Hassan et al. [30] emphasized on improving transparency and trust in ML that detect cyber-attacks in connected vehicle environment. They used Random Forest to model intrusion detection at application layer using a large network dataset. The model achieved 99.84% accuracy.

Research in routing and data dissemination area has significantly improved. Wang et al. [21] using a two-dimensional Markov analytical model for IEEE 802.11 p/bd, studied medium access parameters suggesting further validation using real mobility data.

Pitchay et al. [31] proposed a privacy preserving low cost V2I authentication scheme. This ensured route confidentiality but lacked cross domain revocation solutions [31].

Shankar et al. [32] studied the edge AI and task offloading frameworks that are applicable to VANETs. They also underlined orchestration and model compression challenges in vehicular deployments [32].

Le et al. [33] merged blockchain assisted validation with edge computing, achieving improved security and low latency verification. Still, consensus capability issues persisted [33].

Zadobrischi and Havriliuc [34] explored the evolution from DSRC to NR-V2X revealing performance advantages of NR side links. They stressed the need for coexistence studies.

Borah et al. [35] studied distributed security frameworks such as federated learning and distributed ledgers. They recognized communication overhead and poisoning resistance as persistent challenges.

Some Deep learning-based DDoS detection methods compared CNN, RNN and ensemble models. These suggested lightweight ensembles for onboard feasibility. Ahsan et al. [36] suggested on expanding IDS frameworks using SHAP based feature ranking to improve transparency in decision making. Khedkar and Mahajan [37] suggested to use lightweight blockchain authentication for decentralized trust verification. They achieved tamper resistance but faced throughput constraints [37]. Mahi et al. [38] studied the cross-layer integration of edge, AI, and SDN technologies. They urged for standardized simulator benchmarking [38]. Li et al. [39] proposed "AuthUp" which is a formal privacy preserving V2I authentication scheme with provable security. Mendes et al. [40] explored V2X in 5G network. They had done a comprehensive survey on location-based services in hybrid scenarios [40]. Alrehan and Alhaidari [41] detects DDoS attack using Machine Learning Techniques on VANET System. Shah et al. [42] explored LTE-V2X/C-V2X scalability. They identified sidelink scheduling and mixed mode coexistence as critical research gaps [42]. Lang et al. [43] gave the overview of the 6G vehicular network vision, highlighted integrating sensing and communication but emphasized the lack of real-world prototypes. Table 7.1 shows the review of recent research carried out in VANETs.

Figure 7.2 shows that Intrusion Detection dominates the overall research focus, highlighting the growing concern over cybersecurity threats in vehicular networks. Authentication and Blockchain also hold strong proportions, indicating a shift toward trust management and secure message communication. In comparison, Routing has the lowest representation, suggesting it is no longer a major research priority relative to emerging security challenges. Edge Computing appears moderately, reflecting rising interest in distributed real-time processing capabilities. Overall, Fig. 7.2 demonstrates that modern VANET research is increasingly shifting toward security-centric and AI-driven solutions.

Figure 7.3 reveals that ML-based Intrusion Detection Systems occupy the largest share, confirming that machine learning has become the preferred security approach in VANET environments. Authentication schemes also hold substantial weight, underscoring the ongoing need for privacy protection and reliable identity verification. Blockchain contributes a significant portion, illustrating the growing use of decentralized trust frameworks. Meanwhile, Cryptography appears minimal,

Table 7.1 Review of recent research carried out in VANETs

Sl. No	Paper (year)	Technique used	Strength	Weakness
1	Dutta et al. [3]	Review of network technologies, clustering and energy-aware routing in V2X	Broad coverage of VANET + V2X trends	Lacks large-scale real-world validation
2	Setia et al. [25]	ML-based DDoS detection pipeline in VANET/ vehicular-cloud	Improved detection accuracy vs threshold methods	Dataset realism and generalization remain weak
3	Lee and Atkison [4]	Application-level survey (safety, infotainment, traffic management)	Good mapping of applications → network requirements	Application-layer to network-layer integration lacks depth
4	Grover [26]	Blockchain for VANET security (survey)	Highlights authentication/trust via blockchain	Latency/ throughput trade-offs not deeply resolved
5	Mundhe et al. [27]	Authentication and privacy mechanism survey	Compares anonymity versus accountability	Few formal privacy proofs, practical scheme evaluation limited
6	Setitra et al. [2]	Deep model (TabNet) for DDoS in SDN-VANET	Good detection performance, modern model	Real-world trace testing lacking
7	Chen et al. [28]	Hybrid packet + physics-based IDS for VANET	Low latency, suitable onboard	Adversarial robustness not fully addressed
8	Rabah et al. [29]	ML pipeline (feature selection + classifier) for VANET IDS	Practical ML workflow validated via simulation	Feature drift across traffic scenarios remains
9	Hassan et al. [30]	Explainability for VANET IDS (feature importance etc.)	Helps trust and tuning of models	Real-time interpretable model latency still high
10	Wang et al. [21]	Analytical 2-D Markov chain model for 802.11p/BD	Parameter-sensitivity study gives insight	Needs validation with real mobility traces
11	Ganesan et al. [31]	V2I fast authentication scheme (privacy-preserving)	Low computational cost and route privacy support	Cross-domain revocation handling needs more work
12	Shah et al. [42]	Cellular V2X (LTE-V2X/C-V2X) scalability study	Good city-scale considerations (scheduling, resource)	Sidelink scheduling and mixed-mode coexistence underexplored

(continued)

Table 7.1 (continued)

Sl. No	Paper (year)	Technique used	Strength	Weakness
13	Lang et al. [43]	6G vehicular network vision survey	Prepares for extreme low-latency, integrated sensing/comm	Mostly visionary; practical prototyping lacking
14	Shankar et al. [32]	Edge-AI in VANETs (resource-aware inference/ offloading)	Highlights orchestration and model compression	Deployment in vehicular setups needs more work
15	Le et al. [33]	Blockchain + edge node integration in VANET	Improved trust + low-latency verification	Blockchain scaling/ consensus overhead trade-offs high
16	Zadobrischi and Havriliuc [34]	DSRC → LTE-V2X → NR-V2X evolution review	Good performance-comparison across technologies	Co-existence/ stability in mixed networks still open
17	Borah et al. [35]	Distributed security (federated learning, distributed ledger) in VANET	Highlights decentralised defence possibilities	Communication cost and poisoning resilience need attention
18	Khedkar et al. [37]	Lightweight blockchain and authentication for VANET	Good for decentralised tamper-evidence	Vehicular latency and throughput trade-offs not fully resolved
19	Mahi et al. [38]	VANET research review: edge computing, AI, SDN	Highlights emergent tech integration	Standardised cross-simulation comparisons lacking
20	Li et al. [39]	V2I privacy-preserving authentication (AuthUp-like)	Formal proofs + revocation support	Practical vehicular deployment scenarios still few
21	Mendes et al. [40]	V2X performance (5G/NR perspectives)	Good mapping of latency/reliability + multi-radio suggestions	Multi-interface testing and real vehicles still limited

showing that traditional security mechanisms are being replaced by more adaptive and intelligent alternatives. Overall, Fig. 7.3 highlights a transition toward learning-based, flexible, and decentralized security systems in vehicular networks.

Figure 7.4 shows that only Random Forest, Ensemble Models, and TabNet appear within the references, indicating that VANET research relies primarily on structured, tabular machine learning methods. The absence of deep learning techniques

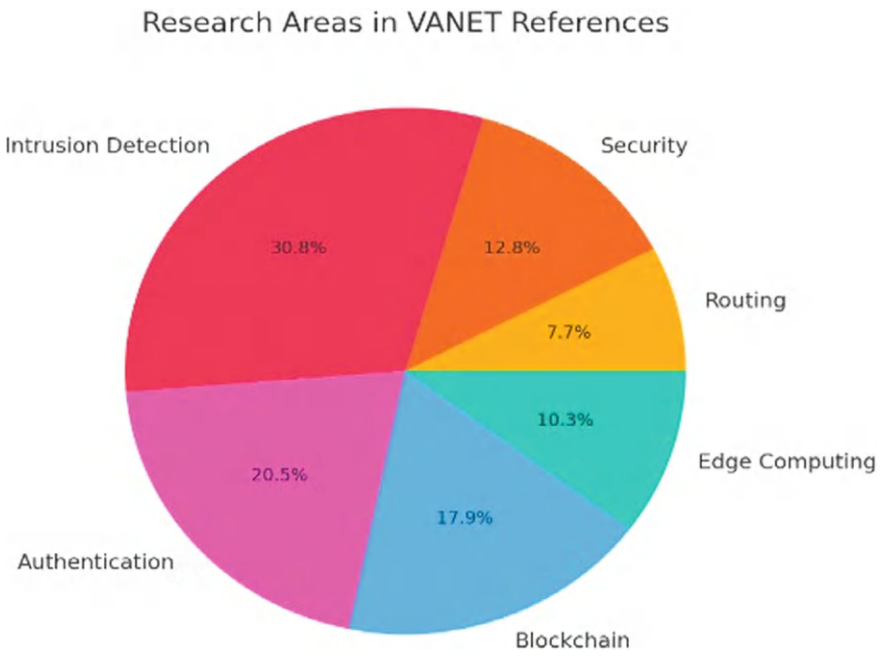


Fig. 7.2 Research areas in VANET references

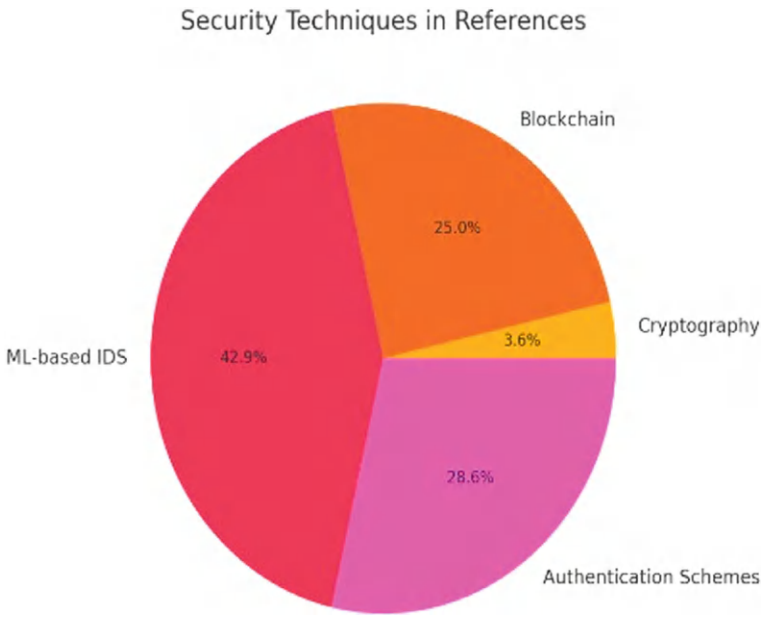


Fig. 7.3 Security techniques in references

such as CNNs and RNNs suggests that lightweight models are preferred for real-time vehicular applications. The presence of TabNet reflects recent adoption of modern attention-based models suitable for structured feature inputs. Ensemble methods appear as frequently as Random Forest, emphasizing the need for strong and stable classifiers in intrusion detection. Overall, Fig. 7.4 demonstrates that VANET researchers favor efficient, explainable, and computationally light ML approaches.

Figure 7.5 shows that the USA accounts for the majority of the publications, indicating its strong leadership in intelligent transportation and vehicular communication research. Germany and the Netherlands contribute equally, reflecting active engagement from European institutions. The absence of Asian contributions in this reference subset contrasts with global research trends, where Asian countries are increasingly active in ICT research. IEEE dominates the sources, which aligns with the USA's leading role in the dataset. Overall, Fig. 7.5 highlights that VANET research remains largely driven by Western academic and publication ecosystems.

Figure 7.6 shows that Intrusion Detection holds the highest priority score, underscoring its importance as the most critical area for future VANET research. Blockchain Security ranks second, showing growing interest in decentralized trust

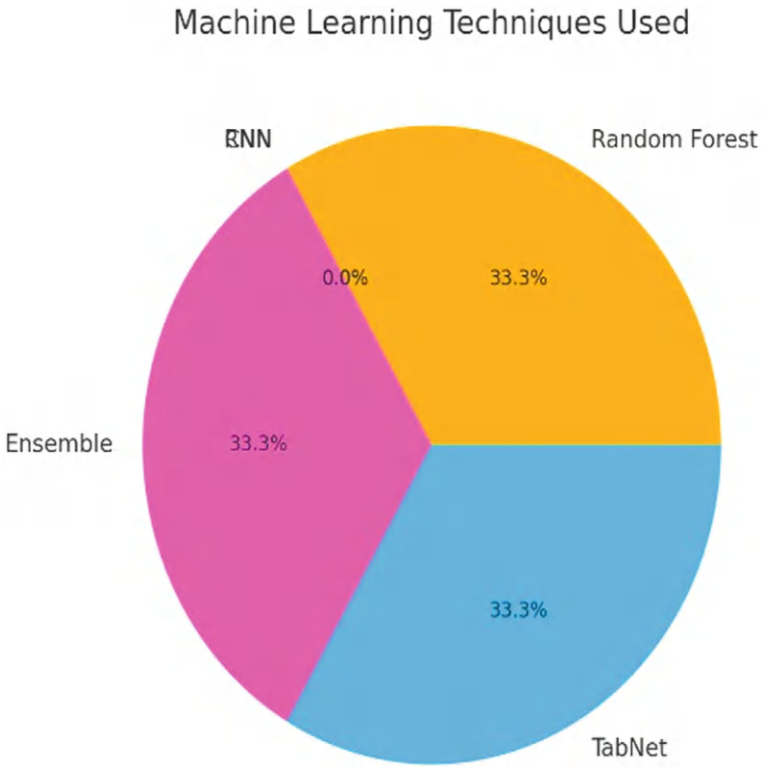


Fig. 7.4 Machine learning techniques discussed in this paper

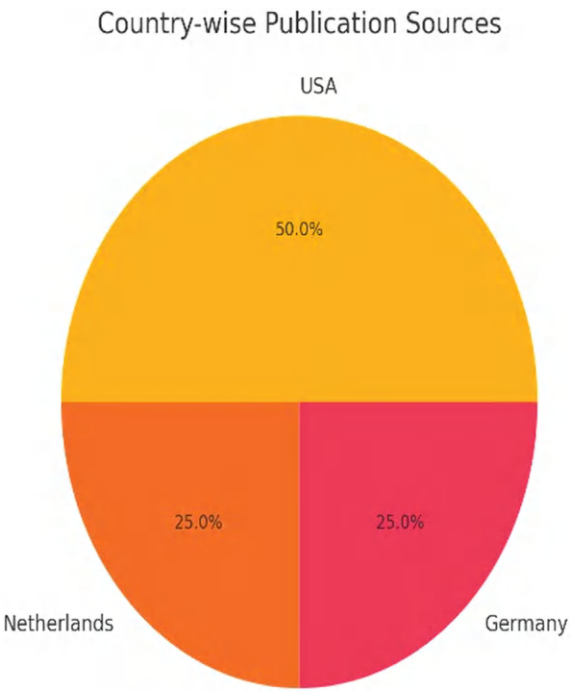


Fig. 7.5 Country wise publications

and tamper resistance. Authentication continues to be a vital focus, driven by the need for strong identity management in high-mobility vehicular networks. Edge Computing and 6G/SDN integration appear as emerging research areas, pointing toward next-generation low-latency and real-time processing challenges. Overall, Fig. 7.6 supports the development of lightweight, intelligent, and privacy-preserving intrusion detection systems tailored for 6G-enabled VANET architectures.

7.5 Open Research Challenges

Though there has been good progress in VANET research after 2020 using Machine Learning Techniques, SDN based routing, blockchain authentication, there are still many technical and problems which are yet to be tackled. These challenges make VANETs less feasible for real world applications, and large-scale implementation. This section discusses the deep analysis of recent challenges discovered by recent findings.

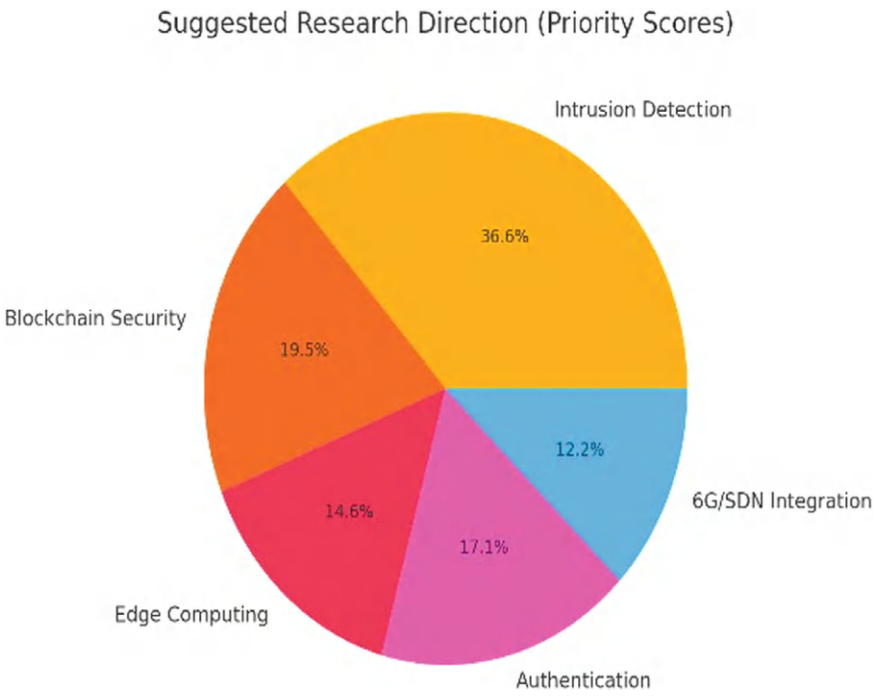


Fig. 7.6 Suggested research directions

7.5.1 Deployment, Scalability and Validation in Real World

The recent research during the last 5 years mostly carries out the simulations in SUMO, VEINS and NS3. Though these environments create flexibility, they cannot capture the sudden change in vehicular interactions, different mobile patterns, and heterogenous wireless environments.

Studies carried out by Setitra and Fan [2] and Adhikary et al. [8] have confirmed that most of the machine learning based models suffer from a lot of biasness, changing environments and real traffic scenarios. Hussein et al. [11] identified that SDN based models suffer from bottlenecks and lack validity in large scale deployments.

After analyzing the challenges, it has been felt to focus on the key research needs such as to use real vehicular traces for training IDS and routing models. Continuous learning models should be used for handling mobile data drifting.

7.5.2 Congestion Control Along with Scalability

As the V2V and V2I messaging increases, it creates severe channel congestion, affecting latency critical applications.

The latest technologies like NR-V2X, Wi-Fi 6 and mmWave improve bandwidth but requires smart coordination. Zadobrischi and Havriliuc [34] and Shah et al. [42] showed that by mixing DSRC with LTE-V2X, the results showed side link performance due to interference and non-harmonized scheduling. The key research needs are AI-driven congestion control that dynamically adjusts transmission rates. Cross-layer frameworks that combine MAC scheduling, network load prediction, and cooperative sensing. Regional congestion management is integrated with SDN-based controllers.

7.5.3 Trade-Off Between Security and Performance

Most of the security mechanisms like Blockchain, PKI based authentication, and cryptographic signing provide significant security and trust. But they add computational cost and network delays which cannot be used in collision warning and platooning. Recent block chain-based schemes proposed by Grover [26] and Le et al. [33] shows integrity improvements but suffer from scalability limitations due to consensus algorithms. Latency spikes during dense traffic. High resource consumption which are unsuitable for low cost OBUs. So, the key focus areas should be lightweight cryptography, scalable and fast blockchain consensus.

7.5.4 Preservation of Privacy

The balance between privacy and traceability remains an unresolved challenge. The systems based on Pseudonym improve anonymity but complicate revocation, especially across heterogeneous domains. Latest researches Mundhe et al. [27], Pitchay et al. [31] show progress in privacy-preserving authentication, but issues like cross-region identity handover, preventing Sybil attacks under pseudonym rotation, ensuring non-repudiation without leaking driver identity persist. The key research needs include hierarchical trust frameworks supporting pseudonym refreshment, blockchain-assisted revocation lists with minimal overhead and privacy-by-design ML models that avoid learning identifiable patterns.

7.5.5 Variety of Network Management

By integrating various technologies like DSRC,6G, NR-V2X, Wi-Fi 6 there is a requirement of cross layer coordination frameworks.

The integration of multiple technologies like DSRC, 5G NR-V2X, Wi-Fi 6, satellite links requires cross-layer coordination frameworks. Mahi et al. [38] and Mendes et al. [40] highlight the difficulties in creating unified, cross-layer architectures capable of seamless handover and cooperative scheduling.

The key research needs to include unified communication stacks that harmonize heterogeneous radio interfaces, AI-based link selection predicting future link stability, and multi-interface scheduling to minimize handover delay.

7.5.6 Dynamic Topology and Link Stability

Frequent topology changes because of varying vehicular speeds creates a fundamental routing challenge. The traditional MANET routing fails under extreme mobility. Recent routing studies shows that SDN-based routing offers flexibility but suffers from controller overload. Predictive routing using ML shows promising results but requires real-world datasets.

The key research needs to include hybrid predictive routing using GPS trajectory forecasting and ML link quality prediction. Cooperative routing using RSUs and edge servers. Protocols that are designed for ultra-dense and sparse highway conditions. There is an utmost challenge to maintain reliable links under high-speed mobility and varying mobility which demands new routing paradigms like predictive or AI-assisted routing.

7.5.7 Standardization and Interoperability

Global standardization remains fragmented across different regions like ETSI, IEEE, 3GPP, affecting cross-border vehicular movement and adoption by manufacturers. Policy challenges also include accountability in autonomous decision-making, Misuse of vehicular data, compliance with privacy regulations like GDPR, national data laws. The key research issues that need to be focussed are unified global V2X communication standards, collaborative testbeds involving governments, automakers, and telecom operators, ethical governance frameworks for data sharing, IDS transparency, and autonomous decision auditing.

7.6 Conclusion

VANETs have developed into a suitable ITS, which offers better safety, traffic management, and onboard infotainments. The merger of AI, SDN, blockchain and edge computing has changed VANET design, which has led to an intelligent, adaptive, and secure communication infrastructure.

Though despite all these advantages, there are still many challenges that remain in achieving global interoperability, large scale deployment, and privacy complaint architectures. The focus in future should be on developing lightweight AI models, reliable decentralized mechanisms, and cross layer optimization techniques capable of adjusting to real time vehicular movements. The collaboration of different technologies gives a high boost to autonomy, safety and resilience making the way for fully connected and ITS ecosystems.

References

1. Hozouri, A., Mirzaei, A., RazaghZadeh, S., Yousefi, D.: An overview of VANET vehicular networks. Netw. Internet Architect. **13**, 6555 (2023)
2. Setitra, M.A., Fan, M.: Detection of DDoS attacks in SDN-based VANET using optimized TabNet. Comput. Stand. Interf. **90**, 103845 (2024)
3. Dutta, A., Campoverde, L.M.S., Tropea, M., De Rango, F.: A comprehensive review of recent developments in VANET for traffic, safety and remote monitoring applications. J. Netw. Syst. Manag. **32**(4), 1–37 (2024)
4. Lee, M., Atkison, T.: VANET applications: past, present, and future. Vehic. Commun. **28**, 100310 (2021)
5. Karabulut, M.A., Shah, A.F.M.S., Ilhan, H., Pathan, A.S.K., Atiquzzaman, M.: Inspecting VANET with various critical aspects: a systematic review. Ad Hoc Netw. **150**, 103281 (2023)
6. Adhikary, K., Bhushan, S., Kumar, S., Dutta, K.: Evaluating the impact of DDoS attacks in vehicular ad-hoc networks. Int. J. Sec. Privacy Pervasive Comput. **12**(4), 1–18 (2020)
7. Adhikary, K., Bhushan, S., Kumar, S., Dutta, K.: Hybrid algorithm to detect DDoS attacks in VANETs. Wirel. Pers. Commun. **114**(4), 6985 (2020)
8. Adhikary, K., Bhushan, S., Kumar, S., Dutta, K.: Evaluating the performance of various SVM kernel functions based on basic features extracted from KDDCUP'99 dataset by random forest method for detecting DDoS attacks. Wirel. Pers. Commun. **123**(4), 6952 (2022)
9. Bibi, R., et al.: Edge AI-based automated detection and classification of road anomalies in VANET using deep learning. Comput. Intell. Neurosci. **2021**(1), 6262194 (2021)
10. Bangui, H., Ge, M., Buhnova, B.: A hybrid machine learning model for intrusion detection in VANET. Computing **104**(3), 503–531 (2022)
11. Hussein, N., et al.: SDN-based VANET routing: a comprehensive survey on architectures, protocols, analysis, and future challenges. IEEE Access **13**, 126801–126861 (2025)
12. Ghori, M.R., Zamli, K.Z., Quosthoni, N., Hisyam, M., Montaser, M.: Vehicular ad-hoc network (VANET): review. In: Proceedings of the 2018 IEEE International Conference on Innovative Research and Development (ICIRD), Bangkok, Thailand, pp. 1–6 (2018). https://doi.org/10.1109/ICIRD.2018.8376311
13. Building innovative transport networks: Business Standard Column
14. Adhikary, K., Bhushan, S.: Recent techniques used for preventing DOS attacks in VANETs. In: Proceeding of the IEEE International Conference on Computing, Communication and Automation, ICCCA 2017 (2017). https://doi.org/10.1109/CCAA.2017.8229864

15. Ahmed, S., Ramesh, N.V.K., Reddy, B.N.K.: An efficient routing mechanism for VANETs in SDN with hybrid meta-heuristic algorithm. Telecommun. Syst. **88**(1), 1–10 (2025). https://doi.org/10.1007/S11235-024-01234-2/METRICS
16. Al-Shareeda, M.A., Alazzawi, M.A., Anbar, M., Manickam, S., Al-Ani, A.K.: A comprehensive survey on vehicular ad hoc networks (VANETs). In: Proceedings of the 2021 International Conference on Advanced Computer Applications, ACA 2021, pp. 156–160 (2021). https://doi.org/10.1109/ACA52198.2021.9626779
17. Maxim, R., Hubaux, J.-P.: Securing vehicular ad hoc networks. J. Comput. Secur. **12**, 70618 (2007). https://doi.org/10.5555/1370616.1370618
18. Shahariar, R., Phillips, C.: "A trust management framework for vehicular ad hoc networks. Int. J. Sec. Privacy Trust Manag. **12**(1), 1–22 (2024). https://doi.org/10.5121/ijsptm.2023.12102
19. Kumar, R., Dave, M.: A review of various VANET data dissemination protocols. Int. J. u-e-Serv. **5**(3), 313 (2012)
20. Kenney, J.B.: Dedicated short-range communications (DSRC) standards in the United States. Proceed. IEEE **99**(7), 1162–1182 (2011). https://doi.org/10.1109/JPROC.2011.2132790
21. Wang, Q., Gao, D., Zhu, W.: Cloud-enabled software-defined vehicular networks: architecture, applications and challenges. J. Internet Technol. **20**(6), 1819–1828 (2019). https://doi.org/10.3966/160792642019102006012
22. Zhang, Y., Tong, F., Xu, Y., Tao, J., Cheng, G.: A privacy-preserving authentication scheme for VANETs based on consortium blockchain. In: Proceedings of the 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall), Victoria, BC, Canada, pp. 1–6 (2020). https://doi.org/10.1109/VTC2020-Fall49728.2020.9348497
23. Saleem, M.A., et al.: Expansion of cluster head stability using fuzzy in cognitive radio CR-VANET. https://doi.org/10.1109/ACCESS.2019.2956478.
24. Dogan, E., Costantini, F., Boennec, R.L.: Ethical issues concerning automated vehicles and their implications for transport. Adv. Transp. Policy Plann. **5**, 215–233 (2020). https://doi.org/10.1016/BS.ATPP.2020.05.003
25. Setia, H., et al.: Securing the road ahead: machine learning-driven DDoS attack detection in VANET cloud environments. Cyber Sec. Appl. **2**, 100037 (2024). https://doi.org/10.1016/J.CSA.2024.100037
26. Grover, J.: Security of vehicular ad hoc networks using blockchain: a comprehensive review. Vehic. Commun. **34**, 100458 (2022). https://doi.org/10.1016/J.VEHCOM.2022.100458
27. Mundhe, P., Verma, S., Venkatesan, S.: A comprehensive survey on authentication and privacy-preserving schemes in VANETs. Comput. Sci. Rev. **41**, 100411 (2021). https://doi.org/10.1016/j.cosrev.2021.100411
28. Chen, X., Qiu, W., Chen, L., Ma, Y., Ma, J.: Fast and practical intrusion detection system based on federated learning for VANET. Comput. Secur. **142**, 103881 (2024). https://doi.org/10.1016/J.COSE.2024.103881
29. Rabah, N., Idoudi, H.: A machine learning framework for intrusion detection in VANET communications. Emerg. Trends Cybersec. Appl. **12**, 209–227 (2022). https://doi.org/10.1007/978-3-031-09640-2_10
30. Hassan, F., Yu, J., Syed, Z.S., Ahmed, N., Al-Reshan, M.S., Shaikh, A.: Achieving model explainability for intrusion detection in VANETs with LIME. PeerJ Comput. Sci. **9**, 1440 (2023). https://doi.org/10.7717/PEERJ-CS.1440
31. Pitchay, A.A., Ganesan, Y., Zulkifli, N.S., Khaliq, A.: Determinants of customers' intention to use online food delivery application through smartphone in Malaysia. Br. Food J. **124**(3), 732–753 (2022)
32. Shankar, A., Jebarajakirthy, C., Nayal, P., Maseeh, H.I., Kumar, A., Sivapalan, A.: Online food delivery: a systematic synthesis of literature and a framework development. Int. J. Hosp. Manag. **104**, 103240 (2022)
33. Le, Y., et al.: Age-of-information analysis for blockchain-based mobile edge computing. IEEE Trans. Commun. **5**, 823 (2025). https://doi.org/10.1109/TCOMM.2025.3597823
34. Zadobrischi, E., Havriliuc, Ș: Enhancing scalability of C-V2X and DSRC vehicular communication protocols with LoRa 2.4 GHz in the scenario of urban traffic systems. Electronics **13**(14), 2845 (2024). https://doi.org/10.3390/ELECTRONICS13142845

35. Borah, A., Paranjothi, A., Thomas, J.P.: A survey on distributed approaches for security enhancement in vehicular ad-hoc networks. Comput. Netw. **261**, 1140 (2025). https://doi.org/10.1016/J.COMNET.2025.111140
36. Ahsan, S.I., Legg, P., Alam, S.M.I.: An Explainable Ensemble-based Intrusion Detection System for Software-Defined Vehicle Ad-hoc Networks. https://arxiv.org/pdf/2312.04956v5 (2024). Accessed on 23 Oct 2025
37. Khedkar, S., Mahajan, R.: View of optimized and efficient authentication in VANET using blockchain. Manag. J. Adv. Res.
38. Mahi, M.J.N., et al.: A review on VANET research: perspective of recent emerging technologies. IEEE Access **10**, 65760–65783 (2022). https://doi.org/10.1109/ACCESS.2022.3183605
39. Li, X., Yin, X., Ning, J., Zhang, T.: AuthUp: a privacy-preserving V2I authentication scheme for vehicular ad hoc networks. IEEE Trans. Veh. Technol. **35**, 603 (2025). https://doi.org/10.1109/TVT.2025.3567603
40. Mendes, B., Araújo, M., Goes, A., Corujo, D., Oliveira, A.S.R.: Exploring V2X in 5G networks: a comprehensive survey of location-based services in hybrid scenarios. Vehic. Commun. **52**, 100878 (2025). https://doi.org/10.1016/J.VEHCOM.2025.100878
41. Alrehan, A.M., Alhaidari, F.A.: Machine learning techniques to detect DDoS attacks on VANET system: a survey. In: Proceedings of the 2019 2nd International Conference on Computer Applications and Information Security (ICCAIS), Riyadh, Saudi Arabia, pp. 1–6 (2019). https://doi.org/10.1109/CAIS.2019.8769454
42. Shah, G., Zaman, M., Saifuddin, M., Toghi, B., Fallah, Y.: "Scalable cellular V2X solutions: large-scale deployment challenges of connected vehicle safety networks. Autom. Innov. **7**(3), 373–382 (2024). https://doi.org/10.1007/S42154-023-00277-6/FIGURES/6
43. Lang, P., et al.: Towards 6G vehicular networks. Comput. Netw. **257**, 916 (2025). https://doi.org/10.1016/J.COMNET.2024.110916

Chapter 8
Navigating Ad Hoc Networks: Current Challenges and Future Perspectives



Aradhana Behura, Pabitra Mohan Khilar, and Abir Sen

Abstract Over the past few years, ad hoc networks have developed very fast and become a leading and emerging field of research. Ad hoc networks are without fixed infrastructure and consist of many categories such as Mobile Ad Hoc Networks (MANETs), Vehicular Ad Hoc Networks (VANETs), Flying Ad Hoc Networks (FANETs), Underground Ad Hoc Networks (UNDGANs), and Sensor Ad Hoc Networks (SANETs). They are general-purpose in applications like disaster management, military missions, environment monitoring, surveillance, safety systems, security services, weather forecasting, and water quality analysis. Ad hoc networks are categorized by communication methods, signal traveling, mission goals, usage environments, mobility behavior, and node deployment. Each category has a different domain. MANETs enable communication between mobile users within terrestrial environments, with the ability to have flexibility in environments with low infrastructure. VANETs provide vehicle-to-vehicle and vehicle-to-infrastructure communication, facilitating road safety and traffic efficiency in urban, rural, and highway environments. FANETs facilitate high-speed aerial communication between UAVs (drones), appropriate for surveillance, reconnaissance, and remote sensing. UNDGANs are designed for communication in underground environments, appropriate for mining, pipeline monitoring, and tunnel surveillance. SANETs function in aquatic environments, having a key role in marine life tracking and underwater condition monitoring. Although they are increasingly significant, there has been a lack of comprehensive and concerted reviews on all these subdomains. This survey will act as a basic reference point, investigating architecture designs, real-world deployment, key features, MAC protocols, routing methods, security issues, comparative

A. Behura (✉) · A. Sen
CSE, KIIT University, Bhubaneswar, Odisha, India
e-mail: aradhana.behurafcs@kiit.ac.in

A. Sen
e-mail: abir.senfcs@kiit.ac.in

A. Behura · P. M. Khilar
CSE, NIT Rourkela, Rourkela, Odisha, India
e-mail: pmkhilar@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_8

studies, current research activity, various challenges, and future directions in ad hoc networking.

Keywords VANET (Vehicular ad hoc network) · MANET (Mobile ad hoc network) · SANET (Sea ad hoc network) · FANET (Flying ad hoc network) · Underwater network · and UNDGAN

Abbreviation

USNs Underwater sensor networks
DRL Deep reinforcement learning
UACNs Underwater acoustic communication networks
AUV Autonomous underwater vehicle
RWP Random Way Point mobility model
EELA Energy-Efficient Localization Algorithm
UAN Underwater Acoustic Network
APD-JFAD Accurate Prevention and Detection of Jelly Fish Attack
NLOS Non line of sight
LOS Line of sight
C-EWA Chronological-earth worm optimization algorithm
AGEN-AODV Automata and Genetic based Ad hoc On-Demand Distance Vector
AUV autonomous underwater vehicles
CRC cyclic redundancy check
MLLE Multiple lines and levels of evidence
TXOP Transmission Opportunity
HCN Host centric network
KBIDC Knowledge based intrusion detection
MANET Mobile Ad-Hoc Network
SANET Ship Ad-Hoc Networks
VANET Vehicular ad-hoc network
FANET Flying ad-hoc network
AGPS Assisted Global Positioning System
GPS Global Positioning System
FHSS Frequency-hopping spread spectrum
DSSS Direct sequence spread spectrum
V2X Vehicle to everything
I2I Intra infrastructure communication
V2V Vehicle to vehicle communication
V2I Vehicle to infrastructure communication
V2S Vehicle to sensor communication
V2PD Vehicle to personal device communication
V2P Vehicle to pedestrian communication
V2CN Vehicle to cellular network infrastructure communication

C-V2X Cellular V2X LoRa/LoRaWAN: Low Power, Wide Area (LPWA) networking protocol

8.1 Introduction

An ad hoc network is a temporary communication system or infrastructure-less system which composed of a set of sensor nodes and transceivers. Mobile communication architecture has the infrastructure, and wireless networks are categorized into two groups: infrastructure-based and infrastructure-less [1]. In an infrastructure-based network, the mobile sensor node can communicate with a nearby base station (BS) within that communication area. In this scenario, a mobile sensor node is the same as a terminal. The terminal does not support routing tasks, and switches can handle switching and data transmission. The cellular system and college LAN are an example of an infrastructure-based network. Each node can dynamically communicate with nodes with a limited coverage area and find paths to other sensor nodes. Due to sensor node mobility, this system does not support fixed topology. On the other hand, if a sensor node desires to send information to another sensor, additional sensors must be turned into a routing system to transmit the information to the envisioned sensor.

The review's objective is to explain the recent trend of ad hoc networks. The primary importance of the paper is surveys: it briefly presents a sequence of fundamental concepts regarding ad hoc networks, mainly containing the classification, explanation, characteristics, advantages, disadvantages, applications, development, and challenges of the networks [1].

In MANET, mobile nodes can change their positions independently in any direction and suffer unpredictable topology changes. The self-motivated surroundings of mobile sensor nodes of MANET challenge not only the stability of the system but also the connectivity of the mobile node. Here, mobile nodes are considered autonomous. The configuration of mobile sensor nodes needs self-configuration, and the nonexistence of a centralized mechanism to build a network. Stability, robustness, and scalability, along with connectivity, are additional issues controlled by clustering techniques. It is challenging to configure the clustering process with suitable quality of service (QoS) parameters [1, 2]. There are two coverage-controlling issues. First, how to solve the conflicts, deciding which sensor nodes must be switched off to save energy, the duty cycle concept can be used to resolve this issue effectively. The second problem is designing the best wake-up system that eliminates awakening more sensor nodes than needed [3, 4]. The clustering technique is used to solve these types of issues [5, 6]. Many clustering algorithms suffer from exploitation and exploration issues. Maintaining a balance between exploitation and exploration is a big challenge [7].

VANETs are an expanding area of smart transportation systems. VANETs support various types of communications, including vehicle-to-vehicle (V2V), vehicle-to-pedestrian (V2P), vehicle-to-roadside unit (V2R), and vehicle-to-infrastructure

(V2I) communications [8, 9]. This paper provides a tabular summarization of VANET applications, related challenges, and research solutions already proposed [10, 11]. It also showcases how new technologies make a positive contribution towards increased security, smart resource usage, better dependability, driver behavior monitoring, high mobility support, optimal coverage, fault tolerance, shortest path calculation, efficient task scheduling, improved resource utilization, reduced link breakage, minimized delay, and decreased congestion. Current advancements also help increase packet delivery rates via multi-hop communication, improving link quality and channel utilization, decreasing link failure, increasing network lifetime, facilitating task offloading, enhancing throughput, and decreasing latency and energy consumption.

The FANET can be applied to monitor real-time applications. In low altitudes, drones can be used to gather information. At medium altitudes, the airplane can be applicable to move from one place to another. At high altitudes, satellites can be used to gather information. The drone helps to monitor the impact of urban air pollution and climate change on the archaeological heritage monuments of Bhubaneswar, Odisha. UAVs can monitor the current state of monuments concerning air pollution and hydro climatological factors. It predicts monument conditions and future weather and air pollution effects on the monuments.

The most significant issues for multi-UAV structures are data transmission, which is essential for collaboration and cooperation among the UAVs. If all airplanes are associated with the infrastructure, for example, a base station, the data transmission among UAV nodes can be comprehended through the air infrastructure [12]. As an outcome of the quick technological improvements in communication technologies and smart electronic sensors, it has been potential to build UAVs, which can be functioned remotely or fly autonomously without taking any human help. According to their flexibility, lower operating expenses, easy installation, and versatility, the UAVs can be used in traffic monitoring, military application, remote sensing, border surveillance, disaster monitoring, wildfire monitoring, and wind estimation [13].

The SANET is a subpart of the USN or underwater sensor network that deals with acoustic communication between ships with base stations or satellites. Some floating sensor helps to monitor the pollution rate in the water means, providing information regarding water quality. In UWSN, the GPS fails due to radio signals, noise, or high attenuation [14]. Underwater communication supports water sensor nodes, such as surface buoys, anchor nodes, and sensor nodes. One significant issue of USN is that sensors have fewer resources because of the non-rechargeable batteries, and it controls the lifetime of the underwater network. Furthermore, USNs deal with acoustic communication, which suffers various challenges such as multipath fading, limited battery power, frequency dispersion, sparse deployment and limited energy with bandwidth [15]. The topology control with an efficient localization method helps save energy and provides less delay and reasonable error with efficient geo-routing protocols or faster data transmission [16, 17].

The underground network is intended primarily for the subsurface region. The various applications include observing infrastructure, shortest route and home infrastructure design, precision agriculture, earthquakes, and landslides. The cost of underground network architecture is high owing to the necessity for particular set-up for

emerging underground networks, such as maintenance, cost, and careful planning. Figure 8.5 depicts an overview of the underground network.

The comparisons between SANET, FANETs, MANETs, and VANETs are explained first, and the design open design challenges, applications, and some current protocols are discussed in Table 8.1. The review is organized as follows. Table 8.2 represents comparisons regarding various ad-hoc networks. Table 8.3 depicts the roadmap to autonomous driving. Table 8.1 represents various challenges, issues, and solutions. This schematic review solicits contributions that include theory, applications, and design methods of intelligent systems and various applications embedded in the fields of cutting-edge technology and the methodologies behind them.

8.2 Mobile Ad-Hoc Network (MANET)

In MANET, sensor nodes can change their positions independently in any direction. It does not support any infrastructure where each sensor node is considered a routing device. The routing devices randomly change positions to establish themselves. Therefore, the topology may change frequently.

8.2.1 Application

The MANET can be used in defense applications, crisis management, telemedicine, virtual navigation, tele-geoprocessing, and educational system. Defense applications need sensor networks, and on-fly data transmission is key for battlefield management. Restoring data communications infrastructure is essential for crisis management. The telemedicine system assists the surgeon in collecting the medical record through video conference. Integrating GIS and GPS enables a high-capacity system known as a tele-geoprocessing application. The remote storage covers the graphical demonstration of streets, large metropolis, and the internal layout, which helps to find any rescue plan in an emergency. Education via the internet: informative opportunities accessible in remote areas or the internet owing to the financial infeasibility of giving subscribers costly wireline connection access in these remote areas.

8.2.2 Characteristics and Issues

The MANET has various characteristics such as low bandwidth, dynamic topology change, infrastructure-less, multi-hop, self-organizing, rapidly deployable, and less battery power. So, optimization of power conservation is essential. The MANET has a lower communication range due to noise, multiple access, signal fading, and interference. This system does not have centralized control, so it collaborates with other

participating sensor nodes. The nodes have less memory; scalability is challenging for an extensive network. The MANET architecture supports decentralization, so the protocol has various challenges, such as security threats, routing overheating, limited bandwidth, scalability, energy consumption, packet loss and dynamic topology change. The sensor nodes may behave as dumb due to the presence of fog or another environmental phenomenon, such as heavy rainfall or temperature. It is a challenging task to maintain connectivity in the network.

8.2.3 Details About Architecture

Figure 8.1 shows the MANET architecture, and MS is denoted as a mobile sensor node. The MS2 node changes its position from MS2 to MS4. Here we consider it impossible to have mobile sensors within each other's communication range altogether. The symmetric links mean if MS3 is within its communication area of MS1, then MS1 is within the communication range of MS3. In MANET, some nodes are stationary, and many sensors are mobile. Thus it is a challenging task to detect a mobile node's movement. For example, Fig. 8.1 signifies MANET; here, we have some mobile nodes that can communicate among themselves. Node 1 can interconnect with other nodes (Devices 2, 5, 4, and 3), but if the other node desires to exchange messages with other nodes (for example, nodes 2, 4, 5, or 3), they must follow through mobile node 1. The DoS attack, eavesdropping, and spoofing are security threats to mobile networks.

8.2.4 MAC Protocol Design Issues

MAC protocols are categorized according to wireless channel access technique and classified as: scheduling-based(SBP) and contention-based protocols(CBPs). In contention-based protocol, mobile nodes share a medium to distribute information, so collisions in data transmission can occur when multiple nodes try to transmit data simultaneously. So, the mobile nodes attempt to reduce crashes by sending data from source to destination. The CSMA/CA (Carrier sense multiple access/collision avoidance) is an example of a CBP. The scheduling-based MAC is categorized as TDMA(Time Division Multiple Access), FDMA (Frequency Division Multiple Access), CDMA (Code Division Multiple Access), and OFDMA (orthogonal frequency division multiple access). In SBPs, sensors transmit information by considering pre-assigned resources; thus, there is no channel access collision. In CBPs, energy is wasted due to corrupt packet retransmission. In SBPs, energy is wasted due to signaling overhead in the resource reservation procedure but reduces exposed or hidden node issues [18]. The MAC system has to be designed to handle QoS-based challenging problems in terms of faster channel utilization, congestion less, bandwidth utilization, energy efficiency, fairness, and scalability.

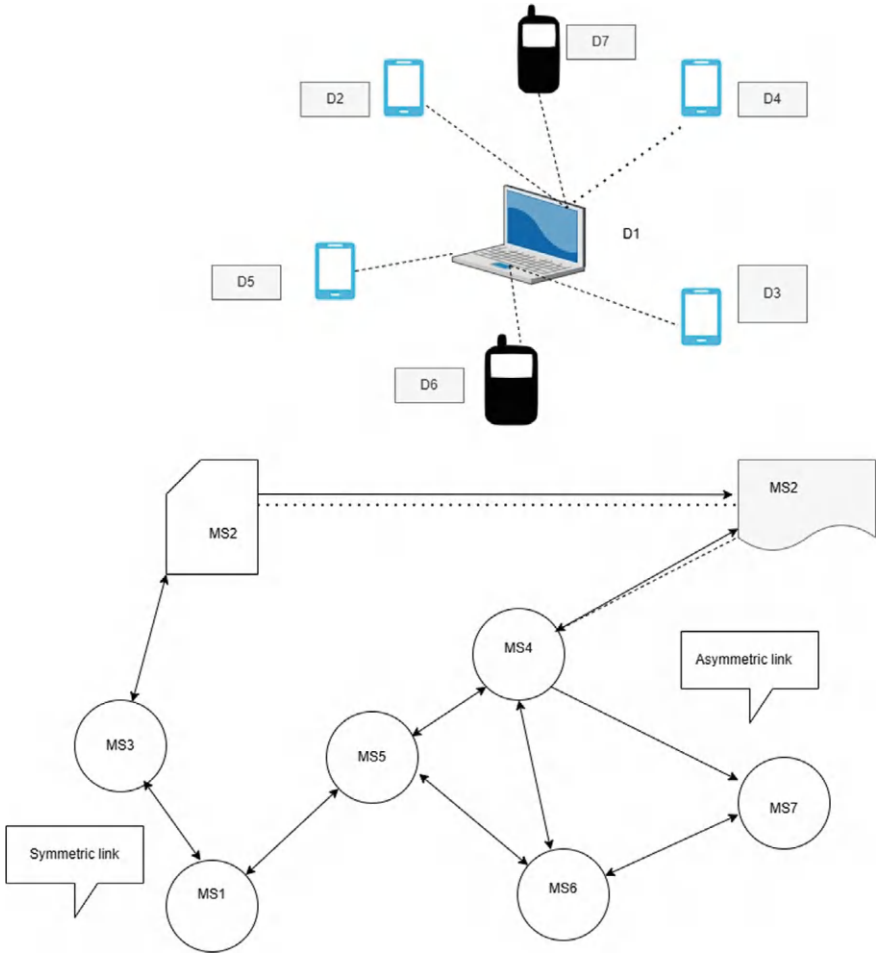


Fig. 8.1 Mobile ad hoc network (MANET) architecture

8.2.5 Topology Management and Energy Efficiency Technique on HCN

Power can be optimized using the C-EWA and topology management scheme [19]. The C-EWA algorithm is used to choose the cluster head, and the objective function consists of several factors such as distance, power, mobility, connectivity, and link lifetime. The Gabriel graph concept is created within the cluster using chronological theory. Inter-cluster and intra-cluster distance minimization is a challenging task in topology management. The fitness function consists of five factors: power, mobility, connectivity, squared distance, and link lifetime. Gabriel graph is constructed for the individual cluster to reduce the data transmission power.

Host-centric energy efficiency technique [20] is categorized as power-aware routing, load distribution, data transmission power control, powerâŁ“down or sleep mode, and cross-layer design. Energy or power-aware routing suffers from unbalanced data consumption. The load distribution criteria create a loop in the multipath routing–more hops results in packet loss and delay, which is a disadvantage of transmission power control. The cross-layer mechanism deals with information gathering, measurement, and stability mechanisms.

8.2.6 Concern Regarding Security

The researchers are focused on various layers and attack types. The application layer deals with malicious program injection in mobile nodes and repudiation. The transport layer deals with session hijacking and SYN flooding. The network layer deals with replay attacks, blackhole, path cache poisoning, wormhole, overflow of the routing table, byzantine, flooding, data disclosure, gray hole, link spoofing, and rushing attack. The MAC layer suffers from jellyfish attacks (one type of DOS attack), bandwidth stealth, and targeted attack. The data link layer deals with traffic attacks, DOS attacks, and MAC-targeted attacks. The physical layer deals with the compromised attack, jamming, eavesdropping, malicious data injection, and device tampering. The Jelly Fish attack detection in MANET degrades the network performance. The SVM technique [21] detects attacks and learns packet-transmitting behavior. The SVM is used to predict whether the situation is abnormal or normal using various parameters such as security, topology change, unpredictable link, dynamic mobility, limited sensors energy, and reliability). The variants of jellyfish attacks can be categorized as periodic dropping attacks, delay variance, and reordering attacks. Multipath-based secure routing is a challenging task. The hybrid genetic algorithm (GA) and Hill climbing are used to select the optimal path among the multipath schedule [22]. The C-means clustering technique is built on cluster head, and density peaks are chosen based on direct, indirect, and recent trust [23, 24]. In KBID protocol, knowledge-based signature patterns and rules are used to generate alerts towards unknown attacks [20, 23, 25–27].

8.2.7 Routing Protocol Scheme

Researchers have designed many routing protocols which are based on various problems, topologies, and applications. The protocol aims to reduce packet loss while increasing throughput and minimizing overhead. The AGEN-AODV routing protocol focused on the learning automata to choose the optimal path by considering parameters such as stability, hop count, energy and traffic, delay, energy consumption, and network lifetime using a genetic algorithm in heterogeneous networks [28].

Nowadays, routing protocols have various limitations, such as no link quality, high cost, a long path with control overhead, the shortest path with long delay, dynamic topology change, congestion control, resource consumption, frequent network disconnection, scalability, mobility prediction, reliability in terms of transmission errors, successive network disconnections, constrained capacity links, data transmission failure, and packet drop. Researchers have focused on router metrics such as distance, link maintenance technique, path discovery mechanism, control overhead, cross-layer, or direction.

MANET routing protocols are QoS, route discovery, and link maintenance mechanisms. The route discovery mechanism is categorized as topology-based and position based. The topology-based protocol is classified as pure flooding and partial flooding. Partial flooding is categorized as probability, area-based, and counter-based. The position-based routing is categorized as greedy, hierarchical approaches, and restricted directional flooding. The greedy-based process follows a node selection and recovery strategy [29, 30]. The QoS routing is categorized as link-based and node based. The link-based routing is classified as a single path, multipath, shortest path, stable path, and power-aware. The node-based routing is used for active monitoring. The link maintenance-based routing is categorized as alternative path, multipath and local repair. Static mobility of the sensor node is a limitation of the sensor node. In this type of routing, the sender sends data to the receiver by relying on the link information of the available network. Packet forwarding choices in the geographical routing concept are based on the position of the receiver and the position of directly connected neighbor nodes. Using position, speed, direction related information regarding nodes can predict the link breakage and connection time. These routing protocols focused on link maintenance, route discovery, and QoS. The AC OLSR routing protocol [31] used an ant colony optimization algorithm to get an optimized shortest path from source to destination by taking parameters such as node speed, packet delivery rate, energy cost, simulation cost, and throughput.

The Sybil attack is a security threat that targets mobile network areas where an individual creates multiple fake or duplicate identities to achieve disproportionate control over the network. Suppose a vendor on the Myntra website generates various fake profiles and contributes the best comments to their goods for improved reach. The susceptibility of some targeted schemes can be evaluated based on how simply the fake profiles can be generated and the grade of considering the inputs without trusted data like authorized identity or certificates proofs. An intruder of the Sybil attack [32] could develop several profiles on one physical system for launching any modification through an attack that is synchronized on the system; this can change profiles for dwindling the attack prediction process, so supporting no responsibility in the mobile network. To prevent direction-related dropping packets, the selected path will not be used for the data transmission. Once the path discovery phase is complete, the sensors will reach the data authentication stage for getting authenticated by the neighboring sensor nodes. A malicious sensor node links the network or system with various IDs. Proof of Personhood technique such as biometric authentication (palmprint, iris, and fingerprint) is used to resist Sybil's attack in the network [32].

Reinforcement learning is achieved through proper interaction between the decision-maker and the learner. This technique gives a reward for every action. The reward may be positive or negative. The subsequent action receives compensation, and an unsuccessful effort receives punishment. The sensor node picks one sensor among the accessible sensor nodes for packet transmitting. The objective of the clustering process is not only to rank the action but also to moderate the figure of actions. The actions are responsible for estimating the group's success rate, packet transmission rate, and delay. After collecting a sensor group that performs well in packet transmission, the sensor with the lowermost figure of the hops is chosen as the upcoming sensor to transmit data packets [33, 34].

8.3 Vehicular Ad Hoc Network

A Vehicular Ad-Hoc Network or VANET is an approach to study the communication between V2V and V2I to provide efficient and safe transportation and improve communication between V2V and V2I. Various communication techniques are available in the vehicular area, including vehicle-to-pedestrian (V2P) and ad-hoc data communication. Combinations among V2R, V2V, and RSUs to TA (trusted authority) facilitate automobiles communicating directly with infrastructure to maintain safety and security. RSUs(roadside units) are fixed infrastructure along roads fitted with equipment for short-distance wireless data transmission, founded on IEEE 802.11p standard, with Dedicated Short Range Communication (DSRC). Figure 8.1 demonstrates a trusted vehicular infrastructure with various data transmission modes?. The system of the anonymous authentication scheme described in the context of a single geographic region consists of three main components: trusted authority (TA), roadside units (RSUs), and vehicles. The Trusted Authority (TA) is the central entity responsible for managing the entire vehicular area. It is considered fully trusted, meaning adversaries cannot compromise it. The TA handles the registration process of both RSUs and vehicles when they join the network. This scheme divides the road transportation system into multiple geographic regions, each with its own TA. When a vehicle moves from one region to another, it undergoes authentication by the TA of the new area. This authentication is done using the shared values exchanged between TAs to verify the legitimacy of the vehicles during roaming. Additionally, the TA generates initial security parameters for vehicles and RSUs and issues them after a successful registration. The roadside units (RSUs) are fixed infrastructure units deployed along the roadside. They serve as intermediaries between the TA and vehicle users, establishing secure links with the TA and wireless communication channels with vehicles. RSUs and vehicles are equipped with DSRC radios to facilitate V2V (Vehicle-to-Vehicle) and V2R (Vehicle-to-Roadside) communications. In this scheme, RSUs are considered semi-trusted, meaning they generally function as expected but may potentially disclose data to adversaries. If an RSU is compromised, the TA can detect it within a limited period. Each automobile has an On-Board Unit (OBU), enabling Communication with other vehicles and RSUs to exchange mes-

sages and enhance driving experiences. The OBUs installed in vehicles make them intelligent transport vehicles. Each OBU contains a tamper-proof device (TPD) to store sensitive information, such as secret keys, securely. It also includes a Global Positioning System (GPS) to provide location and time information and an event data recorder (EDR) to record relevant data about vehicle crashes or other incidents. In summary, the described system model illustrates the interaction among the Trusted Authority, Roadside Units, and vehicles in a single geographic region. The TA manages the entire VANET system and authenticates vehicles during roaming. RSUs act as intermediaries, facilitating secure communication between vehicles and the TA. Vehicles are equipped with OBUs, enabling them to communicate with other vehicles and RSUs while storing sensitive information securely.

8.3.1 Application

The VANET application is categorized into two parts: entertainment or comfort and safety applications. Comfort application includes weather or traffic data, passenger's comfort level, and nearest hotel or accommodation location. The safety application takes care of intersection collision avoidance (example: warning about violating traffic signals, warning about blind merge detection, intersection warning), public safety (example: post-crash warning), sign extension (example: curve speed warning, wrong-way driver warning, work zone warning), vehicle diagnostics and maintenance (example: safety recall notice, just in time repair notification).

8.3.2 Architecture

Vehicular Ad-hoc Networks (VANET) can be considered autonomous and self-configurable wireless ad hoc networks. It uses vehicles as mobile nodes to create a network and helps to decrease traffic deaths and improve travel comfort by increasing inter-vehicle coordination. The most commonly considered applications are related to public safety and traffic coordination. The procedure of sending data to several users simultaneously in a computer network, i.e., the sender has the choice of sending the data to some users. Due to the real-time mobility feature of a node (vehicle) VANET network, it is difficult to harness the data now. Moreover, connectivity between nodes and infrastructure forms the primary basis for exchanging information and taking the necessary action based on the data.

Routing is one of the most important components in the operation of any ad hoc network. Therefore, a good routing protocol must be in place to provide efficient message delivery from a source node to one or more destination nodes. To deal with some of the above-mentioned issues, multicast routing is used. It is a method of efficient management of resources and mitigates network congestion and workload

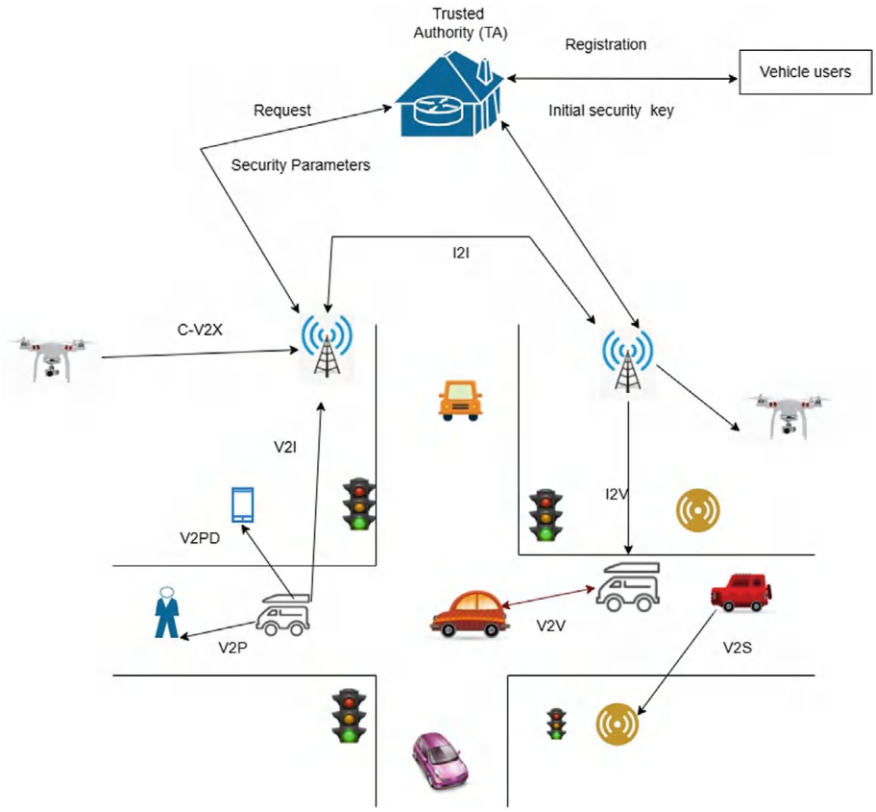


Fig. 8.2 Vehicular ad hoc network (VANET) architecture

by sending multiple copies of messages to various vehicles simultaneously. Due to this, it saves power usage, decreases transmission and control overhead.

From Fig. 8.2, we conclude different vehicular communication strategies such as V2X (vehicle to everything), I2I (infrastructure to infrastructure), V2V (vehicle to vehicle), V2I (vehicle to infrastructure), V2S (vehicle to sensor), V2PD (vehicle to personal device), V2P (vehicle to pedestrian), V2CN (vehicle to cellular network), and C-V2X (cellular vehicle to everything).

8.3.2.1 RSU Node Deployment Strategies

In the clustered coverage communication area, RSUs can act as a gateway to collect vehicular information to broadcast safety and secure information to other RSUs (RSU 2 RSU) or I2I (infrastructure to infrastructure) to facilitate load-balancing and faster data communication; in addition, an automobile can transmit data through multi-hopping whereby in a single hop other automobiles unable to communicate with

RSUs. RSUs deployment is considered an NP-hard problem with multi-objectives such as load balancing, cost minimization, reducing communication delay, maximizing network coverage, optimizing connection, minimizes latency and interconnection gaps. The aim of RSUs deployment is used to improve data transmission coverage, which changed the node deployment issue into the Coimbatore's optimization such as set coverage problem (SCP), vertex coverage (VCP) problem, maximum coverage problem (MCP), Knapsack problem, and facility location problem. In urban areas, the RSUs coverage network is affected by various obstacles, such as trees, buildings, and flyovers.

8.3.3 Characteristics and Issues

The main aim of ITS focuses on the minimization of fuel consumption, utilization of traveling time, rate of road accidents, smart city, traffic management, and autonomous driving.

Big data analytics are used to handle a large amount of data, platoon-based driving or traffic efficiency, throughput, fuel consumption, reducing the emission of the vehicles, vehicle monitoring or reliable data analysis, telematic features or emergency assistance, automobile users' comfort, safety, security, environmental pollution monitoring, and channel coordination.

Most VANET applications require that the data be transmitted in a multi-hop fashion, thus prompting the need for a routing protocol. In multicast routing protocols, messages are transmitted from a single sender toward a group of interested nodes, which would be our destination nodes. Therefore, we can use multicast routing protocols to resolve some situations in the most effective way that is countered above. Hence, Multicast routing protocols are the most active research area due to their efficiency and mobility within a dynamic environment like VANET. Due to the fast movement of the vehicles, VANET needs to be entailed with a routing path between the source and destination node that is up-to-date and needs to be frequently constructed, taking into consideration the position of the current nodes. Here we need information exchange for which we need a routing algorithm. It is the crux of the routing protocol in charge of constructing a connection between the source node and destination nodes. Therefore we need QoS when the routing algorithm has built a path and we need to exchange the information. The vehicular routing parameters in VANETs are buffer, path length, data delivery rate, packet loss ratio, communication delay, bandwidth delay, transmission delay, delay jitter, and transmitter and receiver power. However, QoS-constrained multicast routing is an NP-complete problem.

Only heuristic algorithms can solve this because the solution is the challenge of falling into the local optimal space instead of the optimal global space. The nature-inspired algorithm helps to minimize exploitation and exploration in the algorithm and helps find an optimized route from the source to the destination [11].

Bajracharya et al. [35] proposed a dynamic price [35] procedure (vehicular communication by taking the help of an unlicensed band) which is used to maximize

profit and utility for 6G to choose high-amount paying customers. They used various pricing domains such as NP(no pricing), FTSP (fixed time slot pricing), VTSP (variable time slot pricing), and NBP (nash bargaining pricing approach).

8.3.4 MAC Protocol Design Issue

Vehicular MAC protocols are classified as virtual tokens, distributed MAC, cluster-based, centralized MAC, coâŁ“operative MAC, and random section-based single and multi-channel protocols [36]. Congestion handling in the vehicular system is an important issue. The CSMA/CA protocol has been cast off for congestion control. Various alterations have also planned proficient techniques to control congestion. Alternate improved wireless channel access, scalability, latency, flexibility, overhead, safety information broadcast, and channel utilization are yet other issues and upcoming future scope in research. Exposed and hidden node issues must be shared out to improve network performance, scalability, and throughput. Time-synchronization and load balancing among automobiles are significant challenges that must be preserved to give collision-free data communication in the multi-channel developments that offer suitable time slot allocation to the vehicular nodes. Centralized MAC schemes suffer from flexibility, overhead, scalability, and latency, while the decentralized methodology is not preserved and has a high budget.

8.3.5 Routing in VANET

Rahman et al. [1] describes the role of big data and machine learning in the field of vehicular network by taking the help of IOE (internet of everything). Due to the massive amount of data, it is tedious to determine how a user can handle unstructured data efficiently, which will outperform better compared to the existing algorithms.

Cognitive Radio (CR) is a key player in numerous network operations including energy estimation, feature selection, spectrum sensing, expectation maximization, backhaul optimization, and cache management. When combined with Machine Learning (ML), Software-Defined Networking (SDN), Network Function Virtualization (NFV), and Multi-access Edge Computing (MEC), the considered protocol substantially improves load balancing, resource allocation, and network virtualization in comparison to traditional approaches. Neural networks, specifically Convolutional Neural Networks (CNNs), are successfully implemented in vehicle license plate recognition, traffic sign detection, and traffic light detection, as mentioned in [1]. The Optimized Decision Model (ODM) enables efficient spectrum sensing, channel estimation, radio resource management, routing, and traffic control.

Wireless communications encounter a number of challenges, including modulation classification, anti-jamming, physical-layer coding, interference alignment, and link performance estimation. These challenges can be efficiently overcome with the

help of different ML methodologies. Fifth-generation (5G) technology is brought in to suppress interference within the network. Slepian-Wolf coding, carried out on cloud servers, not only facilitates secure and economic data processing, storage, and transmission but also becomes an essential mechanism in this regard.

This paragraph also underlines the significance of data analytics in cloud environments, especially handling huge volumes of duplicate vehicular data. Cloud computing has an important role in data storage and management efficiency. The Hidden Markov Model (HMM) is used to forecast data transfer results, determine idle states, and handle intricate issues like multi-hop communication, inefficient cluster coordination, hidden and exposed terminal issues, dynamic topological changes, and diverse traffic conditions (saturated and unsaturated) [20].

Three major issues are solved effectively using HMM methodologies:

- Sequence likelihood estimation—addressed with the forward-backward algorithm.
- Optimal state sequence determination—resolved via the Viterbi algorithm.
- Parameter re-estimation—accomplished with the Baum-Welch or forward-backward technique.

The probability of the observation sequence is known as the likelihood of a sequence. The best possible state sequence is given as the observation sequence to compute the probability of the state sequence. So, we would like to find out the best possible state sequence, given the observation sequence and the meaning of the best possible here is the sequence with the highest probability, given the observation sequence. The third problem is to find out the transition and observation probabilities or the parameters of HMM, given the output sequence or the observation sequences, which are accepted by the machine or generated by the machine. Yao [3] proposed PRHMM routing using a forward-backward algorithm [3, 37]. This protocol not only provides mobility patterns but also gives a faster data transmission rate.

One of the significant disadvantages of the algorithm is its random choice of cluster heads upon the start of every round. In order to mitigate this disadvantage, solutions such as Distributed Load Balancing with Unequal Clustering using Fuzzy logic (DUCF) have been introduced. In these fuzzy-based clustering protocols, residual energy and node degree are taken into consideration during every round for the selection of cluster heads.

Yet, repeatedly choosing cluster heads in each round leads to higher energy consumption, and this is one drawback of this approach [38]. Most researchers have considered designing Driver Assistance Systems (DASs) and Driver Monitoring Systems (DMSs), which are key elements of intelligent transportation systems [39–42]. Fuzzy logic has been found promising in facilitating energy-efficient solutions in VANETs. In all these researches, it is presumed that all RSUs and nodes are static, and distances are calculated based on the Euclidean approach. Sensor nodes are placed in a given area to send messages to RSUs, and the use of the fuzzy membership function helps ensure efficient communication, aggregation, and retrieval [2].

Support Vector Machines (SVMs) are trusted machine learning models used to train vehicular datasets. Blockchain technology can be used once the system has

been trained to improve system security, albeit at the expense of very high time usage and computational overhead. The foremost application of vehicular networks is to provide seamless connectivity between drivers and passengers on urban roads.

With increasing global traffic jams, contemporary traffic management systems are hoped to effectively tackle this problem. Signal control is the central component in most of the smart traffic solutions. Creating smart cities demands sophisticated traffic management and effective processing of big data, realizable by integrating the Internet of Things (IoT) with machine learning. To address these challenges, this review discusses an intelligent traffic control protocol based on reinforcement learning for optimizing vehicle throughput, balancing traffic signals, and regulating road intersections [43].

The Q learning technique deals with exploitation and exploration. The exploration deals with local search, and exploration deals with global optimization. The exploitation criteria select an action maximizing the reward from learned data. It creates the randomly best suitable choice with the past learned data which challenges to meet various experiences. The exploitation criteria use history and recently learned data to analyze the task, which helps for the next signal in the traffic. The exploration allocates action randomly, which receives a traffic signal. Q learning helps to minimize the standard deviation and maximizes the throughput of the queue lengths. Throughput denotes the total number of automobiles managed at the road intersections and the standard deviation is calculated to the traffic balance of the road in the queue.

8.3.6 Concept Regarding Autonomous Driving

This review fills gaps in the current literature by discussing recent developments in secure driver support technologies. It also investigates ongoing issues in creating secure driver support systems, where it is stressed that most studies center on improving ad hoc network technologies, the Quality of Service (QoS) being a top priority issue. Our review is balanced between theoretical findings and realistic outcomes and discusses several architectural models that have been implemented to accomplish various safety-related functions and emphasizes the advantages of varied prototypes.

Deep learning methods are an important part of autonomous driving systems. This survey puts focus on both existing challenges and prospective research areas. Deep learning has outperformed many tasks, such as detecting pedestrians, detecting drowsiness, avoiding collision, detecting traffic signs and lanes, detecting road surface, and detecting vehicles. Additionally, it enables QoS-driven services such as driver behavior prediction, pedestrian-autonomous vehicle interaction modeling, and traffic sign recognition. Table 8.3 provides the road to development for achieving complete autonomous driving.

8.4 Flying Ad Hoc Network (FANET)

The collaboration and synchronization among the sensors are essential for data transmission in FANET. The FANET topology frequently interchanges as compared to MANET or VANET architecture. There is various simulator used to simulate protocols. The simulators are BonnMotion, FANETsim, NS2, AVENS, NS3, Ardupilot, NS3, Omnet++, ONE, OPnet, ONE, Netsim, GlomoSim, QualNet, TOSSIM, SSFNet, VC++, J—sim, MATLAB, and YANS.

8.4.1 Application

The UAVs help to handle battlefield, and rescue operations, flooding, earthquake, and agricultural field monitoring [44, 45], and architectural planning. The UAV is useful for classification and crop monitoring for intelligent smart farming. Bho et al. [46] proposed IOT and AI-assisted UAV model for rice pest prediction in an intelligent manner [46]. Researchers deploy multi UAVs in the agricultural field for data-relevant collection. After collecting data, UAV will send this data to the server for pre-processing and analysis using AI. But this protocol suffers inefficient task offloading and security. Mukherjee et al. [47] proposed an edge UAV-based resource allocation model for efficient path selection. But if the dataset is a massive amount, a multi-UAV performs better for data collection than a single UAV [47]. Menouar et al. [48] proposed UAV assisted smart transportation system which detects accidents and highway maintenance using floating RSU on the road [48]. Mohamed[49] discussed various applications in various fields for the smart city [49]. Forgery can be eliminated using proxy signature and UAV to maintain a smart city [50].

8.4.2 Architecture

The communication system of the FANET is classified as U2I and U2V, shown in Fig. 8.3. In a flying ad hoc network, the sensors exchange information among themselves using multi-hop data transmission, where one sensor is allied to the associated infrastructure and other has multi-hop data communication. Thus, UAVs don't have to interconnect with base stations or satellites. The GPS helps to know about geospatial localization [12, 13]. The use of multi-UAV is advantageous compared to the single UAV in terms of high scalability, extensive coverage, directional antenna, supports heterogeneous configuration, fast mission speed, high survivability, medium bandwidth, and simple topology. The single UAV has limited scalability, omnidirectional antenna, slow mission speed, high bandwidth, low complexity control, poor survivability, inapplicable heterogeneous configuration, small area coverage, and simple and direct network connection. The multi-UAV system has high scalability,

directional antenna, medium bandwidth, high complexity control, high survivability, extensive coverage, and system configuration can be done.

According to wings, UAVs can be categorized as rotary wings, fixed wings, low altitude, and high altitude UAVs. High altitude-based UAVs have long endurance and are capable of environmental monitoring. Low altitude-based UAVs have fast deployment, approximately 400 KG, and are applicable for coverage enhancement. The fixed wings-based UAV is 125 âŁ“ 1250 KG and is useful in aerial mapping & oil pipeline monitoring. The rotary wings-based UAVs are applicable in environment monitoring and help to collect data. In this situation, multi- UAV helps collect field information effectively if massive information is available. Various mobility models are used in the FANET, such as the random waypoint model, random direction, gauss-Markov, spring model, distributed pheromone repel, smooth-turn, semi-random circular movement, and paparazzi model.

8.4.3 Overview Regarding Routing Scheme

The routing algorithm can be categorized into store carry and forward, secure, greedy forwarding, energy-based, prediction, hierarchical, link state, and clustering. The cluster ground routing can be dynamic, bio-inspired, hybrid, weight-based, fuzzy-based, heuristic, and compound based. The routing approach can be categorized into static, proactive routing, reactive routing, hybrid routing, probabilistic routing, deterministic routing, stochastic routing, position-based, and hierarchical-based routing. The route reestablishment, the multipath concept used to improve survivability, path planning, less congestion, and delay, is very important. Less routing overhead, less delay, network lifetime enhancement, less power consumption, data encryption, and machine learning techniques are required for design the g a suitable mobility model for the secure routing protocol. The frequent topology change and high speed of the sensor node create a problem for localization and packet transmission within a particular short interval. The routing protocol is categorized based on the clustering concept, link state, hierarchical, energy-based, delay tolerant network ground (DTNG), position ground, heterogeneous ground, and security. The clustering-based protocol divides the area into a cluster or small zone where the coordinator controls each cluster, leading to network overhead. The link state routing provides the shortest path among the UAVs. Hierarchical routing is a tree-like structure where multiple nodes transmit data to each other. The battery-powered system and packet loss elimination are essential in the energy-based scheme. The DTNG is categorized as a social network, stochastic and deterministic. Stochastic routing deals with the epidemic, estimation, node movement, and coding. The epidemic routing is based on swarm optimization. The position ground routing is based on a multipath and single path. The multipath routing may be reactive or predictive. The single-path routing may be greedy or predictive. The topology ground routing is categorized as static, hybrid, reactive, or proactive.

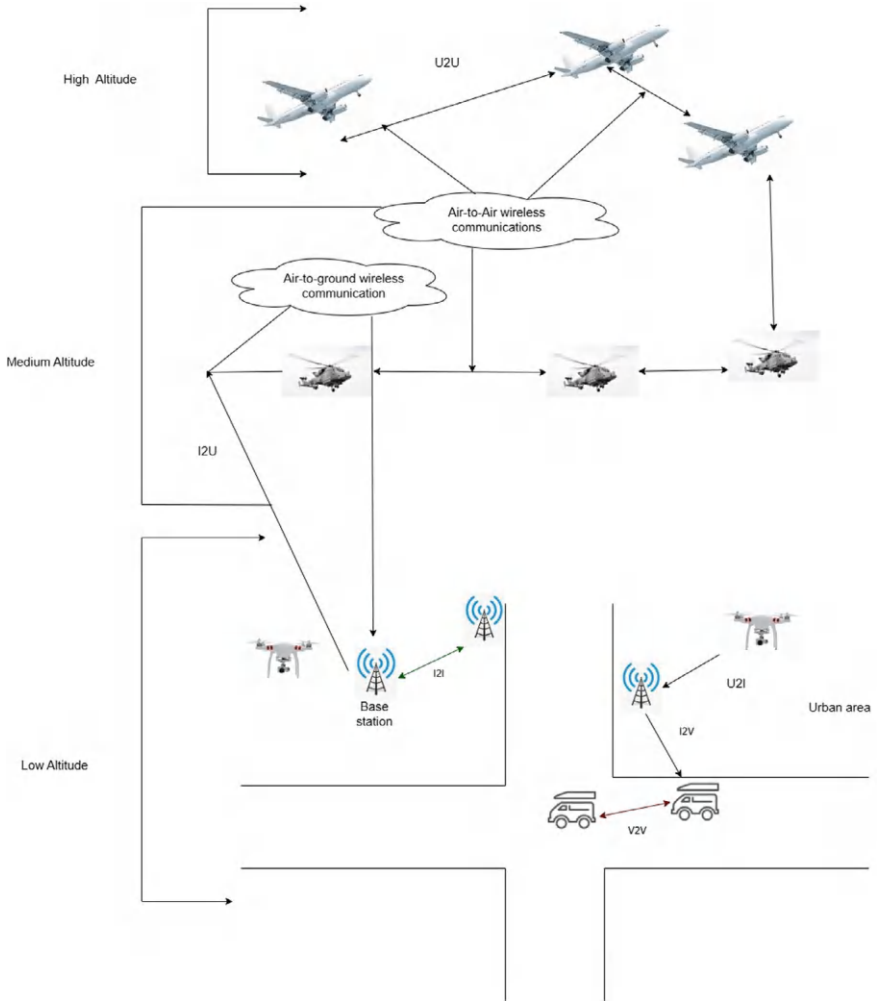


Fig. 8.3 Flying ad hoc network (FANET) architecture

By executing and observing the outcomes of those actions, an agent learns how to behave in a given environment via reinforcement learning, a feedback-based technique. The agent receives compliments for each positive activity and is penalized or given negative feedback for each negative action. The optimal design of a FANET for the cases such as routing, flight trajectory selection, and jamming protection are a challenging issue. It outperforms in terms of delay, energy consumption, and stability. This review described multi-objective routing based on Q learning. In the DQL (deep Q learning) framework, initialize the random weight and experience replay memory (ERM). Second, choose an action analyzing the present state based on the probability value. Third, analyze the immediate reward value and the upcoming state.

The mini-batch can be formulated with the random sample from ERM. Next, optimize the weight value from ERM of DNN with MDP (Markov model) with gradient descent. In QMR protocol [51, 52], Q learning helps to change exploration, learning rate, exploitation, and discount factor. This protocol consists of routing decisions, neighborhood discovery, Q learning, and a penalty mechanism. The QMR protocol collects the location information using GPS and transmits packets that contain discount factor, sensor's geo-location, queuing delay, and mobility model to initiate the path discovery process. The routing table contains MAC delay, arrival time, information regarding packets, and learning rate. The learning rate varies with delay, and the discount factor depends on the node's velocity. For faster sensor neighbors node, it consumes a low discount factor. The trade-off between exploitation and exploration depends on link quality on the packet traveling velocity over the communication links and neighbor relationship. The sender computes link quality based on the Q value to transmit the information and receive the maximum reward. Re-establishing data transmission is unreliable if a sensor gets lost.

8.4.4 Characteristics, Issues, and Challenges

The UAV system varies in speed, energy, storage, and angle constraints. The UAV categorization depends on its elevation range; it may be high altitude, medium, or low altitude. The network is disrupted by extraneous noise, inter-distance of UAV, power transmission, and sensor node mobility.

Inadequate communication, like link quality and path distance, causes delays in the network. The mobility model is randomized, time-dependent, path planning, topology control, and group-based. The elevation range of UAVs is classified as low altitude (LAU), medium altitude (MAU), and high altitude (HAU). The LAU deals with the UAV network, MAU deals with the aircraft network, and HAU deals with the satellite network.

According to wings, drones are categorized as multi-rotor drones, single-rotor drones, and fixed-wing drones. The multi-rotor has the following features maximum of 25 âŁ“ 30 minutes of flying time, restricted speed, and high energy consumed and used in inspection, aerial photography, agriculture, security, and construction.

The fixed-wing UAV has the following features: a flight time of several hours, continuous forward motion, high cost, and a gas-engine based system that can fly for at least 16 hours. It is capable of carrying heavier payloads and is commonly used for inspection and aerial mapping. The single rotor has the ability of the following feature to fly vertically, heavy payload, operational risk, and high complexity, and it is used in aerial scan and research surveying. It has a similar structure to helicopters. The fixed-wing hybrid VTOL adds the advantages of rotor-based and fixed-wing-based sensors (accelerometers and gyros).

The optimized route and faster data transmission are very crucial issues. It is a challenging task that the routing protocol must be low complexity, utilizes maximum

bandwidth, has fast mobility, low latency, no overhead, and be able to handle large and small networks.

Now we will discuss various issues of the radio propagation prototype used in FANET. Electromagnetic waves are emitted from the source to the destination through wireless mediums or channels. When analyzed with another wireless network, the FANET has numerous issues in the radio propagation model. The issues presented such as deviations in the data communication distance, link quality, trend of the collaborating pairs with the antenna data radiation pattern, environmental condition, aircraft altitude, signal or shadowing reflection effects, interferences, and hostile jamming [12]. In P2P and UAV data communication, the workload is divided through an efficient task-scheduling technique, which is challenging. Here we will discuss various research challenges regarding P2P data communications, regulatory authority, battery power, security, standardization of the data communication bands, fault tolerance, 5G impact, IoT and machine learning integration, and cross-layer routing.

The UAV is banned in most areas. So airspace registration of regulation of drones is essential for assisting and monitoring. The green energy concept can be used to utilize solar energy. Authentication, authorization, and integrity are necessary for secure data communication, freeing the channel from various possible attacks.

Standardizing the data communication bands is necessary to reduce the channel congestion issue. The UHF (ultra high frequency) is between 300 MHz and 3GHz, and the L band is 1-2 GHz. Faults can able to break down the entire network. So backup is necessary to handle these issues. The cross-layer routing is a challenging task in FANET. Low density and highly dynamic criteria are a concern in defining and designing routing. The data link and physical layer exchange data about a particular situation or condition and act accordingly. Integrating 5G, machine learning, IoT, and the cloud helps provide better decision, storage, and reliable communication.

8.4.5 MAC Protocol Design Issue [53]

The FANET MAC layer is categorized according to the antenna as: omnidirectional and directional. Omnidirectional antennae have various challenges, such as short data transmission, high energy and power consumption for all directional signal transmission, lack of location awareness, improper wireless channel access, and collision avoidance methodology. The directional antenna has various issues, such as exposed and hidden terminal problems, deafness issues for beam blockage, and unheard CTS/RTS signals. Line blocking issue arises for FIFO queuing methodology. MAC protocols are classified as contention-based, contention-free, and hybrid MAC. The contention-based MAC comprises random access, handshaking, and controlled access protocol. The random access schemes are categorized as ALOHA and IEEE 802.11. The controlled access MAC deals with token-based protocols. The SDMA and TDMA are coming under contention-free MAC. The security-aware, contention-free & based integration of TDMA and CSMA are coming under hybrid protocols.

Many problems are challenging for dynamic FANET network configuration. The problems are dynamic topology handling, power consumption, low localization error, shortest congestion-free path planning, fault tolerance, scalability, minimal latency, low coverage, communication coordination, fault tolerance, network density, proper channel allocation, mmWave communication, cross-layered architecture and secured data communication.

8.4.6 Security System

The STRIDE [54] threat system outlines the functionality of the security model, which focuses on authentication, integrity, non âŁ“ repudiation, confidentiality, availability, and authorization. The STRIDE model illustrates the surveys regarding spoofing identity, tampering with data, repudiation, data disclosure, denial of service, and elevation of privilege. This survey focused on the percentage or ratio of various security threats in FANET. The threats are active interfering, black holes , data tampering, DoS, eavesdropping, flooding , GPS spoofing, gray hole, impersonation, jamming, MITM, Modification, reply attack, rushing, selfishness attack, Sybil and wormhole. We have to design a security protocol so that it can handle all the above-said issues by considering the following parameters: high mobility, low computational power, low sensor node density, lack of standardized secure communication channel, and frequent network topology changes. Using IoT, 5G, SDN, machine learning, deep learning, and protocol performance means QoS can be enhanced.

8.5 Sea Ad Hoc Network (SANET)

The SANET is a subpart of an underwater sensor network that deals with communication between ships with base stations or satellites. Underwater communication supports these water sensor nodes, such as surface buoys, anchor nodes, and sensor nodes. SANET helps to monitor water animal condition prediction, geographic data prediction, photo-mosaicking, tsunami âŁ“ flood âŁ“ red tide forecast, leisure activities (rowing, scuba diving, sailing, snorkeling, and skiing), corrosion investigation, information regarding gemstone and types of water plant [45]. Underwater species detection is essential in terms of fish recognition, plant recognition, and coral recognition. AlexNet, self-organizing map (SOM), regression tree, LeNet, LDA with SVM, flat SV, BEOTR, decision tree with SVM, softmax, KNN, deepish, GoogleNet, VGGNet, deepish with SVM outperforms fish recognition, plant and coral recognition [55]. Energy harvesting and conservation can be affordable using tiny rechargeable batteries, which are low-cost and efficient for channel modeling, traffic control, routing, and security. Various simulators are used for SANET, such as AQUASIM, OMNET++, SUNSET, NS3, WOSS, NS2, DESERT, JSIM, Riverbed Modeler, Netsim, and QualNet.

8.5.1 Applications

The application of UWSN is divided into three categories: long-term and short-term monitoring. The long-time monitoring consists of shoaling, oil spill monitoring, climate change warnings, water quality, malfunctioning offshore equipment, and temperature-salinity [56]. The short-time monitoring describes seaquake monitoring, surveillance, navigation assistance, and military application. Temperature monitoring, disaster prevention, ocean exploration, marine life management, and multimedia surveillance are essential applications for underwater multimedia networks.

Yan et al. [57, 58] proposed a deep learning and privacy-preserving-based localization estimator that helps to minimize localization error and is helpful in marine applications. The ray compensation approach is used to eliminate sensor localization bias. Soares et al. [59] discussed the LocDyn protocol, which supports dynamic localization, helps to reject outliers, and estimates trustworthy position. Coutinho et al. [56] proposed a joint path-based routing scheme that follows the duty-cycling concept to enhance the lifetime of the network [56] and performs better in terms of energy efficiency, data delivery, and delay. Yuan et al. [17] discussed the EELA protocol regarding topology coverage control using game theory-based nash equilibrium. It provides low bandwidth, high delay with minimum cost, low error rate, and better link quality [17]. Ojha et al. [16] analyzed SEAL protocol for multi-stage coverage prediction, and dead-reckoning is used to handle the airplane's trajectory. Inter-node distance depends upon received RSSI, signal strength indication, TDoA, or time difference of arrival method. Xiao et al. [15] discussed or Q-learning-inspired anti-jamming relay protocol [16]. The Q value is restructured in reinforcement learning through the Bellman-based iterative formula, with the discount factor and the learning rate [15].

Zhou et al. [14] proposed the QLFR protocol, which follows reinforcement learning in any path routing schemes to prolong network lifetime with low delay [14]. The depth of information and residual energy plays a key role in making efficient routing decisions. Coutinho et al. [60] analyzed an OMUS protocol, an opportunistic routing scheme that supports faster data transmission with low energy consumption [60]. The forwarder candidate node can be chosen, or the suitability can be measured by considering the fitness function. The fitness function consists of residual energy, distance, link quality, and data transmission. Yan et al. [57, 58] proposed a reinforcement-learning-inspired localization protocol capable of minimizing error rate and noise. This protocol can find anchor nodes and estimate the location and range measurement.

8.5.2 Architecture Design

Numerous sensors are deployed on the underwater / seabed. The sink sensor sends information to the control or base center for processing. The control center or base

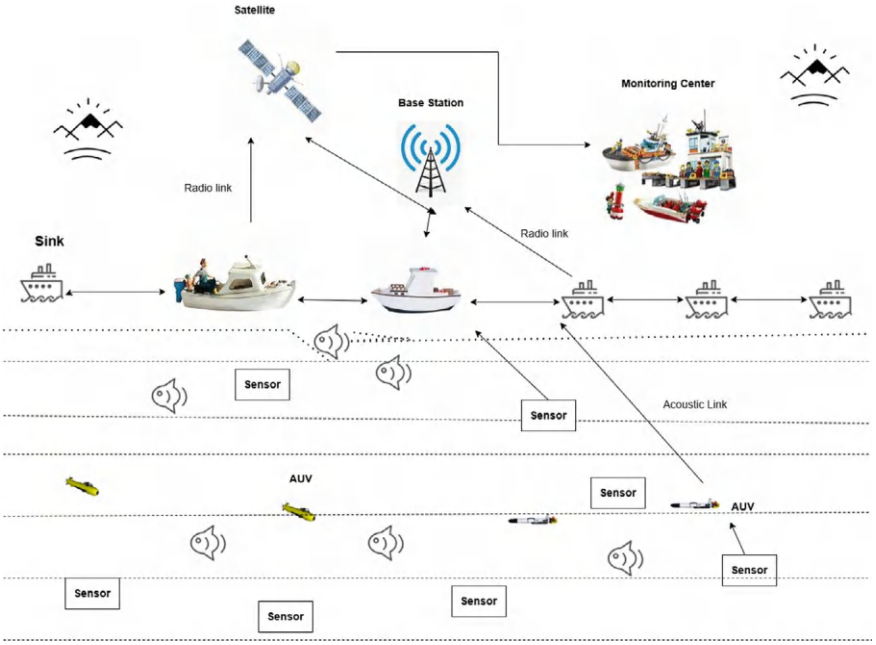


Fig. 8.4 Sea ad hoc network architecture

station can be deployed randomly or on the seabed. Multi-hop-based routing is popular and efficient in transmitting data from sender to receiver for message delivery. Figure 8.4 depicts regarding architecture and communication link of SANET.

8.5.3 Challenges

Wireless communication technology does not require hardware connections and considerable human resources to lay wires or cables. The communication technology is divided into two types such as short distance (example: UWB (ultra-wideband), Zig-Bee, NFC (Near field communication), Bluetooth and long distance (DSRC, GPRS/CDMA, digital radio, WiMAX) based wireless communication. Integration of various techniques to form a cross-layer platform is crucial [61]. Location tracking, monitoring, lifetime maximization [62], massive data collection [63], and task offloading are also challenging tasks nowadays. Designing an energy-efficient load-balanced routing scheme is challenging in an underground ad hoc network. The developed integrated architecture has massive issues regarding airâŁ“sea and air-VANET integrated applications. The issues handled by the MAC layer contain error control techniques, dynamic topology control, high data rate, channel utilization, low-latency-based data transmission, and multi-priority support.

8.5.4 Optimized MAC Protocol

High data transmission delay, low packet rate, less channel capacity, high error, and floating sensor node mobility are essential issues for UMWSNs. The TXOP is used for the relay sensor node to evaluate traffic demands for improved efficiency, and compression and SSIM are used to evaluate the underwater image quality. Efficient management of channel resources and reliable data communication in MAC is challenging. Acoustic communication improves coverage control and data rate compared to radio wave data communication. The MAC protocol design faced challenges such as considerable propagation delay, long guard time, channel efficiency, complexities with discovery messages, and high probe.

In this RL-inspired MAC system, the sensor uses a Q-table to search collision-free data transmissions without focusing on the beacons from relays and AP. An efficient compression algorithm is used to eliminate the network overhead, and an underwater image quality enhancement scheme is used for multimedia streaming. The RL âŁ“ MAC protocol [64] focused on the data transmission scheduling technique to eliminate dependency on TIM (Traffic Indication Map) and, the beacon. Further, data compression algorithms and image quality enhancement are essential to achieve efficiency and reliability. The network model consists of sensor nodes, relays, and AP. The relay sensor nodes are useful for message forwarding. Reinforcement learning algorithm used to improve channel usage. The Q learning is used to choose one slot from the frame. The Q table is used to choose the Q value for successful data transmission. The TIM and schedules are not required for periodic control information exchange and are used to eliminate computation overhead. The RL technique provides an efficient channel access technique. DL âŁ“ MAC protocol [65] reduces time length and faster data transmission.

8.5.5 Description Regarding Communication Channels

Today's underwater communication utilizes VHF (very high frequency) to identify shipping information which causes attenuation in the channel and affects the telecommunication system. This survey focussed on electromagnetic attenuation, acoustic communication technology, and optical attenuation. In electromagnetic attenuation, the radio frequency and electromagnetic waves are unable to propagate well underwater for the high conductivity nature of seawater. The conductivity and permittivity are used as channel parameters. It works on NLOS with limited communication range and low latency. In acoustic communication, the temperature, pressure, and salinity are considered channel parameters and suffer from a narrow frequency bandwidth. It supports NLOS and provides long data transmission, high latency, low bandwidth with low data rate, and harms marine life. In optical attenuation, scattering or turbidity and absorption are two causes of signal attenuation in SANET. It has low latency,

low cost, high bandwidth with limited range, suffers from environmental parameters, and uses LOS.

8.5.6 Routing Scheme

This section represents the routing taxonomy, techniques, and challenges. The SANET operates underwater areas to monitor mineral or oil extraction, marine environment, disaster prevention, assisted navigation, and collect information for various purposes. In SANET, routing protocols are prone to low bandwidth, high data propagation delay, low throughput, and low power consumption. The taxonomy of protocols is categorized as depth-based, cluster-based, cooperation-reliability-oriented, and AI-based routing. In the in-depth-based scheme, the forwarder sensor is chosen based on the hierarchical topology approach and the depth level of the sensor. This type of routing algorithm reduces unnecessary data transmission with low throughput. The cluster-based routing is categorized as layer-based and grid-oriented clustering.

The difference between the two types is that in layer based scheme, the seabed area is divided into multiple layers, and each layer has multiple cluster heads. In contrast, in a grid-based clustering scheme, the underwater area is divided into a grid, and each grid is considered as a cluster with one cluster head. In cooperation with reliability-oriented routing, the source forwards packets to the destination using a relay sensor node. The receiver receives multiple packets, one from the relay node and another from the source node. The receiver aggregates the data packets and collects the relevant information. If the link fails due to environmental or human-made factors, this technique ensures successful data transmission but cannot handle duplicate information transmission. The AI-inspired routing algorithms [66–70] enhance the QoS. In a firefly-based optimized routing scheme, this nature-inspired algorithm helps to choose a suitable forwarder sensor based on the highest priority value. The priority range is based on the node position, angle, distance of the neighboring sensor, and residual energy. The flower pollination-based routing scheme chooses the optimal path with less delay and the highest packet delivery. The Q learning-based routing scheme is energy-efficient but requires storing and updating its Q value in each step, so time complexity increases in the large network [55, 56, 64, 71].

Many efficient routing schemes are available, but it is challenging to select a cluster head, forwarder node, and relay node that will provide QoS. The routing schemes have many challenges, such as heavy energy consumption, unbalanced transmission, node deployment, information transmission unreliability, signal interface, dynamic topology handling, and time synchronization.

8.5.7 Security Concern

The attacks are categorized into passive and active attacks. Eavesdropping (which means "passive data collecting"), node malfunctioning, node destruction, traffic analysis, and node outage are coming under passive attack categories. The node compromise-based attack, repudiation-oriented attacks, packet-oriented, protocol-oriented, and DoS attacks are coming under active attack. This section describes various attacks on each layer. The physical layer suffers in node capture and jamming DoS attacks. In node capture, the intruder seizes control of the sensor node. Here, the hijacked sensor can create unlimited requests on behalf of intruders and give erroneous data to a genuine person. Jamming DoS is used to interrupt data communication and signals. The DSSS and FHSS approaches deal with noisy environments and handle multipath interference. The data link layer suffers from a denial of sleep, collision attack, jamming, exhaustive and unfairness attacks. The denial of sleep or sleep deprivation torture attack will create energy depletion. The collision attack requests CRC error and consumes low data transmission. In an exhaustive attack, the intruder keeps the data communication channel busy and drains the energy by deploying a malicious sensor node. Unfairness attacks, instead of blocking the entire communication system, degrades the system's performance. The network layer suffers from selective forwarding issues, misdirection attacks, greed and neglect attacks, gray hole and black hole attack, Sybil attacks, and homing attacks. In homing attack, the intruder sends dummy packets to the destination for anti-traffic analysis, which is used to hide the base station location or position from observers. Still, these dummy packets consume a large amount of energy. The transport layer is responsible for desynchronization and synchronization flooding attacks. Due to the heavy message flooding, it is challenging to determine or differentiate between fraudulent and authentic traffic. In a desynchronization attack, the intruder interrupts existing connections by transmitting fake packets that desynchronize the receiver end. Nowadays, many challenges such as reliability, low propagation delay, mobility of nodes, good communication range, low-cost hardware dependence, reliability of link, security, and privacy are crucial to achieving [72].

8.5.8 Challenges

The MAC protocol suffers various challenges, such as spectrum selection, reduced communication range, large delays in acoustic transmission, high bit error with low data rates, decreased throughput, and increased data retransmission probability. So, the efficient MAC scheme design is very important in terrestrial or underwater wireless communications. Underwater vehicle communication affects the aquatic animals. Therefore, it is a challenging task to automate the system in a way that benefits all. The surface buoys have direct data communication capacity with satellite systems and adopt the GPS for getting position information. The surface buoys provide self-

localization provision to the anchor nodes. The anchor node provides localization reference to the sensor nodes. The water sensors are static in nature and are concerned about their position privacy. Its position data can be estimated by cooperating with other anchor nodes, so security is essential to encrypt the position information. There are various future directions and challenges, such as efficient marine life management system, affordable low-cost sensor development, energy harvesting and conservation, large-scale communications, security, design of communication protocols, channel modeling, poor navigation, and positioning, non-destructive efficient underwater application testing, degraded image, lack of information leveraging tool, traffic control, and signal routing. Next, we will discuss various signal impairments in terms of noise, ambient, signal path, and frequency. The impairment causes intermodulation, crosstalk, suspended sediments, lack of resources, nodes, sinks, mobility, corrosions, erosions, high attenuation, fading, reverberation, long propagation delay, jitter, small channel capacity, large antenna size, distortion, and dispersion. Situational abnormality prediction using AI techniques is an important issue. This automated path optimization is vital regarding ice-covered waters and weather conditions [73]. The path planning is changed if efficiency or safety can be increased. Figure 8.4 discusses the underwater communication among ships, base stations, and satellites.

Underwater image contrast enhancement is essential to marine scientific outings, object detection, and underwater robotics. The underwater image contrast enhancement with minimum color loss is a challenging issue [74]. The MLLE technique [74] is used to enhance the contrast of the underwater image. The color balance approach is presented to Stabilize the color changes between channels b and a in the well-known color space such as CIELAB. The Jaffe-McGlamey underwater imaging model consists of forwarding, backward and direct scattering elements. The forward scattering sources blur and color deviations. The background scattering is the important aim for color deviations and contrast degradation. Both background and forward scattering rest on the medium data transmission map. Using a similar attenuation coefficient or data in the underwater area cannot imitate the detail of light selective attenuation underwater. So, the signal wavelength reliant on the attenuation reimbursement for the images is vital for finding a qualitative outcome using the UIQS image dataset.

8.5.9 Overview Regarding Autonomous Underwater Vehicles

In recent decades, underwater researchers have witnessed a noticeable advancement regarding underwater robotic automobiles for undersea exploitation and exploration. Risk analysis is a significant concern in AUV systems to achieve accuracy, hazard identification, consequence analysis, risk estimation, and improved uncertainty of risk factor analysis [75]. AUVs are useful for situational awareness in undersea or shelf ice, coastal water, and open water areas. Sometimes the thickness of ice can create underwater operational risk. It may damage propeller blades and antennas and

cause leakage. Ice thickness may create a collision, affect the data communication efficiency, and make AUV relocate difficultly. The cause of risk may be environmental, technical, or human factors. Various environmental factors such as surface wind, surface waves, gravitational tides, water depth, water density, salinity, inaccurate navigation, buoyant control failure, invisibility, integrity failure, poor communication, power degradation, collision, sea ice condition, sea bed condition, ice concentration, and thickness help to identify risk. The abnormal condition prediction can prevent accidents. The AUV subsystems are categorized as a propulsion system, navigation system, communication system, power system, environmental condition prediction system, and emergency system. The functionality of propulsion systems is buoyancy change or gives the propulsive force. The risk factor of this system may be thruster failure, rubber broken, pump failure, bladder leak, and actuator failure.

The functionality of navigation systems is dead reckoning navigation, which measures data's velocity, position, and altitude. The risk factors are depth sensor node failure and altimeter failure. The principle of communication systems is to control and transfer information or mission instruction. The risk factors are radio communication, signal transmission, and acoustic sensor node failure. The power system parameter gives electrical energy but sometimes causes energy depletion. The usefulness of environmental situation prediction systems is obstacle elimination and avoiding collision in the sea areas. The issues can be camera failure, light sources, and sonar suite defects or failure. To ensure safety, the emergency system deals with less risky route planning and jettisoning weight. The risk factors may be leak detection, node failure, and mission aborting message failure. The risk analysis can be done through three approaches such as quantitative, semi-quantitative, and qualitative. The safety layer technique and tree diagram come under quantitative analysis [76–78]. The Bow-tie process and failure mode analysis are semi-quantitative approaches [79]. The Markov chains, Bayesian network, Kaplan-Meier model, fault tree, and event tree techniques come under the quantitative method [80]. Various environmental parameters play an essential role in SANET providing QoS. The parameters [55] are temperature, conductivity, salinity, depth, pressure, hydrophone, turbidity, optical attenuation, optical backscatter, spectral irradiance, wave, and tide direction and elevation.

8.6 Underground Adhoc Network (UNDGAN)

The underground network is intended primarily for the subsurface region. The hazardous and complex surroundings of an underground mine may add to occupational injuries and fatalities. Deploying sensors have it possible to develop the safety of miners. The assessment has been considered and deliberated under the three predominant groupings: location tracking, environmental monitoring, body kinematics, and physiological monitoring. Additional sink sensor deployment is essential over the ground area network for information transmission from underground sensors to the gateway base station and safeguards resilience. Since nodes are underground,

recharge is problematic; the data transmission signal from the underground sensors gets attenuated.

8.6.1 Applications

UNDGAN helps monitor the environment in terms of gas, temperature, humidity, noise, dust, fire, airflow pressure level, and roof fall. Physiological sensors are used to monitor body kinematics in terms of fatigue monitoring, stress control, and musculoskeletal disorder monitoring [81]. Safety monitoring in underground (coal mining) Mohanadi coal field (MCL) is an interesting area. Machine learning can be implemented in routing for optimized data transmission from the source to the destination. The risk assessment of the worker is challenging because many workers are buried underground at the time of mining in the industry. The area's water level and fire monitoring is also crucial issue. Singh et al. [82] proposed an IoT-enabled smart HELMET, a safety device to protect the health condition of mine laborers [82]. The smart device architecture is categorized into three units: sensing, computing, and monitoring. The sensing unit deals with several sensors according to the problem statement. The computing unit deals with computing data analysis. The monitoring unit deals with visualizing and analyzing data in a real-time situation. It helps to trigger action and alert. To design a smart device, we must focus on a few steps. Firstly, collect data from the real-time field. Secondly, read environment values through Arduino. Thirdly, measure the real-time temperature, gases, humidity, fire, and silica dust. Fourthly, apply cutting-edge technology techniques to determine the nature of the work. Data is saved in the cloud for future health analysis of miner workers. If the working environment is unsafe, generate an alert to monitor and rescue the miner worker.

8.6.2 Architecture Design

Figure 8.5 depicts an overview of the underground network. The underground sensor nodes sense data and transmit it to the gateway node. The gateway node transmits data to the surface sink node. The surface sink transmits information to the cloud infrastructure, and the user can access relevant details per the requirement report. The cost of underground network architecture is high owing to the necessity for particular set-up for emerging underground networks, such as maintenance, price, and careful planning.

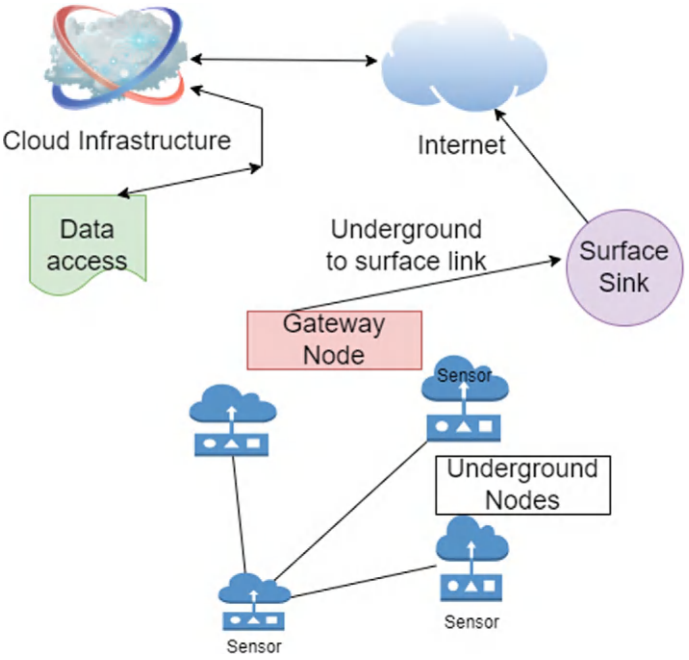


Fig. 8.5 Underground area architecture

8.6.3 Challenges

Wireless communication technology does not require hardware connections and considerable human resources to lay wires or cables. The communication technology is divided into two types such as short distance (example: UWB (ultra-wideband), Zig-Bee, NFC (Near field communication), Bluetooth and long distance (DSRC, GPRS/CDMA, digital radio, WiMAX) based wireless communication. Integration of various techniques to form a cross-layer platform is crucial [61]. Location tracking, monitoring, lifetime maximization [62], massive data collection [63], and task offloading are also challenging tasks nowadays. Designing an energy-efficient load-balanced routing scheme is challenging in an underground ad hoc network. The developed integrated architecture has massive issues regarding airâŁ“sea and air-VANET integrated applications. The issues handled by the MAC layer contain error control techniques, dynamic topology control, high data rate, channel utilization, low-latency-based data transmission, and multi-priority support.

8.7 Comparisons Among Various Ad Hoc Network

This review describes the various methodologies of intelligent transportation systems from a real-world point of view: solving real-time issues and the role of machine learning and cutting-edge technology. Wireless communication is essential for emergency response in the field of military application, wireless body area networks (WBANs), disaster relief, and vehicular ad hoc networks (VANET). From Table 8.2, we can see the comparison among the various ad hoc networks in terms of ,topology change, simulators, radio propagation, routing protocols, power consumption, number of sensor nodes, mobility, density, security, line of sight, and localization. UNDGAN is an emerging area discussed in section VI; cross-layer technology helps provide quality service.

8.8 Research Challenges in the Ad Hoc Network

Ad hoc networks have essential ingredients of self-healing, self-organization, and self-management for various operations in the ubiquitous environment. It needs cooperation to enable shared operations regarding various protocols.

Research and investigations are being done on low dimension and directional antenna design for antenna design. An efficient MAC layer must focus on a cooperative, energy-efficient, multichannel approach. Routing and security in data management also challenge research on how to build an efficient protocol that will mitigate or overcome various issues and outperform various QoS parameters. All the standards are focused on a particular aspect. There is an essential design standard that looks or focuses on the global picture.

Due to the different environments and functions, the low-cost sensor design, resource management, task scheduling, and signal variation control are challenging in various ad-hoc networks. For heterogeneous networks, task offloading, data collection, and transmission are crucial. The efforts should incorporate recent techniques for providing and integrating emerging wireless networks and cutting-edge technology. Optimized research is essential in improving the challenges mentioned in Table 8.1.

8.9 Conclusion and Future Directions

The ad hoc network supports a self-organizing structure that consists of accessible data communication that does not hang on infrastructure provision and control due to the robustness of the networking. This review focuses on fundamentals, architecture design, protocol analysis, future direction, and upcoming research challenges. Ad hoc networks have various research issues such as security, data routing, malicious sensor

Table 8.1 Various research issues regarding Ad hoc network

Sl. No.	Research issues	Challenges	Solutions
1	Antenna Design	Interference, channel capacity and multipath effect	Smart antenna design
2	Standardization	Heterogeneity, less harmony	IEEE, IETF
3	Energy management	Co-ordinated power management, low powered sensor node	Energy efficient data transmission and power control of data transmission
4	Mobility model	Realistic data simulation	Mobility protocol patterns generator
5	MAC layer	Line blocking, exposed terminal and hidden terminal problems, multichannel, coupling and scalability	Contention based and contention free solutions, QoS based MAC protocol for antenna design, Time slotting and random access
6	Routing routing loop, worm hole	black hole attack, New parameters considering: Fault tolerance, queuing, spectrum awareness, switching delay, route maintenance, delay and connection management	Hybrid, proactive, reactive and position based efficient routing, traffic aware multipath routing
7	Data management	Data discovery and data interoperability	Opportunistic information sharing and cross layer data exchange
8	Multicasting	Limited resource, Topology change in the network and node mobility	Hybrid, tree and mesh protocol
9	Cooperation	Lack of trust	Incentive and punishment based technique creates cooperation enforcement
10	Transport protocol	Due to congestion control and low throughput	Network layer data utilization
11	Security	Lack of secure infrastructure, authorization, authentication, integrity and trust establishment	Intrusion detection using cryptography âŁ“ blockchain and digital forensic
12	Addressing	DoS attack, no data addressing server	State full and state less approach
13	Resource discovery	Heterogeneity, limited battery, resource sharing	Cross layer design, D2D communication
14	Power control	Signal variation and mobility	AI technique

(continued)

Table 8.1 (continued)

Sl. No.	Research issues	Challenges	Solutions
15	Scheduling	Scheduling criteria, network partitioning, node mobility and failures	Centralized, distributed, hierarchical, optimized scheduling
16	Economic model	Ensure cooperation, utility assignment, fairness utilization of the task or resources, payment mechanisms negotiation	
17	Task offloading	Distance and heterogeneity	Virtual machine migration, machine learning technique (reinforcement learning) [83], remote method call and invocation
18	Energy management	Energy aware and limited resources	Node cooperation and network coding
19	Spectrum sensing	Channel condition monitoring i.e. noise, multipath effects and interference, spectrum awareness	Cooperative and non cooperative mechanism
20	Spectrum decision and sharing	Power control, coordination mechanism and energy management	Game theory concept and spectrum sharing
21	Channel control mechanism	Jamming attack and channel control contention	ISM and UWM band, dedicated licensed spectrum
22	QoS	Service provisioning, link failure, delay, connection management, congestion handling and signal jamming	AI technique can be implemented
23	Simulation and testbeds	Cost of physical devices (hardware) and network complexity	Design of simulator

Table 8.2 Comparisons of ad-hoc network

Contents	MANETs	VANETs	FANETs	SANETs
Used area	Sensor nodes move on specific land	VANET nodes move the city, rural area and highways	FANET nodes fly in the sky	SANET is useful for the exchange of data between soldiers and headquarters when they are in underwater area or ocean
Range of communication	Up to 100 m	Up to 500 m	U2U: 5 GHz band and U2G: 2.4 GHz band	1–10 km (Acoustic), 10 m (RF), 100 m (optical), 1.5 m–5 m (coverage range), 2.4 Kbps (data rate), acoustic, RF and optical media is used for deep sea data transmission, water-terrestrial message transmission and short range data transmission
Bandwidth of network	Up to 100 kps	Up to 1000 kps		Limited bandwidth, 10–100 kbp (acoustic), 1-10 Mbps(RF), 10–100 Gbps (optical)
MAC Protocols	IEEE 802.11 b	IEEE 802.11p	IEEE 802.11 n/ac	R-MAC, DL âŁ“ MAC [65]
Propagation loss model	Friis	Two ray ground	In the air and Thorp's propagation, channel model: Urick model, loss rate: 0.1–4 dB/km (acoustic), 3.5–5 dB/m (RF)	0.39–11 dB/m (optical) [113]
Frequency band	30 MGHzâŁ“5 GHz	5.9 GHz	UAV-2-UAV-5 GHz and UAV-2-G-2.4 GHz	10 Hz to 1 MHz
Position acquisition	Using ultrasonic	Using GPS, RADAR	GPS	Acoustic

(continued)

Table 8.2 (continued)

Contents	MANETs	VANETs	FANETs	SANETs
Topology status	Topology changes low	Topology changes fast	Topology altering is rapid and very fast	Topology changes medium, supports 1D, 2D and M-D topology [113]
Characteristics	Dynamic network topologies, low bandwidth, multi-hop, scalability, limited battery power, infrastructure-less, self organizing and distributed, temporary deployable	High mobility, High dynamic topology, communication environment, high delay constraints, safety	Node mobility, access to LoS (Line of Sight), low node density with a large distance among the UAVS, GPS provides the geospatial localization.	Safety communications, satellite communications, channel modeling, reliability, range extension
Radio propagation	The radio propagation typical close to the ground	The radio propagation typical close to the ground	The radio propagation typical, not close to the ground	The radio propagation typical close to the ground
Application	Military battlefield, business associates, personal area network	Safety, driver assistance, entertainment applications, traffic management	Military operation (works without human help)	Weather forecasting, water movements, predicts tsunamis, disaster managements
Power consumes	Power consumes low	Power consumes high	Power consumption is high for mini-UAV/small UAV not needed	Power consumes high
Power computational	Limited	High	High	High
Mobility	Node/device mobility is low	Node/device mobility is high	Node/device mobility is very high	Node/device mobility is medium
Mobility model	The mobility model is random	The mobility model is regular	The mobility model is regular under the condition and categorized as pure randomized, time dependent, path planning, topology control	Group based mobility model [114] and The mobility model is random, RWP

(continued)

Table 8.2 (continued)

Contents	MANETs	VANETs	FANETs	SANETs
Line of sight	Not available for all cases	Available in some cases	Available in most cases	Available in some cases
Security	Data confidentiality, data integrity and data reliability is high	Data confidentiality, data integrity is high and data reliability is high	Data confidentiality, data integrity is high and data reliability is high	Data confidentiality, data integrity is high and data reliability is high
Challenges and Issues	Routing, scalability, limited bandwidth and energy, security.	Delay, reliability, channel congestion, Quality of services	Synchronization among UAVs swarms, Sufficient low latency and bandwidth availability, mobility , geographical constraints	Security, D2D time synchronization ,Scalability, Re-synchronization , Handles unexpected disconnections in the routing protocols, Reliability, Quality of Service (QoS): packet error, delays and throughput
Localization	GPS	DGPS, GPS, AGPS	IMU, GPS, AGPS and DGPS	GPS, DGPS
Density	Low	High	Very low	High
Number of nodes	High	High	Low	Low
Update topology	Not often	Often	Often	Not often
Routing protocol	AGEN-AODV, AASR, RETRUST, t2fcatr	IRNN-LEACH, B-IOMV, DTN, NOTRINO, VMBP	Q-FANET, DroneNetx, CLDVPS	LOCDYN, OMUS, QLFR, QLR, SEAL, EELA

Table 8.3 Roadmap to autonomous driving

Level	Autonomous status	Automation criteria	Driver strategies	Implementation status
0	No automation	Driver is fully engaged	All the time driver control	Done
1	Driver assistance	Automation of individual functions	Driver can feet off or hand off	Done
2	Partial automation	Automation of multiple functions	Driver can both feet off and hands off but eyes must be on road	Testing phase
3	Conditional automation	Automation of multiple functions, driver responds to a request	Driver can both feet off, hands off and eyes off but must be able to resume control quickly	2020 above
4	High automation	Automated in certain conditions	Driver not expected to monitor road driver has no responsibility during automated mode	2025 above
5	Full automation	Situation independent automated driving	Driver has no responsibility during driving	2030 above

node detection, link establishment, resource provision, connection policy, mobility, integrated architecture, less congestion (congestion detection and prevention), safety, comfort, and excellence of road experience or QoS (quality of services).

Ad hoc networks can be applied in various scenarios, such as weather forecasting, marine satellite data communications, disaster relief, battlefield communications, and emergency rescue. So this system is essential for upcoming wireless communication. Here, we analyzed the fundamentals of ad hoc systems, applications, various performance parameters, pros, characteristics, cons, and some recent protocols from past research.

New cutting-edge technologies (cloud-fog-edge computing, blockchain, quantum computing, SDN, green computing, federated learning, IoT, Machine learning, deep learning, 5G, 6G, Big data, game theory, and graph theory) are presently being used or taken in many domains for their robust prospective to enhance performance metrics.

This review could assist as a reference model so beginners can understand the fundamentals of ad hoc networks, simulation tools, current protocols of various issues, architecture design, research status, comparisons, and future direction.

References

1. Rahman, M.A., Rahim, M.A., Rahman, M.M., Moustafa, N., Razzak, I., Ahmad, T., Patwary, M.N.: A Secure and intelligent framework for vehicle health monitoring exploiting big-data analytics. IEEE Trans. Intell. Transp. Syst
2. Memon, I., Hasan, M.K., Shaikh, R.A., Nebhen, J., Bakar, K.A.A., Hossain, E., Tunio, M.H.: Energy-efficient fuzzy management system for internet of things connected vehicular ad hoc networks. Electronics **10**(9), 1068 (2021)
3. Yao, L., Wang, J., Wang, X., Chen, A., Wang, Y.: V2X routing in a VANET based on the hidden Markov model. IEEE Trans. Intell. Transp. Syst. **19**(3), 889–899 (2017)
4. Luo, L., Sheng, L., Yu, H., Sun, G.: Intersection-based V2X routing via reinforcement learning in vehicular ad hoc networks. IEEE Trans. Intell. Transp. Syst. (2021)
5. Tang, H., Wu, H., Qu, G., Li, R.: Double deep Q-network based dynamic framing offloading in vehicular edge computing. IEEE Trans. Netw. Sci. Eng. (2022)
6. Behura, A.: Optimized data transmission scheme based on proper channel coordination used in vehicular ad hoc networks. Int. J. Inf. Technol. 1–10 (2021)
7. Al-Absi, M.A., Al-Absi, A.A., Sain, M., Lee, H.: Moving ad hoc networks—A comparative study. Sustainability **13**(11), 6187 (2021)
8. Nguyen, V., Kim, O.T.T., Pham, C., Oo, T.Z., Tran, N.H., Hong, C.S., Huh, E.N.: A survey on adaptive multi-channel MAC protocols in VANETs using Markov models. IEEE Access **6**, 16493–16514 (2018)
9. Wang, Q., Leng, S., Fu, H., Zhang, Y.: An IEEE 802.11 p-based multichannel MAC scheme with channel coordination for vehicular ad hoc networks. IEEE Trans. Intell. Transp. Syst. **13**(2), 449–458 (2011)
10. Yao, Y., Zhang, K., Zhou, X.: A flexible multi-channel coordination MAC protocol for vehicular ad hoc networks. IEEE Commun. Lett. **21**(6), 1305–1308 (2017)
11. Behura, A., Srinivas, M., Kabat, M.R.: Giraffe kicking optimization algorithm provides efficient routing mechanism in the field of vehicular ad hoc networks. J. Ambient Intell. Hum. Comput. 1–20 (2022)
12. Lakew, D.S., Sa'ad, U., Dao, N.N., Na, W., Cho, S.: Routing in flying ad hoc networks: a comprehensive survey. IEEE Commun. Surveys Tutor. **22**(2), 1071–1120 (2020)
13. Bekmezci, I., Sahingoz, O.K., Temel, Ş: Flying ad-hoc networks (FANETs): a survey. Ad Hoc Netw. **11**(3), 1254–1270 (2013)
14. Zhou, Y., Cao, T., Xiang, W.: Anypath routing protocol design via Q-learning for underwater sensor networks. IEEE Internet Things J. **8**(10), 8173–8190 (2020)
15. Xiao, L., Jiang, D., Chen, Y., Su, W., Tang, Y.: Reinforcement-learning-based relay mobility and power allocation for underwater sensor networks against jamming. IEEE J. Oceanic Eng. **45**(3), 1148–1156 (2019)
16. Ojha, T., Misra, S., Obaidat, M.S.: SEAL: self-adaptive AUV-based localization for sparsely deployed Underwater Sensor Networks. Comput. Commun. **154**, 204–215 (2020)
17. Yuan, Y., Liang, C., Kaneko, M., Chen, X., Hogrefe, D.: Topology control for energy-efficient localization in mobile underwater sensor networks using Stackelberg game. IEEE Trans. Veh. Technol. **68**(2), 1487–1500 (2018)
18. Jung, J.Y., Choi, H.H., Lee, J.R.: Survey of bio-inspired resource allocation algorithms and MAC protocol design based on a bio-inspired algorithm for mobile ad hoc networks. IEEE Commun. Mag. **56**(1), 119–127 (2018)
19. Devika, B., Sudha, P.N.: Power optimization in MANET using topology management. Eng. Sci. Technol. Int. J. **23**(3), 565–575 (2020)
20. Muchtar, F., Abdullah, A.H., Hassan, S., Masud, F.: Energy conservation strategies in host centric networking based MANET: a review. J. Netw. Comput. Appl. **111**, 77–98 (2018)
21. Doss, S., Nayyar, A., Suseendran, G., Tanwar, S., Khanna, A., Thong, P.H.: APD-JFAD: accurate prevention and detection of jelly fish attack in MANET. IEEE Access **6**, 56954–56965 (2018)

22. Srilakshmi, U., Veeraiah, N., Alotaibi, Y., Alghamdi, S.A., Khalaf, O.I., Subbayamma, B.V.: An improved hybrid secure multipath routing protocol for MANET. IEEE Access **9**, 163043–163053 (2021)
23. Nadeem, A., Howarth, M.P.: A survey of MANET intrusion detection & prevention approaches for network layer attacks. IEEE Commun. Surveys & Tutor. **15**(4), 2027–2045 (2013)
24. Farkhana, M., Hanan, A.A., Suhaidi, H., Tajudin, K.A., Zuhairi, Z.K.: Energy conservation of content routing through wireless broadcast control in NDN based MANET: a review. J. Netw. Comput. Appl. **131**, 109–132 (2019)
25. Salunke, K., Ragavendran, U.: Shield techniques for application layer DDoS attack in MANET: a methodological review. Wirel. Pers. Commun. **120**(4), 2773–2799 (2021)
26. Kurode, E., Vora, N., Patil, S., Attar, V.: MANET routing protocols with emphasis on zone routing protocol—An overview. In: 2021 IEEE Region 10 Symposium (TENSYMP), pp. 1–6. IEEE (2021)
27. Fasunlade, O., Zhou, S., Sanders, D.: Comprehensive review of collaborative network attacks in MANET. In 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1542–1545. IEEE (2020)
28. Nabati, M., Maadani, M., Pourmina, M.A.: AGEN-AODV: an intelligent energy-aware routing protocol for heterogeneous mobile ad-hoc networks. Mob. Netw. Appl. **27**(2), 576–587 (2022)
29. Quy, V.K., Nam, V.H., Linh, D.M., Ngoc, L.A. Routing algorithms for MANET-IoT networks: a comprehensive survey. Wirel. Pers. Commun. 1–25 (2022)
30. Mohsin, A.H.: Optimize routing protocol overheads in MANETs: challenges and solutions: a review paper. Wirel. Pers. Commun. 1–40 (2022)
31. Jain, R. Ant colony inspired energy efficient OLSR (AC-OLSR) routing protocol in MANETS. Wirel. Pers. Commun. 1–14
32. Brindha, N.V., Meenakshi, V.S. A secured optimised AOMDV routing protocol in MANET using lightweight continuous multimodal biometric authentication. J. Ambient Intell. Human. Comput. 1–17 (2022)
33. Ghaffari, A.: Real-time routing algorithm for mobile ad hoc networks using reinforcement learning and heuristic algorithms. Wirel. Netw. **23**(3), 703–714 (2017)
34. Kaviani, S., Ryu, B., Ahmed, E., Larson, K.A., Le, A., Yahja, A., Kim, J.H. Robust and scalable routing with multi-agent deep reinforcement learning for MANETs (2021). arXiv:2101.03273
35. Bajracharya, R., Shrestha, R., Hassan, S.A., Konstantin, K., Jung, H. Dynamic pricing for intelligent transportation system in the 6G unlicensed band. IEEE Trans. Intell. Transport. Syst. (2021)
36. Hota, L., Nayak, B.P., Kumar, A., Ali, G.M.N., Chong, P.H.J.: An analysis on contemporary MAC layer protocols in vehicular networks: state-of-the-art and future directions. Future Internet **13**(11), 287 (2021)
37. Tang, F., Mao, B., Kato, N., Gui, G.: Comprehensive survey on machine learning in vehicular network: technology, applications and challenges. IEEE Commun. Surveys Tutor. (2021)
38. Nabati, M., Maadani, M., Pourmina, M.A. AGEN-AODV: an intelligent energy-aware routing protocol for heterogeneous mobile ad-hoc networks. Mob. Netw. Appl. 1–12 (2021)
39. Mozaffari, S., Al-Jarrah, O.Y., Dianati, M., Jennings, P., Mouzakitis, A.: Deep learning-based vehicle behavior prediction for autonomous driving applications: a review. IEEE Trans. Intell. Transp. Syst. **23**(1), 33–47 (2020)
40. Ji, J., Khajepour, A., Melek, W.W., Huang, Y.: Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. IEEE Trans. Veh. Technol. **66**(2), 952–964 (2016)
41. Liu, H., Zhang, S., Zhang, P., Zhou, X., Shao, X., Pu, G., Zhang, Y.: Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing. IEEE Trans. Veh. Technol. **70**(6), 6073–6084 (2021)
42. Muhammad, K., Ullah, A., Lloret, J., Del Ser, J., de Albuquerque, V.H.C.: Deep learning for safe autonomous driving: Current challenges and future directions. IEEE Trans. Intell. Transp. Syst. **22**(7), 4316–4336 (2020)

43. Joo, H., Ahmed, S.H., Lim, Y.: Traffic signal control for smart cities using reinforcement learning. Comput. Commun. **154**, 324–330 (2020)
44. Erdelj, M., Król, M., Natalizio, E.: Wireless sensor networks and multi-UAV systems for natural disaster management. Comput. Netw. **124**, 72–86 (2017)
45. Lin, C., Han, G., Zhang, T., Shah, S.B.H., & Peng, Y. Smart underwater pollution detection based on graph-based multi-agent reinforcement learning towards AUV-based network ITS. IEEE Trans. Intell. Transp. Syst. (2022)
46. Bhoi, S.K., Jena, K.K., Panda, S.K., Long, H.V., Kumar, R., Subbulakshmi, P., Jebreen, H.B.: An Internet of Things assisted Unmanned Aerial Vehicle based artificial intelligence model for rice pest detection. Microprocess. Microsyst. **80**, 103607 (2021)
47. Mukherjee, A., Misra, S., Chandra, V.S.P., Obaidat, M.S.: Resource-optimized multiarmed bandit-based offload path selection in edge UAV swarms. IEEE Internet Things J. **6**(3), 4889–4896 (2018)
48. Menouar, H., Guvenc, I., Akkaya, K., Uluagac, A.S., Kadri, A., Tuncer, A.: UAV-enabled intelligent transportation systems for the smart city: applications and challenges. IEEE Commun. Mag. **55**(3), 22–28 (2017)
49. Mohamed, N., Al-Jaroodi, J., Jawhar, I., Idries, A., Mohammed, F.: Unmanned aerial vehicles applications in future smart cities. Technol. Forecast. Soc. Chang. **153**, 119293 (2020)
50. Xu, Z., Luo, M., Vijayakumar, P., Peng, C., Wang, L.: Efficient certificateless designated verifier proxy signature scheme using UAV network for sustainable smart city. Sustain. Cities Soc. **80**, 103771 (2022)
51. Liu, J., Wang, Q., He, C., Jaffrès-Runser, K., Xu, Y., Li, Z., Xu, Y.: QMR: Q-learning based multi-objective optimization routing protocol for flying ad hoc networks. Comput. Commun. **150**, 304–316 (2020)
52. Yang, Q., Jang, S.J., Yoo, S.J.: Q-learning-based fuzzy logic for multi-objective routing algorithm in flying ad hoc networks. Wirel. Pers. Commun. **113**(1), 115–138 (2020)
53. Arafat, M.Y., Poudel, S., Moh, S.: Medium access control protocols for flying ad hoc networks: a review. IEEE Sens. J. **21**(4), 4097–4121 (2020)
54. Tsao, K.Y., Girdler, T., Vassilakis, V.G. A survey of cyber security threats and solutions for UAV communications and flying ad-hoc networks. Ad Hoc Netw. 102894 (2022)
55. Jahanbakht, M., Xiang, W., Hanzo, L., Azghadi, M.R.: Internet of underwater things and big marine data analytics—A comprehensive survey. IEEE Commun. Surveys Tutor. **23**(2), 904–956 (2021)
56. Coutinho, R.W., Boukerche, A., Vieira, L.F., Loureiro, A.A.: A joint anypath routing and duty-cycling model for sustainable underwater sensor networks. IEEE Trans. Sustain. Comput. **4**(4), 314–325 (2018)
57. Yan, J., Meng, Y., Yang, X., Luo, X., Guan, X.: Privacy-preserving localization for underwater sensor networks via deep reinforcement learning. IEEE Trans. Inf. Forensics Secur. **16**, 1880–1895 (2020)
58. Yan, J., Gong, Y., Chen, C., Luo, X., Guan, X.: AUV-aided localization for internet of underwater things: a reinforcement-learning-based method. IEEE Internet Things J. **7**(10), 9728–9746 (2020)
59. Soares, C., Gomes, J., Ferreira, B.Q., Costeira, J.P.: LocDyn: robust distributed localization for mobile underwater networks. IEEE J. Oceanic Eng. **42**(4), 1063–1074 (2017)
60. Coutinho, R.W., Boukerche, A.: OMUS: efficient opportunistic routing in multi-modal underwater sensor networks. IEEE Trans. Wireless Commun. **20**(9), 5642–5655 (2021)
61. Singh, P., Singh, R.P., Singh, Y.: An optimal energy-throughput efficient cross-layer solution using naked mole rat algorithm for wireless underground sensor networks. Mater. Today: Proc. **48**, 1076–1083 (2022)
62. Tam, N.T., Binh, H.T.T., Dung, D.A., Lan, P.N., Yuan, B., Yao, X.: A hybrid clustering and evolutionary approach for wireless underground sensor network lifetime maximization. Inf. Sci. **504**, 372–393 (2019)
63. Liu, G.: Data collection in MI-assisted wireless powered underground sensor networks: directions, recent advances, and challenges. IEEE Commun. Mag. **59**(4), 132–138 (2021)

64. Gazi, F., Ahmed, N., Misra, S., Wei, W.: Reinforcement learning-based MAC protocol for underwater multimedia sensor networks. ACM Trans. Sens. Netw. (TOSN) (2022)
65. Geng, X., Zheng, Y.R.: Exploiting propagation delay in underwater acoustic communication networks via deep reinforcement learning. IEEE Trans. Neural Netw. Learn. Syst. (2022)
66. Faheem, M., Tuna, G., Gungor, V.C.: QERP: quality-of-service (QoS) aware evolutionary routing protocol for underwater wireless sensor networks. IEEE Syst. J. **12**(3), 2066–2073 (2017)
67. Luo, J., Chen, Y., Wu, M., Yang, Y.: A survey of routing protocols for underwater wireless sensor networks. IEEE Commun. Surveys Tutor. **23**(1), 137–160 (2021)
68. Subramani, N., Mohan, P., Alotaibi, Y., Alghamdi, S., Khalaf, O.I.: An efficient metaheuristic-based clustering with routing protocol for underwater wireless sensor networks. Sensors **22**(2), 415 (2022)
69. Faheem, M., Ngadi, M.A., Gungor, V.C.: Energy efficient multi-objective evolutionary routing scheme for reliable data gathering in Internet of underwater acoustic sensor networks. Ad Hoc Netw. **93**, 101912 (2019)
70. Sivakumar, V., Rekha, D.: A QoS-aware energy-efficient memetic flower pollination routing protocol for underwater acoustic sensor network. Concurr. Comput.: Practice Exp. **32**(4), e5166 (2020)
71. Kumar, N., Chaudhry, R., Kaiwartya, O., Kumar, N., Ahmed, S.H.: Green computing in software defined social internet of vehicles. IEEE Trans. Intell. Transp. Syst. **22**(6), 3644–3653 (2020)
72. Ahmad, I., Rahman, T., Zeb, A., Khan, I., Ullah, I., Hamam, H., Cheikhrouhou, O.: Analysis of security attacks and taxonomy in underwater wireless sensor networks. Wirel. Commun. Mobile Comput. (2021)
73. Thombre, S., Zhao, Z., Ramm-Schmidt, H., García, J.M.V., Malkamäki, T., Nikolskiy, S., Hammarberg, T., Nuortie, H., Bhuiyan M.Z., Särkkä, S., Lehtola, V.V. Sensors and ai techniques for situational awareness in autonomous ships: a review. IEEE Trans. Intell. Transp. Syst. (2020)
74. Zhang, W., Zhuang, P., Sun, H.H., Li, G., Kwong, S., Li, C.: Underwater image enhancement via minimal color loss and locally adaptive contrast enhancement. IEEE Trans. Image Process. **31**, 3997–4010 (2022)
75. Chen, X., Bose, N., Brito, M., Khan, F., Thanyamanta, B., Zou, T.: A review of risk analysis research for the operations of autonomous underwater vehicles. Reliab. Eng. Syst. Saf. **216**, 108011 (2021)
76. Thieme, C.A., Utne, I.B.: Safety performance monitoring of autonomous marine systems. Reliab. Engi. Syst. Safety **159**, 264–275 (2017)
77. Du, L., Goerlandt, F., Kujala, P.: Review and analysis of methods for assessing maritime waterway risk based on non-accident critical events detected from AIS data. Reliab. Eng. Syst. Safety **200**, 106933 (2020)
78. Lim, G.J., Cho, J., Bora, S., Biobaku, T., Parsaei, H.: Models and computational algorithms for maritime risk analysis: a review. Ann. Oper. Res. **271**(2), 765–786 (2018)
79. Yu, M., Venkidasalapathy, J.A., Shen, Y., Quddus, N., Mannan, S.M. Bow-tie analysis of underwater robots in offshore oil and gas operations. In: Offshore Technology Conference. OnePetro (2017)
80. Xiang, X., Yu, C., Zhang, Q.: On intelligent risk analysis and critical decision of underwater robotic vehicle. Ocean Eng. **140**, 453–465 (2017)
81. Sadeghi, S., Soltanmohammadlou, N., Nasirzadeh, F.: Applications of wireless sensor networks to improve occupational safety and health in underground mines. J. Safety Res. (2022)
82. Singh, N., Gunjan, V.K., Chaudhary, G., Kaluri, R., Victor, N., Lakshmanna, K.: IoT enabled HELMET to safeguard the health of mine workers. Comput. Commun. **193**, 1–9 (2022)
83. Yang, C., Liu, B., Li, H., Li, B., Xie, K., Xie, S.: Learning based channel allocation and task offloading in temporary UAV-assisted vehicular edge computing networks. IEEE Trans. Veh. Technol. (2022)
84. Rajpoot, V., Garg, L., Alam, M.Z., Parashar, V., Tapashetti, P., Arjariya, T.: Analysis of machine learning based LEACH robust routing in the Edge Computing systems. Comput. Electr. Eng. 107574 (2021)

85. Goswami, P., Mukherjee, A., Hazra, R., Yang, L., Ghosh, U., Qi, Y., Wang, H.: AI based energy efficient routing protocol for intelligent transportation system. IEEE Trans. Intell. Transp. Syst. (2021)
86. Sajedi, S.N., Maadani, M., Moghadam, M.N. F-LEACH: a fuzzy-based data aggregation scheme for healthcare IoT systems. J. Supercomput. 1–18 (2021)
87. Yassien, M.B., Aljawarneh, S.A., Eyadat, M., Eaydat, E.: Routing protocol for low power and lossy network–load balancing time-based. Int. J. Mach. Learni. Cybern. 1–14 (2021)
88. da Costa, L.A.L., Kunst, R., de Freitas, E.P.: Q-FANET: improved Q-learning based routing protocol for FANETs. Comput. Netw. **198**, 108379 (2021)
89. Gupta, R., Tanwar, S., Kumar, N.: B-IoMV: Blockchain-based onion routing protocol for D2D communication in an IoMV environment beyond 5G. Veh. Commun. 100401 (2021)
90. Pathak, N., Misra, S., Mukherjee, A., Roy, A., Zomaya, A.Y.: UAV virtualization for enabling heterogeneous and persistent UAV-as-a-Service. IEEE Trans. Veh. Technol. **69**(6), 6731–6738 (2020)
91. Park, S.Y., Shin, C.S., Jeong, D., Lee, H.: DroneNetX: network reconstruction through connectivity probing and relay deployment by multiple UAVs in ad hoc networks. IEEE Trans. Veh. Technol. **67**(11), 11192–11207 (2018)
92. Yazici, A., Kirlik, G., Parlaktuna, O., Sipahioglu, A.: A dynamic path planning approach for multirobot sensor-based coverage considering energy constraints. IEEE Trans. Cybern. **44**(3), 305–314 (2013)
93. Haghighi, M.S., Aziminejad, Z.: Highly anonymous mobility-tolerant location-based onion routing for VANETs. IEEE Internet Things J. **7**(4), 2582–2590 (2019)
94. Liu, W., Yu, M.: AASR: authenticated anonymous secure routing for MANETs in adversarial environments. IEEE Trans. Veh. Technol. **63**(9), 4585–4593 (2014)
95. Sakai, K., Sun, M.T., Ku, W.S., Wu, J., Alanazi, F.S.: Performance and security analyses of onion-based anonymous routing for delay tolerant networks. IEEE Trans. Mob. Comput. **16**(12), 3473–3487 (2017)
96. Darabkh, K.A., Al-rawashdeh, W.S., Hawa, M.: MT-CHR: a modifed threshold-based cluster head replacement protocol for wireless sensor networks. Comput. Electr. Eng. **72**, 926–938 (2018). https://doi.org/10.1016/j.compeleceng.2018.01.032
97. Panag, T.S., Dhillon, J.S.: Dual head static clustering algorithm for wireless sensor networks. AEU Int. J. Electron. Commun. **88**, 148–156 (2018). https://doi.org/10.1016/j.aeue.2018.03.019
98. Rawat, P., Chauhan, S.: Probability based cluster routing protocol for wireless sensor network. J. Ambient. Intell. Humaniz. Comput. (2020). https://doi.org/10.1007/s12652-020-02307-1
99. Omar, H.A., Zhuang, W., Li, L.: Gateway placement and packet routing for multihop in-vehicle internet access. IEEE Trans. Emerg. Top. Comput. **3**(3), 335–351 (2015)
100. Guo, C., Li, D., Zhang, G., Zhai, M.: Real-time path planning in urban area via VANET-assisted traffic information sharing. IEEE Trans. Veh. Technol. **67**(7), 5635–5649 (2018)
101. Bi, Y., Shan, H., Shen, X.S., Wang, N., Zhao, H.: A multi-hop broadcast protocol for emergency message dissemination in urban vehicular ad hoc networks. IEEE Trans. Intell. Transp. Syst. **17**(3), 736–750 (2015)
102. Gayathri, N.B., Thumbur, G., Reddy, P.V., Rahman, M.Z.U.: Efficient pairing-free certificateless authentication scheme with batch verification for vehicular ad-hoc networks. IEEE Access **6**, 31808–31819 (2018)
103. Nagendranth, M.V.S.S., Khanna, M.R., Krishnaraj, N., Sikkandar, M.Y., Aboamer, M.A., Surendran, R.: Type II fuzzy-based clustering with improved ant colony optimization-based routing (t2fcatr) protocol for secured data transmission in manet. J. Supercomput. 1–19 (2022)
104. Ahmad, F., Kurugollu, F., Kerrache, C.A., Sezer, S., Liu, L.: NOTRINO: a NOvel hybrid TRust management scheme for INternet-Of-vehicles. IEEE Trans. Veh. Technol. (2021)
105. He, D., Chen, C., Chan, S., Bu, J., Vasilakos, A.V.: ReTrust: attack-resistant and lightweight trust management for medical sensor networks. IEEE Trans. Inf Technol. Biomed. **16**(4), 623–632 (2012)

106. Li, Y., Yang, B., Wu, H., Han, Q., Chen, C., Guan, X.: Joint offloading decision and resource allocation for vehicular fog-edge computing networks: a contract-stackelberg approach. IEEE Internet of Things J. (2022)
107. Mkiramweni, M.E., Yang, C., Li, J., Zhang, W.: A survey of game theory in unmanned aerial vehicles communications. IEEE Commun. Surveys Tutor. **21**(4), 3386–3416 (2019)
108. Wang, H., Li, Y., Qian, J.: Self-adaptive resource allocation in underwater acoustic interference channel: a reinforcement learning approach. IEEE Internet Things J. **7**(4), 2816–2827 (2019)
109. Wengle, E., Potter, J., Dong, H.: Underwater ad-hoc networks: a review (2021)
110. Rezwan, S., Choi, W.: A survey on applications of reinforcement learning in flying ad-hoc networks. Electronics **10**(4), 449 (2021)
111. Krummenacher, G., Ong, C.S., Koller, S., Kobayashi, S., Buhmann, J.M.: Wheel defect detection with machine learning. IEEE Trans. Intell. Transp. Syst. **19**(4), 1176–1187 (2017)
112. Hu, H.X., Tang, B., Zhang, Y., Wang, W.: Vehicular ad hoc network representation learning for recommendations in Internet of Things. IEEE Trans. Ind. Inf. **16**(4), 2583–2591 (2019)
113. Bello, O., Zeadally, S.: Internet of underwater things communication: architecture, technologies, research challenges and future opportunities. Ad Hoc Netw. **135**, 102933 (2022)
114. Srivastava, A., Prakash, J.: Future FANET with application and enabling techniques: anatomization and sustainability issues. Comput. Sci. Rev. **39**, 100359 (2021)

Chapter 9
Application of Software-Defined Networks



U. Prabu, N. Malarvizhi, V. Geetha, and Anand Nayyar

Abstract Software-Defined Networks (SDN) have developed as a transformational paradigm in modern computer networking by separating the control plane from the data plane, thus enabling centralized programmability, automation, and simplified management of networks. This chapter presents an overview of SDN, beginning with its historical evolution and the fundamental principles that distinguish it from traditional networking architectures. It discusses the SDN architecture in detail, including its key components such as the application, control, and data planes, along with the role of northbound and southbound interfaces. The chapter then explores a wide range of SDN applications, including Traffic Engineering and Load Balancing, Software-Defined Wide Area Networking (SD-WAN), integration with Network Function Virtualization (NFV), Data Centers, Internet of Things (IoT), Cloud Computing, and Telecommunications. The advantages and difficulties of deploying SDN are vitally analyzed, followed by real-world case studies that highlight its practical adoption by leading technology companies.

Keywords Software-Defined Networks (SDN) · Control plane · Data plane · SDN controller · Northbound API · Southbound API

U. Prabu · N. Malarvizhi
Department of CSE, Velagapudi Ramakrishna Siddhartha School of Engineering, Siddhartha Academy of Higher Education (Deemed to be University), Vijayawada, India
e-mail: prabu@vrsiddhartha.ac.in

N. Malarvizhi
e-mail: nmalarvizhi16@vrsiddhartha.ac.in

V. Geetha
Department of IT, Puducherry Technological University, Puducherry, India
e-mail: vgeetha@ptuniv.edu.in

A. Nayyar (✉)
School of Computer Science and Artificial Intelligence (SCA), Duy Tan University, Da Nang, Vietnam
e-mail: anandnayyar@duytan.edu.vn

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_9

9.1 Introduction to Software-Defined Networks

9.1.1 Software-Defined Networks (SDN)

Software-Defined Networks (SDN) signify a modern networking architecture in which the Control Plane (CP) is logically detached from the Data Plane (DP) [1, 2]. By supporting centralized and direct programmability, SDN provides a flexible framework that overcomes the restrictions of traditional, hardware-driven networks. This paradigm shift is devised to significantly simplify the complications of network management while concurrently supporting continuous innovation, adaptability, and the continuing evolution of networking technologies.

The control functionality is managed by a logically centralized controller that possesses a comprehensive view of the entire network. This controller governs multiple packet-forwarding devices, such as switches, which can be configured through standardized interfaces like Forwarding and Control Element Separation (ForCES) [3] and OpenFlow [4]. For instance, an OpenFlow-enabled switch maintains one or more forwarding tables that are dynamically managed by the centralized controller, thus accomplishing programmability in the CP. Forwarding tables determine how packets are processed, including whether they are forwarded, dropped, or modified. Based on the policies defined by the controller, an OpenFlow switch can operate as a router, switch, Network Address Translator (NAT), firewall, or assume other functions depending on the specified packet-handling rules. The decoupling of the CP and DP makes SDN a transformative networking technology. It not only simplifies network operation and management but also facilitates the expansion of innovative applications and novel network architectures.

9.1.2 Origin of SDN

The limitations of traditional network architectures, where proprietary hardware closely coupled the control and forwarding functions, served as the primary motivation for the development of SDN. The notion of decoupling the CP from the DP originated from initial investigations into programmable networks, active networks, and network virtualization. These investigations emphasized the advantages of decoupling control functions from hardware, thus allowing researchers and operators to inspect new protocols and rules without necessitating modifications to physical devices. The introduction of OpenFlow was a key advancement in the history of SDN, offering a realistic and standardized interface that allows external controllers to directly define the forwarding behaviour of switches. Recommended by the Open Networking Foundation (ONF), OpenFlow rapidly established itself as the standard foundation of SDN, stimulating both research and commercial implementation.

9.1.3 Key Features of SDN

SDN characterizes a transformative network paradigm that decouples the CP and DP, centralizes network management, and enables programmability. These design principles give rise to several key features that distinguish SDN from conventional networking architectures. The following are the key features:

a. **Simplified Network Management—**Centralized control lessens complexity in configuring devices and enforcing policies across the network [5].
b. **Enhanced Network Visibility—**A global network view allows comprehensive monitoring, traffic analysis, and data-driven decision-making [6].
c. **Dynamic Programmability—**The network can be programmed via software applications, enabling rapid deployment of services, policies, and protocols [5].
d. **Interoperability and Vendor Independence—**Open standards such as OpenFlow promote multi-vendor compatibility and reduce dependency on proprietary hardware [6].
e. **Flexibility and Adaptability—**Network devices can perform multiple roles (e.g., switching, routing, firewalling) based on controller-defined policies, adapting dynamically to traffic and user demands [7].
f. **Faster Innovation and Service Deployment—**Programmable interfaces and CP abstraction expedite experimentation and diminish time-to-market for novel network services [6].

9.1.4 Traditional Networking Versus SDN

SDN signifies a major evolution in network architecture, moving away from the rigid, device-centric model of conventional networking. In traditional networks, the CP and DP are strongly coupled within each network device, such as switches and routers. Each device operates autonomously, making routing and forwarding decisions based on localized information. This distributed control structure limits scalability, complicates management, and deters the deployment of innovative services [8]. In contrast, SDN implements a centralized control architecture wherein a logically centralized controller administers the entire network. This separation of the CP from the DP improves network programmability, flexibility, and overall performance [9].

The management and scalability benefits of SDN are among its most significant advantages over traditional networking. Traditional architectures require manual, device-specific configurations, often resulting in configuration errors and operational delays. SDN simplifies this by centralizing management within software controllers, enabling automated configuration, dynamic traffic management, and policy enforcement across the network [10]. Moreover, SDN's separation of network intelligence from hardware allows seamless scalability and integration of new technologies such as cloud computing and the Internet of Things (IoT). This architectural flexibility supports the rapid deployment of services and adaptive response to varying network conditions [9].

From a performance and security perspective, SDN further improves upon the limitations of traditional networks. With a global network view, SDN facilitates real-time traffic engineering, fault detection, and automated optimization to ensure efficient utilization of network resources. Centralized security control also allows for uniform policy enforcement and immediate responses to network threats [11]. Traditional networks, by contrast, depend on localized, device-level configurations that can lead to inconsistencies and vulnerabilities. SDN's intelligent, controller-driven design therefore provides enhanced reliability, resilience, and operational efficiency, establishing it as a foundation technology for the next generation of programmable and adaptive networks [12].

9.1.4.1 Objectives of the Chapter

The objectives of the chapter are:

1. To explain the fundamental concepts and motivation behind SDN;
2. To describe the historical evolution and key milestones in SDN development;
3. To illustrate the architectural components and principles of SDN;
4. And, to discuss major applications and real-world use cases of SDN.

The rest of the chapter is organized as: Section 2 discusses the history and evolution of SDN, including early networking challenges, the rise of programmable networks, and the development of OpenFlow and the ONF. Section 3 presents the fundamental principles of SDN, such as the separation of control and data plane, centralized control, programmability, and APIs. Section 4 describes the SDN architecture and its key components across the application, control, and data plane. Section 5 explores major applications of SDN in areas like traffic engineering and load balancing, SD-WAN, NFV, data centers, IoT, cloud computing, and telecommunications. Section 6 provides case studies and real-world use cases, and finally, Sect. 7 concludes the chapter with future scope.

9.2 History and Evolution of SDN

9.2.1 Early Networking Challenges

Before the emergence of SDN, traditional network architectures were characterized by a tight coupling between the CP and DP, creating significant operational and scalability challenges. Network devices such as switches and routers contained proprietary control logic, making configuration and management complex and time-consuming [5].

One of the major constraints of traditional networks was the absence of centralized control. Each network device operated autonomously, relying on distributed control

protocols such as Open Shortest Path First (OSPF) and Border Gateway Protocol (BGP), which provided limited global visibility and coordination. This decentralized approach made it difficult for administrators to implement end-to-end Quality of Service (QoS) or optimize traffic flows across the entire network [13]. Moreover, vendor dependency and lack of interoperability impeded innovation. Networking equipment relied heavily on proprietary interfaces, restricting the ability to integrate multi-vendor devices and to customize network behavior. Researchers and operators were unable to experiment or modify protocols easily due to closed hardware and firmware designs [5].

Traditional networks also relied on manual configuration and static policies, which increased operational overhead and made networks prone to human error. As modern infrastructures such as data centers, cloud computing, and mobile networks began to expand, these manual methods became inefficient and unsustainable. Consequently, these challenges highlighted the need for a programmable, flexible, and automated networking paradigm—one that could separate control from forwarding, centralize management, and introduce programmability to achieve adaptability and innovation [5, 13].

9.2.2 Emergence of Programmable Networks

The emergence of programmable networks marked a significant stage in the evolution of SDN. Several early research initiatives, beginning in the 1990s, explored the idea of decoupling network control from data forwarding, introducing programmability, and centralizing control logic to overcome the rigidity of traditional hardware-centric architectures.

Active Networks were among the first attempts to make networks programmable by allowing packets to have executable code that network nodes could process dynamically [14]. This concept emphasized DP programmability, enabling intermediate devices to perform customized operations on packets. Projects such as ANTS [15] and SwitchWare [16, 17] expanded this approach by exploring flexible packet processing and dynamic protocol deployment. However, despite its innovation, the model faced scalability and security challenges, limiting widespread adoption.

Around the same time, the Open Signaling (OPENSIG) initiative emerged, advocating open programmable interfaces for network control in Asynchronous Transfer Mode (ATM) and telecommunication systems [18]. This approach promoted the separation of transport, control, and management functions and encouraged third-party service innovation through middleware frameworks such as CORBA.

Efforts to decompose control and DP gained traction through projects like Tempest and the ForCES standard, which formalized the logical separation of network functions [3]. Similarly, projects such as Routing Control Platform (RCP), Path Computation Element (PCE), and SoftRouter introduced the idea of centralized control, allowing a single controller to manage distributed forwarding devices [19]. Further

initiatives, such as Devolved Control of ATM Networks (DCAN), emphasized delegating control and management away from hardware devices, laying a conceptual foundation similar to OpenFlow [20]. The 4D Project [21] later proposed a clean-slate design with distinct decision, dissemination, discovery, and DP, influencing later SDN architectures and controllers like NOX [22].

In 2006, the NETCONF protocol [23] introduced a standardized approach to network configuration through structured APIs, improving upon SNMP [24] but focusing primarily on configuration rather than control-plane separation. The Ethane project [25] soon followed, representing the most direct predecessor to modern SDN. Ethane established a centralized controller to apply security and policy decisions across enterprise networks, communicating with programmable switches through secure channels, concepts that directly shaped the OpenFlow architecture.

Jointly, these early efforts from Active Networks to Ethane gradually shaped the foundations of SDN by introducing network programmability, control–data plane separation, and centralized management, which together define the modern SDN paradigm.

9.2.3 Evolution of OpenFlow and ONF

The evolution of OpenFlow and the formation of the ONF represent critical milestones in the formalization and global adoption of SDN. These developments transformed SDN from a research concept into a standardized and deployable networking architecture.

The OpenFlow protocol emerged from the need to offer a practical means of implementing SDN's principle of CP and DP separation. Originating from the Stanford University Clean Slate Program in 2008, OpenFlow empowered a secure communication channel amongst a centralized controller and network devices, allowing external software to define flow-based forwarding rules [4]. Each OpenFlow-enabled switch contained one or more flow tables that matched packet headers to corresponding actions, while the controller maintained a global network view for policy-based control.

OpenFlow was built upon prior work such as Ethane, which demonstrated the feasibility of using a centralized controller to manage enterprise networks via flow-based policies [25]. This evolution provided the first practical realization of SDN, where network intelligence resided in software controllers, while hardware devices functioned as simple forwarding elements.

To accelerate OpenFlow's adoption and standardize SDN technologies, the ONF was founded in 2011 by industry leaders including Deutsche Telekom, Facebook, Google, Microsoft, and Verizon [26]. The ONF's mission was to promote open standards, develop SDN architectures, and ensure interoperability across vendors. It formalized SDN's northbound and southbound interfaces, clarified controller-switch communication through OpenFlow specifications, and encouraged innovation through open, community-driven development [27].

Successive versions of the OpenFlow protocol enhanced its flexibility and scalability. OpenFlow 1.0 (2009) defined the basic architecture and flow table structure. Version 1.1 (2011) introduced multiple tables and group abstractions for improved pipeline processing. Version 1.3 (2012) incorporated IPv6 support, meters, and tunneling mechanisms, expanding its applicability to large-scale and heterogeneous networks [26]. These advancements laid the groundwork for SDN controllers such as Floodlight, NOX, OpenDaylight, and POX, which implemented OpenFlow as their southbound interface.

The ONF also demonstrated an essential role in driving the open networking ecosystem by integrating concepts like Network Function Virtualization (NFV) and Software-Defined Wide Area Networking (SD-WAN). Over time, the organization shifted focus toward intent-based networking and software-defined infrastructures, expanding beyond OpenFlow to support broader use cases and programmable frameworks [26].

The evolution of OpenFlow and the institutionalization of the ONF transformed SDN from a conceptual innovation into an industry-standard paradigm. This progress enabled network programmability, interoperability, and automation, forming the cornerstone of modern cloud, data center, and carrier network architectures.

9.3 Fundamental Principles of SDN

9.3.1 Separation of Control Plane (CP) and Data Plane (DP)

The separation of the CP and the DP is a predominant architectural principle of SDN, distinguished as one of its defining features in both foundational and recent literature. In traditional network architectures, network devices such as switches and routers integrate both planes: they perform both the forwarding of packets (data plane) and decision-making functions like routing, topology discovery, and policy enforcement (control plane). SDN decouples these two functionalities, relocating the CP logic to a logically centralized controller while transforming forwarding devices into simpler, programmable DP elements [29–32].

9.3.2 Centralized Controller and Global View

A core architectural principle of SDN is the logical centralization of control. In contrast to traditional networks, where control decisions are distributed across routers and switches, SDN decouples the CP from the DP and consolidates the control logic into a central entity known as the SDN controller. This controller keeps a global view of the entire network, enabling it to make coordinated, optimized, and policy-driven decisions [33].

The centralized controller acts as the brain of the network, collecting real-time information from forwarding devices and maintaining knowledge of the network topology, link states, and traffic flows. With this comprehensive visibility, the controller can dynamically adjust routes, manage load distribution, and enforce QoS and security policies consistently across all devices [34]. Such a global perspective allows for network-wide optimization and automation, which are often difficult to achieve in traditional distributed control systems.

From a management perspective, the global view simplifies policy enforcement and configuration. Network administrators can define high-level policies, which the controller transforms into device-specific instructions. This eliminates the complexity of configuring individual routers or switches, reducing operational overhead and the likelihood of misconfiguration.

Despite its advantages, the centralized control model introduces several architectural challenges. Scalability is one of the most significant concerns, as the controller must handle a large volume of control messages and maintain up-to-date state information across numerous switches. Studies on SDN scalability emphasize that as the number of flows and network devices expands, maintaining a timely and accurate global view becomes computationally intensive. Furthermore, delays in state collection can cause stale or inconsistent views, resulting in suboptimal routing or degraded network performance.

To overcome these limitations, many modern SDN deployments implement a logically centralized but physically distributed controller architecture. In this model, multiple controllers operate jointly, sharing state information to maintain a unified global view while distributing the control load across the network. Research on distributed controller clustering demonstrates that such architectures enhance scalability, reliability, and fault tolerance while preserving the logical centralization that defines SDN [35].

The placement of controllers also plays a critical role in ensuring optimal performance. Studies on controller placement imply that latency between switches and controllers directly impacts the responsiveness of control operations [36]. Proper placement and clustering strategies can therefore minimize control-plane latency and enhance synchronization among controllers, ensuring consistency and high availability of network state information.

The concept of a centralized controller and its global view characterizes a paradigm shift in networking from device-centric management toward an intelligent, software-driven, and policy-oriented architecture. This shift enables real-time adaptability, efficient network management, and the foundation for innovations such as network automation, intent-based networking, and machine-learning-driven optimization.

9.3.3 Programmability and Flexibility

One of the most transformative characteristics of SDN is its inherent programmability and flexibility, which distinguish it from traditional, rigid network architectures. Programmability refers to the ability to control, modify, and optimize network behavior using software-based instructions rather than relying on static, vendor-specific configurations. Flexibility, in turn, represents the network's capacity to adapt dynamically to changing demands, traffic conditions, and emerging services without requiring fundamental hardware changes. Together, these features make SDN a foundation for agile, scalable, and innovation-driven network management.

The introduction of a programmable CP allows network operators to specify high-level policies that are automatically translated into device-specific forwarding rules through the controller. This paradigm enables dynamic updates to routing policies, traffic engineering, and load balancing, while reducing configuration complexity. Unlike traditional architectures, where device firmware dictates the network's operational logic, SDN offers a platform where new protocols, forwarding behaviors, and management functions can be implemented through software updates. The programmability has revolutionized network design, allowing the CP and DP to evolve independently and supporting fine-grained, application-aware control of network flows.

Flexibility extends beyond the CP into the DP, which has undergone a considerable transformation in recent years. Early SDN models, such as OpenFlow and ForCES provided limited programmability, primarily restricted to predefined match-action rules. Recent research highlights the emergence of fully programmable DP that enable operators to define packet processing behavior, parsing logic, and flow actions dynamically. Languages such as Programming Protocol-independent Packet Processors (P4) have become instrumental in realizing this vision, providing the means to describe how packets are processed, independent of hardware or protocol constraints.

An innovative architecture was introduced that integrates SDN control logic with reconfigurable hardware using Multiprocessor System-on-Chip (MPSoC) technology [37]. The design employs FPGA logic for high-speed data-plane processing and embedded processors for control-plane operations, achieving throughput up to 100 Gbps. This hybrid architecture exemplifies how programmability and flexibility can coexist with high performance, enabling on-the-fly reconfiguration of network functions to meet evolving demands. Such designs illustrate the transition from fixed-function networking hardware to software-defined, hardware-accelerated environments capable of supporting complex, customizable operations.

Beyond performance enhancement, SDN's programmability introduces new opportunities for service innovation. Network operators can deploy custom applications such as intrusion detection, traffic shaping, and virtualized network slicing without modifying the underlying hardware. The combination of SDN and NFV further reinforces this flexibility, enabling dynamic orchestration of network resources and services across distributed environments [38]. The resulting networks

are highly adaptive and proficient of responding in real time to user requirements and traffic variations.

However, with these benefits come numerous challenges. Researchers have noted that high programmability can introduce additional overhead in terms of latency and computational load. Fully programmable switches, for instance, may not always match the raw performance of fixed-function ASIC-based switches. Furthermore, the increased configurability heightens the risk of misconfiguration, inconsistent policy enforcement, and security vulnerabilities if not managed carefully. Maintaining balance between programmability, performance, and reliability thus remains a critical area of ongoing research [37].

Programmability and flexibility are at the core of SDN's transformative potential. They shift networking from a hardware-centric to a software-driven discipline, enabling continuous innovation, rapid deployment of services, and real-time network adaptability. Through advancements in programmable languages, hybrid architectures, and intelligent control mechanisms, SDN continues to redefine how networks are designed, managed, and evolved to meet the complex demands of modern digital infrastructures.

9.3.4 Northbound and Southbound APIs

One of the underlying norms of SDN is the use of well-defined northbound and southbound interfaces to allow seamless communication between the CP, the DP, and applications. Northbound APIs offer a structured communication channel between the SDN controller and the applications or orchestration layers that operate above it. These interfaces enable programs to request network services, establish policies, and articulate high-level network intents without requiring comprehension of the underlying infrastructure's complexities. By abstracting the network details, northbound APIs enable developers to focus on application logic while the controller translates these intents into actionable configurations for the network [39]. Standardized protocols such as RESTful APIs have become widely adopted due to their simplicity, interoperability, and alignment with modern web technologies, facilitating easier integration across diverse applications and SDN controllers.

Southbound APIs, in contrast, govern the interaction among the controller and the network devices in the DP, including switches, routers, and other forwarding elements. These interfaces let the controller program device behavior, gather network statistics, and enforce policies dynamically, thereby providing granular control over traffic flows. Protocols such as OpenFlow and NETCONF are widely used to implement southbound communication, each offering mechanisms to configure, manipulate, and monitor network devices in real time. The performance and scalability of southbound interactions are influenced by the choice of transport protocols, with research indicating that alternatives like QUIC can offer performance advantages over traditional TCP connections [40]. By providing a structured and reliable way

to control devices, southbound APIs ensure that the centralized intelligence of the controller can be effectively translated into operational network behaviors.

The integration of northbound and southbound APIs is essential to the SDN architecture, supporting the decoupling of CP and DP while promoting programmability, flexibility, and automation. Northbound APIs allow applications to communicate desired network behaviors to the controller, while southbound APIs ensure that these behaviors are faithfully executed across the network infrastructure. The continuous evolution and standardization of these interfaces are critical for achieving interoperability among different controllers, devices, and applications, which is essential for scalable, robust, and flexible SDN deployments.

9.4 SDN Architecture and Key Components

The SDN architecture signifies a paradigm shift from the traditional, hardware-centric approach to network management toward a more flexible, programmable, and centralized framework. At its core, SDN separates the CP, which makes decisions about where traffic should be sent, from the DP, which actually forwards the packets to their destinations. This separation enables centralized control, simplified management, and dynamic programmability of network resources. The ONF formally defines SDN as "the physical separation of the network CP from the forwarding plane, and where a CP controls several devices" [26].

An SDN architecture in general is composed of three fundamental layers—the application layer/plane, the control layer/plane, and the infrastructure layer/data plane—which interact through well-defined interfaces. The application plane resides at the top and contains a variety of network applications and services that utilize the centralized control of SDN. The CP sits at the center of the SDN architecture and serves as the "brain" of the network. The DP, also referred to as the infrastructure layer, forms the foundation of the SDN architecture (Fig. 9.1).

9.4.1 Application Plane

The application plane represents the topmost layer in the SDN architecture and plays a vital role in translating business and user requirements into network behavior through the SDN controller. This plane consists of a range of network applications and services designed to optimize network performance, enhance security, and provide intelligent management functionalities. Examples of such applications include network monitoring tools, firewalls, intrusion detection and prevention systems, load balancers, and traffic engineering modules. By operating at a high level of abstraction, the application plane enables administrators and developers to define desired network outcomes without dealing directly with the underlying hardware configurations.

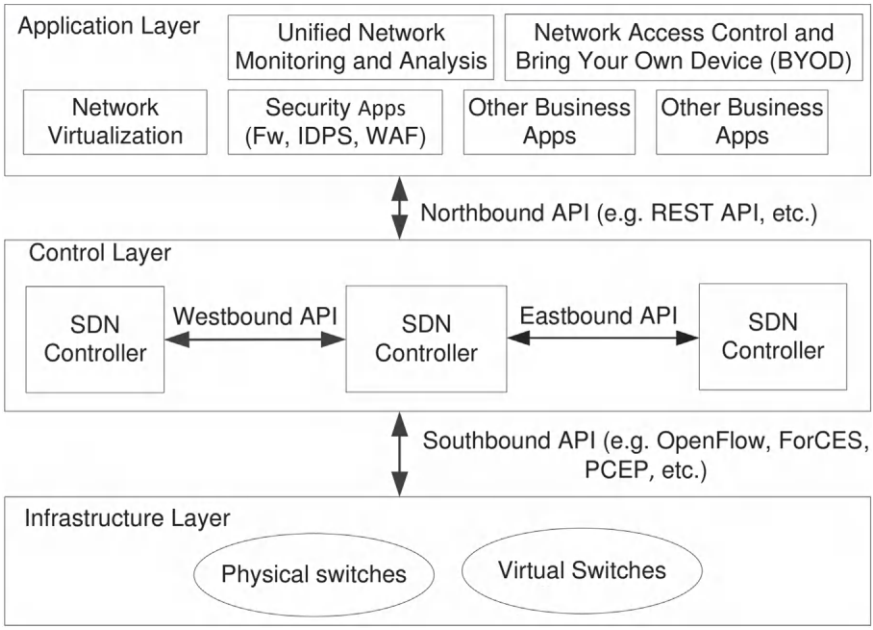


Fig. 9.1 SDN architecture [42]

The interaction between the application plane and the CP occurs through northbound Application Programming Interfaces (APIs). These APIs, often implemented using RESTful or gRPC-based frameworks, allow the applications to communicate their network requirements, performance goals, or policy intentions to the SDN controller. The controller then interprets these high-level requests and translates them into specific instructions that are enforced at the DP level. This abstraction layer ensures that network applications can operate independently of the hardware vendor, promoting interoperability and innovation in network service design.

A key benefit of the application plane is its capability to leverage the global view of the network provided by the controller. Since the controller maintains real-time information about network topology, resource utilization, and traffic patterns, applications in this plane can make data-driven decisions to improve efficiency and performance [41]. For example, traffic engineering applications can dynamically adjust flow paths to minimize congestion, while security applications can detect and respond to anomalies based on holistic network visibility.

Additionally, the programmability offered by the application plane allows for automation and customization of network behavior. Developers can build tailored applications that address specific organizational goals, such as ensuring QoS for priority users or implementing energy-efficient routing strategies. This flexibility transforms the network into an intelligent, adaptive system that responds proactively to changes in demand or operational conditions.

The application plane embodies the intelligence and adaptability of SDN by bridging user intent with network functionality. It empowers network operators to deploy innovative services, automate management tasks, and optimize performance through centralized control and abstraction [42]. By decoupling network applications from proprietary hardware dependencies, the application plane significantly enhances the agility, scalability, and programmability of modern networks.

9.4.2 Control Plane

The CP forms the central and most intelligent layer of SDN architecture. It acts as the "brain" of the network by maintaining a comprehensive and real-time view of the entire network topology, traffic conditions, and resource utilization. In traditional networks, the control logic—responsible for making routing and forwarding decisions—is embedded within each device, such as routers and switches. However, in SDN, this intelligence is decoupled from the devices and centralized within the CP. This architectural shift enables dynamic and programmable management of the network through a logically centralized controller.

The SDN controller, residing within the CP, plays a fundamental role in translating high-level policies received from the application plane into low-level instructions that can be executed by the devices in the DP. It communicates with network applications through northbound APIs and interacts with the underlying infrastructure via southbound APIs such as OpenFlow, NETCONF, or P4Runtime. Through these interfaces, the controller installs forwarding rules, monitors traffic flow, gathers network statistics, and enforces policies that align with the objectives defined by the applications.

One of the defining features of the CP is its ability to retain a global network view. By continuously collecting and analyzing information from all devices in the network, the controller can make optimized decisions related to routing, load balancing, fault recovery, and security enforcement. This centralized intelligence allows for consistent and coherent network behavior, overcoming the limitations of distributed control found in traditional architectures. For example, when congestion occurs, the controller can quickly reroute traffic via less congested paths, guaranteeing optimal utilization of network resources and improved QoS.

The CP enhances network programmability and agility. Administrators and developers can program network policies dynamically, enabling automated provisioning, real-time adaptability, and simplified network reconfiguration. This is particularly helpful in large-scale data centers and cloud environments where traffic patterns change frequently [43]. Advanced SDN controllers, such as ONOS, OpenDaylight, and Ryu, provide modular and extensible frameworks that support diverse control applications and protocols, promoting interoperability and scalability.

Despite its advantages, the centralization of control introduces challenges related to scalability, fault tolerance, and reliability. A single controller may become a performance bottleneck or a point of failure in large-scale networks. To report these issues,

modern SDN implementations often adopt distributed or hierarchical CP architectures, where multiple coordinated controllers share state information and collectively manage network operations [44]. This distributed approach preserves the logical centralization of SDN while ensuring resilience and scalability.

The CP is the central intelligence of the SDN architecture, bridging the gap between application-level policies and device-level configurations. Through centralized decision-making, network-wide visibility, and programmability, it enables efficient, flexible, and automated control over network operations, marking a fundamental departure from traditional networking paradigms.

9.4.3 Data Plane

The DP, also referred to as the forwarding plane, is a fundamental component of the SDN architecture, responsible for handling the actual transmission of packets across the network infrastructure. It comprises physical and virtual network devices—such as switches, routers, and access points—that perform packet forwarding, filtering, and encapsulation based on the instructions given by the CP. Unlike in traditional networks, where these devices also manage control logic, SDN's DP is simplified to focus solely on executing forwarding rules, thereby reducing complexity and improving performance efficiency.

In SDN, the devices in DP operate in the instructions of a logically centralized controller via standardized southbound interfaces such as OpenFlow. These interfaces allow the controller to dynamically configure the forwarding behavior of each device without requiring manual intervention. For instance, when a new flow is detected, the DP forwards the packet to the controller, which determines the appropriate path and installs flow entries across relevant switches [4]. This enables a high degree of programmability, adaptability, and centralized control over network traffic.

The DP's role extends beyond simple packet forwarding to include tasks such as traffic monitoring, queue management, and policy enforcement. By offloading decision-making to the CP, the DP can operate at wire speed, maintaining high throughput and low latency while executing predefined rules efficiently. Furthermore, the separation of control and DP enables rapid innovation in networking, as network operators can experiment with new routing algorithms or policies at the controller level without altering the underlying hardware.

Despite its operational simplicity, the DP must support scalability, reliability, and interoperability to ensure the overall robustness of the SDN environment. Modern implementations often incorporate hardware acceleration, virtual switching technologies, and hybrid SDN approaches to optimize packet handling and performance under varying workloads. In summary, the SDN DP serves as the execution layer of the network, enforcing the CP's decisions to achieve the goals of programmability, flexibility, and dynamic network management.

9.4.4 Application Programming Interfaces

Application Programming Interfaces (APIs) form the communication bridges that interconnect the various planes in the SDN architecture specifically linking the application plane, CP, and DP. These interfaces enable programmability, interoperability, and abstraction in SDN environments, allowing network behavior to be dynamically managed and optimized based on high-level policies and application requirements. APIs facilitate the interaction between the application plane and the CP through northbound APIs, and between the CP and the DP through southbound APIs.

The northbound APIs provide an abstraction layer that allows applications to communicate their network requirements and intents to the controller. These APIs enable applications such as security systems, traffic analyzers, or load balancers to express policies in a high-level language without concerning themselves with underlying network complexities. For instance, an application might request a particular QoS level for video traffic, and the SDN controller translates it into concrete forwarding rules for the DP. RESTful APIs are commonly employed in northbound communication due to their simplicity, scalability, and ease of integration with diverse applications.

Conversely, the southbound APIs function as the communication interface between the SDN controller and the forwarding devices. The most widely adopted southbound protocol OpenFlow allows the controller to install, update, or delete flow entries in the switches dynamically. This real-time programmability enables efficient traffic management, network virtualization, and rapid fault recovery. Apart from OpenFlow, other southbound protocols such as NETCONF, BGP-LS, and P4Runtime have emerged to address limitations related to scalability, hardware diversity, and protocol flexibility.

The design of robust APIs is critical to the success of SDN, as they determine the degree of openness, extensibility, and vendor neutrality in network control. Through well-defined API layers, SDN achieves modularity and flexibility, enabling innovation at the application and control layers without disrupting the DP. This separation of concerns promotes interoperability across heterogeneous network infrastructures and supports the evolution of intent-based and policy-driven networking paradigms.

9.5 Applications of SDN

9.5.1 Traffic Engineering and Load Balancing

(a) ***Traffic Engineering***

SDN architecture facilitates Traffic Engineering (TE) by allowing optimal resource allocation to meet application-specific QoS requirements, enhancing network efficiency, performance, and user satisfaction.

SDN TE mechanisms are classified into five main types [45]:

a. **Topology Discovery:** Essential for accurate network management, routing, diagnosing, and load balancing, allowing the controller to gain a comprehensive view of the network. This includes host, switch, and inter-connected link discovery.
b. **Traffic Measurement:** Crucial for evaluating network parameters, optimizing traffic, and analyzing network behavior to meet Service Level Agreements (SLAs) and user satisfaction.
c. **Traffic Load Balancing:** Aims to distribute network traffic efficiently to prevent congestion and optimize resource utilization, addressing challenges like 'elephant flows'. This is explored for both DP and CP operations.
d. **QoS Provisioning:** Focuses on ensuring different QoS levels for various applications by managing parameters such as bandwidth, delay, jitter, and packet loss.
e. **Dependability:** TE mechanisms contribute to network dependability by enhancing reliability, availability, and performance, often through solutions like fault-tolerant routing.

TE in SDN is a comprehensive approach that leverages the network's programmability to optimize resource utilization, ensure quality of service, manage traffic flows effectively, and maintain a robust and reliable network infrastructure.

(b) ***Load Balancing***

(i) ***Conventional Load Balancing Techniques***

Conventional load balancing techniques in SDN utilize traditional algorithms to distribute load across the network. These techniques typically follow a top-down approach to problem-solving and rely on symbolic logic and formalism. They aim to optimize network performance, manage incoming traffic, and QoS. Various traditional algorithms are employed for load balancing in SDN, including least connections, equal-cost multipath routing, and round-robin. These techniques address problems such as inefficient load balancing, network congestion, and the need for efficient resource allocation. For instance, some techniques focus on balancing the load on controllers by duplicating them and sharing the load, or by splitting traffic into Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) flows to ensure high availability. Others aim to measure real-time traffic to minimize communication cost and controller computation.

Conventional load balancing techniques offer several improvements, such as reduced delay, improved bandwidth utilization, and enhanced throughput. They can also lead to fewer switch migrations, lower computation times, and better packet transmission performance. For example, techniques like controller duplication and migration have been shown to reduce delay and improve bandwidth and throughput. Other techniques focus on dynamic load balancing, ensuring guaranteed quality of transmission, and adapting to mobile networks.

Despite their benefits, conventional load balancing techniques have limitations. These can include increased complexity, the need for additional controllers, higher

energy consumption, and issues with scalability and accuracy. Some techniques may also suffer from increased overhead, underutilization of resources, or being topology dependent. For instance, round-robin techniques can incur overhead due to requests that don't complete execution within allocated time and may not prioritize critical traffic [46].

(ii) ***Artificial Intelligence-Based Load Balancing Techniques***

Artificial Intelligence (AI)-based load balancing techniques employ a metaheuristic approach to address real-world problems in Software-Defined Networking (SDN). These techniques mimic human characteristics through rule-based systems to implement real-time applications within the system. They offer enhanced learning capabilities and facilitate decision-making within the SDN environment. AI-based methods solve problems by drawing on patterns from previous behavior, utilizing techniques such as swarm intelligence, rule-based systems, case-based reasoning, genetic models, and hybrid models. The primary goal of categorizing these techniques is to establish a foundation for selecting or enhancing methods to balance load in SDN. The key approaches of AI-based load balancing techniques are:

a. **Heuristic Approaches:** Researchers often use heuristic approaches to migrate switches from heavily loaded controllers to less loaded ones, thereby balancing the load. For instance, one study presented a heuristic approach that integrates swap movements and shifts into a search scheme to solve migration problems. The algorithm measures the value for both migration and exit controllers at each move and can continue searching for more complex movements if a simple switch migration is not feasible Another example involves Online Controller Load Balancing (OCLB) [47], which emphases on reducing the average response time of the controller through switch migrations, considering the distribution of real-time application delivery phases.
b. **Optimization Techniques:** Genetic-based optimization techniques are employed to minimize load imbalance in software-defined elastic optical networks and decrease service delay. These techniques typically involve three phases: selecting available solutions, verifying feasibility through defragmentation algorithms, and evaluating validity with power budget algorithms. Some studies merge ant colony optimization with dynamic load balancing to find the least loaded server and the optimal path. Particle Swarm Optimization (PSO) [47] is also used for its fast solution convergence rate, particularly in wireless Internet of Vehicles, to efficiently find vehicular nodes with lower energy to reduce power consumption and ensure reliable information delivery.
c. **Neural Networks and Fuzzy Logic:** Deep Neural Networks (DNNs) and Q-learning are utilized for path selection in SDN, especially for handling complex network traffic information that cannot be distributed with a specific probability. A backpropagation neural network combined with K-Mean clustering can train the network to forecast future user access to networking devices. Fuzzy logic is another AI subset used to analyze parameters affecting server load, leveraging SDN's control features to monitor server data, program virtual server functions, and manage energy consumption [49].

d. **Game Theory:** This approach is applied to manage social situations among competing entities within SDN. It has been used to train networks to learn appropriate flow rates from switch to controller to achieve Wardrop equilibrium, leading to stable policies and balanced load [50]. Some non-cooperative load balancing approaches also learn Wardrop equilibrium and achieve better load balancing through migration.

9.5.2 Software-Defined Wide Area Networks (SD-WAN)

Software-Defined Wide Area Networks (SD-WANs) represent one of the most impactful applications of SDN in modern enterprise and carrier networks. Traditional WAN architectures, often based on legacy technologies such as ATM, Frame Relay, or Multi-Protocol Label Switching (MPLS), suffer from limitations in flexibility, scalability, and cost efficiency. By leveraging the programmability and centralized control principles of SDN, SD-WAN addresses these challenges, offering advanced capabilities in network optimization, traffic engineering, service orchestration, and security management [51–53] (Fig. 9.2).

(a) ***Evolution of SD-WAN Applications***

The evolution of SD-WAN technology can be outlined into three key phases [51], each representing advancements in architectural design and functional capability.

(i) ***SD-WAN 1.0***

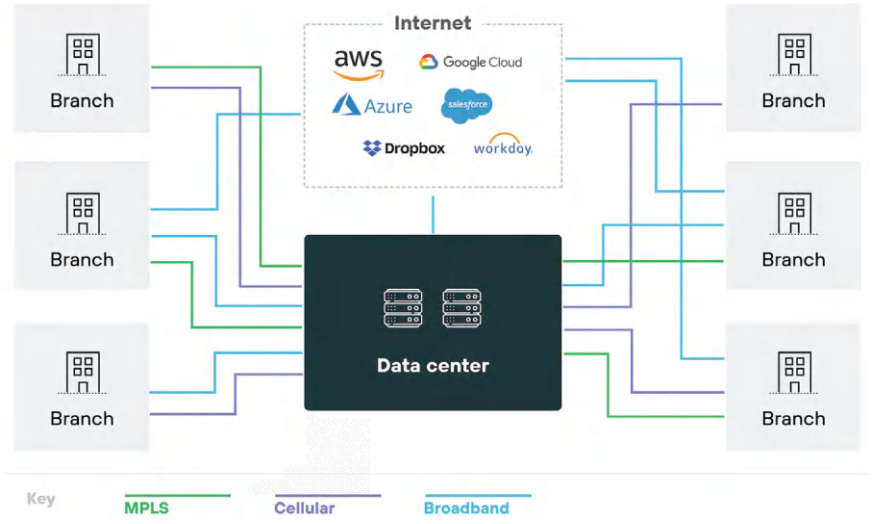


Fig. 9.2 SD-WAN architecture [54]

The earliest applications of SD-WAN focused on network optimization in interconnected data centers. Key objectives included bandwidth utilization, load balancing, and elastic computing, with Google's B4 WAN being a notable example. These early deployments emphasized cost-efficient management, traffic control, and enhanced utilization efficiency.

(ii) ***SD-WAN 2.0***

Current SD-WAN solutions have evolved into service provision platforms that integrate capabilities such as secure remote access, QoS enforcement, zero-touch deployment, and policy-driven traffic management. The integration of NFV has enabled automated interconnections across domains including cloud, edge computing, and the IoT.

(iii) ***SD-WAN 3.0***

The envisioned next phase is the use of SD-WAN as a distributed computing system, incorporating technologies like Artificial Intelligence (AI), fog computing, blockchain-based orchestration, advanced security, and IoT integration. This will expand SD-WAN from a connectivity solution into a key enabler for intelligent, self-adapting networks.

(b) ***Fundamental Applications of SD-WAN***

(i) ***Centralized Management and Control***

A fundamental application of SD-WAN lies in the separation of the CP and DP, enabling centralized decision-making for traffic routing. This architecture reduces management complexity, provides real-time network visibility, and automates monitoring tasks, thereby simplifying WAN operations [53].

(ii) ***Traffic Engineering***

SD-WAN enables advanced traffic measurement, scheduling, and failover mechanisms. It can dynamically route traffic based on application requirements, balance load across multiple underlay links (MPLS, LTE, 5G, broadband), and ensure resilience through backup tunnels and recovery schemes [53].

(iii) ***Network Optimization***

SD-WAN applications address optimization challenges such as controller placement, bandwidth allocation, and throughput maximization. Solutions also extend to multi-point VPNs [55], satellite-based connectivity for remote areas [56], Internet of Vehicles (IoV), and white-label SD-WAN systems [57]. Additionally, SD-WAN supports cloud-optimized routing, allowing direct access to SaaS applications without backhauling traffic [51].

(iv) ***Service Orchestration***

Through Service Function Chaining (SFC) and orchestration platforms, SD-WAN supports on-demand network performance guarantees and dynamic overlay selection. Cloud service providers (AWS, Azure, GCP) also deploy managed SD-WAN to ensure optimized connectivity to their cloud platforms [58].

(v) ***Security Applications***

SD-WAN integrates a range of cybersecurity capabilities such as End-to-end encryption using IPsec and NGFWs [59], Anomalous traffic detection using machine learning models [60], Zero-trust frameworks and blockchain-based orchestration for auditable and secure service delivery [61].

9.5.3 Network Function Virtualization (NFV)

The integration of NFV with SDN has opened new avenues for creating agile, programmable, and cost-efficient network infrastructures. One of the primary applications of NFV in SDN is SFC, where SDN dynamically steers traffic through a sequence of Virtual Network Functions (VNFs) such as firewalls, intrusion detection systems, and load balancers. NFV enables these network functions to be deployed flexibly as software instances on commodity hardware, while SDN provides centralized control for flow management and orchestration. A reinforcement learning-based SDN/NFV framework is proposed for QoS/QoE-aware SFC in 5G networks, where Deep Q-Network algorithms optimize VNF placement and routing decisions to enhance end-user experience while maintaining quality of service constraints [62].

Another significant application lies in 5G network slicing and virtualization, where NFV facilitates the deployment of VNFs for each network slice and SDN ensures isolation and programmable control. This integration enables operators to deliver customized virtual networks on shared infrastructure, supporting diverse use cases such as enhanced mobile broadband and ultra-reliable low-latency communication. VirtRAN, an SDN/NFV-based framework is introduced for 5G Radio Access Network (RAN) slicing, which supports multi-RAT environments and allows dynamic allocation of virtual resources among slices, improving scalability and service differentiation [63].

In addition, NFV enhances network security and monitoring in SDN environments by enabling the dynamic deployment of virtualized security functions. SDN controllers can redirect suspicious traffic to VNFs such as intrusion detection or anomaly monitoring systems on demand. A Trie-based Function Composition Algorithm within an SDN/NFV security protection architecture is proposed, where SDN controllers orchestrate the chaining of virtualized security functions for adaptive threat management [64]. Overall, NFV applications in SDN spanning service orchestration, 5G slicing, and security demonstrate the transformative potential of this synergy, though challenges in performance, orchestration complexity, and standardization remain areas for continued research.

9.5.4 Data Centers

The rapid evolution of data centers to meet ever-increasing service demands has introduced significant challenges in network management. In these massive and complex environments, careful traffic management and strict policy enforcement are critical to avoiding service disruptions that can lead to profit loss [65]. SDN offers a compelling approach to address these issues by providing a more flexible and programmable control over the network infrastructure.

One of the notable benefits of applying SDN in data centers is the potential for significant improvements in energy efficiency. For example, an SDN-based power manager known as ElasticTree can dynamically analyze network traffic to identify the minimal subset of switches required to handle the current load. By systematically shutting down unneeded network devices, this approach has been shown to achieve energy savings between 25 and 62% under varying conditions [66]. The potential for energy conservation can be further enhanced by integrating network management with server and virtualization management, as seen in systems like Honeyguide, which optimizes Virtual Machine (VM) migration to improve resource utilization [67].

While SDN simplifies traffic management, its implementation in large-scale data centers requires balancing the benefits of centralized control with potential scalability and performance overhead. To address this trade-off, solutions like DevoFlow have been proposed. DevoFlow introduces design changes that intelligently keep short-lived flows within the DP, while only escalating long-lived or more complex flows to the controller. This method effectively reduces the number of flow table entries and control messages, mitigating the burden on the central controller [68].

The viability of SDN in demanding, large-scale environments has been demonstrated by real-world implementations. Google's B4, a wide-area network that connects its data centers globally, serves as a practical example of a successful SDN deployment. This network has showcased the power of SDN to achieve highly efficient link utilization, custom routing, and sophisticated traffic engineering. The experience from this deployment also highlighted the importance of carefully considering the communication overhead between the CP and the DP in future work and designs [69].

9.5.5 Internet of Things (IoT)

The Internet of Things (IoT) aims to interconnect billions of diverse devices, but this vision is hindered by significant challenges, including device heterogeneity, widespread security vulnerabilities, and massive scalability requirements [70–72]. SDN offers a promising solution by centralizing network control and making the network programmable.

SDN addresses IoT's complexity by enabling fine-grained resource allocation and dynamic network reconfiguration to manage changing traffic patterns and diverse QoS needs [73]. Through abstraction and integration with virtualization, SDN can effectively manage a heterogeneous device ecosystem. Critically, it enhances security by providing a central point for monitoring network flows, detecting anomalies, and enforcing security policies to protect the network from insecure IoT devices, particularly at its edge.

This programmable approach makes SDN an ideal candidate for orchestrating complex IoT applications in areas like urban environments (smart cities), vehicular ad-hoc networks (VANETs) [74–76], and wireless sensor networks [77]. While SDN provides a powerful framework for advancing IoT, its full potential depends on overcoming its own challenges, such as developing greater abstractions and robust security.

9.5.6 Cloud Computing

The incorporation of SDN is fundamentally reshaping cloud and edge computing environments by changing the network into a flexible, programmable resource. By decoupling the network's CP from its DP, SDN enables dynamic resource allocation and centralized management, which are critical for the demanding nature of modern cloud applications. This architecture simplifies network operations and fosters innovation by allowing programmable control over the entire cloud network infrastructure.

At its core, SDN presents a global, intelligent view of the network, allowing cloud operators to automate and optimize resource distribution, including bandwidth, storage, and computational power. This capability is essential for efficient load balancing and dynamic adaptation to changing workloads, particularly in latency-sensitive applications like video streaming and online gaming [78, 79]. Through programmable interfaces, the network can be proactively managed, with policies automatically adjusting traffic flows to enhance performance and ensure balanced resource utilization across the cloud.

This enhanced programmability improves connectivity between geographically distributed data centers, edge nodes, and end-users, leading to significant operational cost reductions and more efficient service delivery. SDN seamlessly integrates with virtualization technologies like NFV, enabling the creation of virtual application networks and dynamic service chaining. This synergy allows service providers to manage virtualized resources efficiently, support multi-tenancy, and scale operations rapidly, effectively turning the static network into an agile fabric essential for modern cloud services.

9.5.7 Telecommunications

The 5G/6G network architecture, when based on SDN, is designed to be highly adjustable and programmable to meet diverse user and service requirements [80]. This architecture is fundamentally divided into three main parts: a control layer, an access layer, and a forward layer [81].

a. **Control Layer:** This layer presents a logical centralized network control function, separating the CP from the user plane. It acts as the control core of 5G/6G cellular networks, managing radio resources, inter-system coordination, mobility, and policy enforcement. The control layer also includes modules for information centers, path management, and network resource layout, and offers open interfaces for API infrastructure and value-added services.
b. **Access Layer:** This layer supports various wireless technologies and network forms. It is composed of numerous small 'water droplets' representing covered areas, which can be flexibly used according to specific usage scenarios. The access layer utilizes cluster-based centralized control and supports access network virtualization to meet diverse user needs.
c. **Forward Layer:** This layer, built on a common hardware platform, ensures low delay, high reliability, and balanced load for service data flow. It processes high-speed data flow and works under the direction of the control cloud.

In telecommunication networks, SDN plays a crucial role in dynamic resource allocation and network virtualization. Through techniques such as Virtual Network Embedding (VNE), SDN allows operators to allocate network resources flexibly, optimize traffic paths, and mitigate congestion in real time. This capability is vital for supporting the diverse service requirements and massive connectivity demands of 5G/6G environments.

SDN also enhances mobility management within dense and heterogeneous mobile networks [82]. Its centralized control structure enables seamless handovers, efficient user tracking, and adaptive resource distribution, improving overall user experience and reducing latency during transitions between network cells. Moreover, SDN contributes significantly to interference management in ultra-dense network deployments. By leveraging a centralized decision-making mechanism that includes modules for interference perception, analysis, and execution, SDN can dynamically adjust network configurations to minimize cross-cell interference and maintain high-quality service.

9.6 Case Studies and Real-World Use Cases

SDN has been widely adopted in real-world telecommunications, cloud, and enterprise networks, demonstrating its capability to improve flexibility, scalability, and network automation [83, 84]. Various case studies highlight how SDN technologies have revolutionized traditional network management approaches and enabled efficient service delivery across different sectors.

One prominent example is Google's B4 WAN, one of the earliest large-scale SDN deployments. Google implemented SDN to manage inter-datacenter traffic using centralized control and dynamic bandwidth allocation [69]. This approach optimized link utilization and enabled the company to achieve high network availability while reducing operational costs through efficient traffic engineering and automated resource management.

Similarly, AT&T's Domain 2.0 initiative represents another large-scale deployment of SDN and Network Function Virtualization (NFV). By integrating SDN into its core network architecture, AT&T achieved improved service agility, rapid deployment of virtualized services, and enhanced scalability to support growing 5G infrastructure demands [85]. The SDN-based automation framework allowed AT&T to transition from static hardware-driven networks to programmable, software-controlled environments. In the European Telecommunications Standards Institute (ETSI) and other European projects such as 5G-PPP, SDN has been employed to enable network slicing and dynamic orchestration of virtualized network functions for 5G networks. These implementations facilitate real-time traffic management and efficient spectrum usage across multiple tenants, significantly improving the Quality of Service (QoS) and energy efficiency.

Beyond telecommunications, cloud service providers such as Amazon Web Services (AWS) and Microsoft Azure have integrated SDN within their data centers to enhance security, scalability, and tenant isolation. The programmable nature of SDN enables these providers to offer customizable virtual networks, automated policy enforcement, and fast service provisioning. Overall, real-world case studies demonstrate that SDN provides a unified and adaptive framework for managing complex network infrastructures. Its adoption across global enterprises, telecommunication operators, and cloud providers underscores its transformative impact on modern network design, operation, and optimization.

9.7 Conclusion and Future Scope

Software-Defined Networks (SDN) has emerged as a powerful networking paradigm that fundamentally transforms how networks are designed, managed, and optimized. By separating the CP from the DP, SDN enables centralized programmability, automation, and simplified management, addressing the limitations of traditional network architectures. This chapter discussed the evolution and key principles of

SDN, including its architectural components the application, control, and DP and the essential role of northbound and southbound interfaces in enabling seamless communication across these layers. The chapter also highlighted various applications of SDN, such as Traffic Engineering and Load Balancing, SD-WAN, NFV integration, Data Centers, IoT, Cloud Computing, and Telecommunications, showcasing how SDN enhances flexibility, scalability, and operational efficiency in modern networks. Through these applications and case studies, it is evident that SDN serves as a foundation for building intelligent, adaptive, and programmable network infrastructures. In conclusion, SDN represents a major step forward in the evolution of networking, offering a unified and efficient framework that bridges innovation with practical deployment. Its ability to simplify network control and enhance performance makes it a cornerstone technology for the future of digital connectivity.

The future scope of SDN depends in its potential to further enhance automation, intelligence, and adaptability in modern networks. As networking demands continue to grow, SDN is expected to evolve through integration with technologies such as Artificial Intelligence, Machine Learning, and 5G to enable predictive and self-managing network operations. Future developments will focus on improving scalability, interoperability, and security, ensuring that SDN-based infrastructures can efficiently support dynamic and large-scale environments. With these advancements, SDN will continue to serve as a key enabler for building intelligent, efficient, and resilient communication systems that meet the needs of the next generation of digital connectivity.

References

1. O.N. Foundation, Software-Defined Networking (SDN) Definition. http://goo.gl/O2eTti.
2. Nayyar, A., Singla, B., Nagrath, P.: (Eds.). Software Defined Networks: Architecture and Applications. John Wiley & Sons (2022)
3. Doria, A., Salim, J.H., Haas, R., Khosravi, H., Wang, W., Dong, L., Ram, G., Halpern, J.: Forwarding and Control Element Separation (ForCES) Protocol Specification (No. rfc5810). https://ietf.org/wg/forces/charter/ (2010)
4. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Scott, S., Turner, J.: OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
5. Kreutz, D., Ramos, F.M., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S., Uhlig, S.: Software-defined networking: a comprehensive survey. Proc. IEEE **103**(1), 14–76 (2014)
6. Latif, Z., Sharif, K., Li, F., Karim, M.M., Biswas, S., Wang, Y.: A comprehensive survey of interface protocols for software defined networks. J. Netw. Comput. Appl. **156**, 102563 (2020)
7. Kaljic, E., Maric, A., Njemcevic, P., Hadzialic, M.: A survey on data plane flexibility and programmability in software-defined networking. IEEE Access **7**, 47804–47840 (2019)
8. Nisar, K., Jimson, E.R., Hijazi, M.H.A., Welch, I., Hassan, R., Aman, A.H.M., ... Khan, S.: A survey on the architecture, application, and security of software defined networking: challenges and open issues. Int. Things **12**, 100289 (2020)
9. Kaur, A., Krishna, C.R., Patil, N.V.: A comprehensive review on Software-Defined Networking (SDN) and DDoS attacks: ecosystem, taxonomy, traffic engineering, challenges and research directions. Comput. Sci. Rev. **55**, 100692 (2025)

10. Priyadarsini, M., Bera, P.: Software defined networking architecture, traffic management, security, and placement: a survey. Comput. Netw. **192**, 108047 (2021)
11. Etxezarreta, X., Garitano, I., Iturbe, M., Zurutuza, U.: Software-defined networking approaches for intrusion response in industrial control systems: a survey. Int. J. Crit. Infrastruct. Prot. **42**, 100615 (2023)
12. Vaezpour, E.: Deep learning-driven multi-objective dynamic switch migration in software defined networking (SDN)/network function virtualization (NFV)-based 5G networks. Eng. Appl. Artif. Intell. **125**, 106714 (2023)
13. Feamster, N., Rexford, J., Zegura, E.: The road to SDN: an intellectual history of programmable networks. ACM SIGCOMM Comput. Commun. Rev. **44**(2), 87–98 (2014)
14. Tennenhouse, D.L., Wetherall, D.J.: Towards an active network architecture. ACM SIGCOMM Comput. Commun. Rev. **26**(2), 5–17 (1996)
15. Wetherall, D.J., Guttag, J.V., Tennenhouse, D.L.: ANTS: a toolkit for building and dynamically deploying network protocols. In: 1998 IEEE Open Architectures and Network Programming, pp. 117–129. IEEE (1998)
16. Alexander, D.S., Arbaugh, W.A., Hicks, M.W., Kakkar, P., Keromytis, A.D., Moore, J.T., ... Smith, J.M.: The SwitchWare active network architecture. IEEE Netw. **12**(3), 29–36 (2002)
17. Calvert, K.L., Bhattacharjee, S., Zegura, E., Sterbenz, J.: Directions in active networks. IEEE Commun. Mag. **36**(10), 72–78 (2002)
18. Campbell, A.T., De Meer, H.G., Kounavis, M.E., Miki, K., Vicente, J.B., Villela, D.: A survey of programmable networks. ACM SIGCOMM Comput. Commun. Rev. **29**(2), 7–23 (1999)
19. Lakshman, T.V., Nandagopal, T., Ramjee, R., Sabnani, K., Woo, T.: The softrouter architecture. In: Proceeding ACM SIGCOMM Workshop on Hot Topics in Networking (Vol. 2004) (2004)
20. Devolved Control of ATM Networks. https://www.cl.cam.ac.uk/research/srg/netos/projects/archive/dcan/
21. Greenberg, A., Hjalmtysson, G., Maltz, D.A., Myers, A., Rexford, J., Xie, G., Zhang, H.: A clean slate 4D approach to network control and management. ACM SIGCOMM Comput. Commun. Rev. **35**(5), 41–54 (2005)
22. Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., Shenker, S.: NOX: towards an operating system for networks. ACM SIGCOMM Comput. Commun. Rev. **38**(3), 105–110 (2008)
23. Enns, R.: NETCONF Configuration Protocol (No. rfc4741) (2006)
24. Case, J.D., Fedor, M., Schoffstall, M.L., Davin, J.: Simple Network Management Protocol (SNMP) (No. rfc1157) (1990)
25. Casado, M., Freedman, M.J., Pettit, J., Luo, J., McKeown, N., Shenker, S.: Ethane: taking control of the enterprise. ACM SIGCOMM Comput. Commun. Rev. **37**(4), 1–12 (2007)
26. Open Networking Foundation.*: OpenFlow switch specification version 1.3.0* (ONF TS-012). Open Networking Foundation. https://opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf (2012)
27. Open Networking Foundation.: SDN Architecture Overview (ONF TR-502). Open Networking Foundation. https://opennetworking.org/wp-content/uploads/2014/10/TR_SDN-ARCH-Overview-1.1-11112014.02.pdf (2014)
28. Open Networking Foundation.: ONF Strategic Plan: Evolving the ONF to Accelerate Transformation of the Networking Industry. https://opennetworking.org/wp-content/uploads/2018/03/ONF-Strategic-Plan.pdf (2018)
29. Abuarqoub, A.: A review of the control plane scalability approaches in software defined networking. Fut. Int. **12**(3), 49 (2020)
30. Hussain, M., Shah, N., Amin, R., Alshamrani, S.S., Alotaibi, A., Raza, S.M.: Software-defined networking: categories, analysis, and future directions. Sensors **22**(15), 5551 (2022)
31. Jain, V., Yatri, V., Kanchan, A., Kapoor, C.: Software defined networking: state-of-the-art. J. High Speed Netw. **25**(1), 1–40 (2019)
32. Xie, J., Guo, D., Hu, Z., Qu, T., Lv, P.: Control plane of software defined networks: a survey. Comput. Commun. **67**, 1–10 (2015)

33. Karakus, M., Durresi, A.: A survey: control plane scalability issues and approaches in software-defined networking (SDN). Comput. Netw. **112**, 279–293 (2017)
34. Aslan, M., Matrawy, A.: Maintaining an Up-To-Date Global Network View in SDN. arXiv preprint arXiv:1612.04944. (2016).
35. Abdelaziz, A., Fong, A.T., Gani, A., Garba, U., Khan, S., Akhunzada, A., ... Choo, K.K.R.: Distributed controller clustering in software defined networks. PloS one **12**(4), e0174715 (2017)
36. Zhang, T., Giaccone, P., Bianco, A., De Domenico, S.: The role of the inter-controller consensus in the placement of distributed SDN controllers. Comput. Commun. **113**, 1–13 (2017)
37. Sha, M., Guo, Z., Guo, Y., Zeng, X.: A high-performance and flexible architecture for accelerating SDN on the MPSoC platform. Micromachines **13**(11), 1854 (2022)
38. Zhang, W., Liu, G., Mohammadkhan, A., Hwang, J., Ramakrishnan, K.K., Wood, T.: SDNFV: Flexible and dynamic software defined control of an application-and flow-aware data plane. In Proceedings of the 17th International Middleware Conference, pp. 1–12 (2016)
39. Alghamdi, A., Paul, D.J., Sadgrove, E.J.: A RESTful northbound interface for applications in software defined networks. In WEBIST, pp. 453–459 (2021)
40. Kumar, P., Dezfouli, B.: *quicSDN: Transitioning from TCP to QUIC for Southbound Communication in SDNs. arXiv preprint* arXiv:2107.08336 (2021)
41. Hu, F., Hao, Q., Bao, K.: A survey on software-defined network and openflow: from concept to implementation. IEEE Commun. Surv. Tut. **16**(4), 2181–2206 (2014)
42. Jarraya, Y., Madi, T., Debbabi, M.: A survey and a layered taxonomy of software-defined networking. IEEE Commun. Surv. Tut. **16**(4), 1955–1980 (2014)
43. Tootoonchian, A., Ganjali, Y.: Hyperflow: A distributed control plane for openflow. In Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking, Vol. 3, No. 10, p. 6 (2010)
44. Hassas Yeganeh, S., Ganjali, Y.: Kandoo: a framework for efficient and scalable offloading of control applications. In: Proceedings of the First Workshop on Hot Topics in Software Defined Networks, pp. 19–24 (2012)
45. Mohammadi, R., Akleylek, S., Ghaffari, A., Shirmarz, A.: Taxonomy of traffic engineering mechanisms in software-defined networks: a survey. Telecommun. Syst. **81**(3), 475–502 (2022)
46. Kaur, K., Kaur, S., Gupta, V.: Least time based weighted load balancing using software defined networking. In: International Conference on Advances in Computing and Data Sciences, pp. 309–314. Springer Singapore, Singapore (2016)
47. Zhang, S., Lan, J., Sun, P., Jiang, Y.: Online load balancing for distributed control plane in software-defined data center network. IEEE access **6**, 18184–18191 (2018)
48. He, X., Ren, Z., Shi, C., Fang, J.: A novel load balancing strategy of software-defined cloud/fog networking in the Internet of Vehicles. China Commun. **13**(Supplement2), 140–149 (2017)
49. Li, G., Gao, T., Zhang, Z., Chen, Y.: Fuzzy logic load-balancing strategy based on software-defined networking. In: International Wireless Internet Conference, pp. 471–482. Springer International Publishing, Cham (2017)
50. Pietrabissa, A., Celsi, L.R., Cimorelli, F., Suraci, V., Priscoli, F.D., Di Giorgio, A., Monaco, S.: Lyapunov-based design of a distributed wardrop load-balancing algorithm with application to software-defined networking. IEEE Trans. Control Syst. Technol. **27**(5), 1924–1936 (2018)
51. Fu, C., Wang, B., Wang, W.: Software-defined wide area networks (sd-wans): a survey. Electronics **13**(15), 3011 (2024)
52. Ouamri, M.A., Alharbi, T., Singh, D., Sylia, Z.: A comprehensive survey on software-defined wide area network (SD-WAN): principles, opportunities and future challenges. J. Supercomput. **81**(1), 291 (2025)
53. Troia, S., Borgianni, L., Sguotti, G., Giordano, S., Maier, G.: A comprehensive survey on software-defined wide area network. IEEE Commun. Surv. Tut. (2025)
54. SD-WAN Architecture. https://www.paloaltonetworks.com/cyberpedia/sd-wan-architecture#what-are-the-main-components-of-sd-wan-architecture
55. Elizabeth, S.J.M., Xavier, J.P.F., Rubén, P.C.M.: SD-WAN Software defined networking using DMVPN for corporate enterprises. In: 2023 18th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1–6. IEEE (2023)

56. Borgianni, L., Adami, D., Giordano, S.: Optimizing network performance and reliability with an integrated sd-wan and satellite 6g architecture. In: 2023 2nd International Conference on 6G Networking (6GNet), pp. 1–4. IEEE (2023)
57. Menoni, P., Palma, J.M., Morais, C.F.: Assessing the feasibility of developing a white label sd-wan solution for smart cities. In: 2023 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON), pp. 1–6. IEEE (2023)
58. Leivadeas, A., Pitaev, N., Falkner, M.: Analyzing the performance of SD-WAN enabled service function chains across the globe with AWS. In: Proceedings of the 2023 ACM/SPEC International Conference on Performance Engineering, pp 125–135 (2023)
59. Lopez-Millan, G., Marin-Lopez, R., Pereniguez-Garcia, F.: Towards a standard SDN-based IPsec management framework. Comput. Stand. Interfac. **66**, 103357 (2019)
60. Zhang, P., He, F., Zhang, H., Hu, J., Huang, X., Wang, J., Li, Y.: Real-time malicious traffic detection with online isolation forest over sd-wan. IEEE Trans. Inform. Forens. Secur. **18**, 2076–2090 (2023)
61. Balachandran, C., Ramachandran, G., Krishnamachari, B.: EDISON: a blockchain-based secure and auditable orchestration framework for multi-domain software defined networks. In: 2020 IEEE International Conference on Blockchain (Blockchain), pp. 144–153. IEEE (2020)
62. Chen, X., Li, Z., Zhang, Y., Long, R., Yu, H., Du, X., Guizani, M.: Reinforcement learning–based QoS/QoE-aware service function chaining in software-driven 5G slices. Trans. Emerg. Telecommun. Technol. **29**(11), e3477 (2018)
63. Nayak Manjeshwar, A., Jha, P., Karandikar, A., Chaporkar, P.: Virtran: An sdn/nfv-based framework for 5g ran slicing. J. Indian Inst. Sci. **100**(2), 409–434 (2020)
64. Hao, Z., Lin, Z., Li, R.: A SDN/NFV security protection architecture with a function composition algorithm based on trie. In: Proceedings of the 2nd International Conference on Computer Science and Application Engineering, pp. 1–8 (2018)
65. Nunes, B.A.A., Mendonca, M., Nguyen, X.N., Obraczka, K., Turletti, T.: A survey of software-defined networking: past, present, and future of programmable networks. IEEE Commun. Surv. Tut. **16**(3), 1617–1634 (2014)
66. Heller, B., Seetharaman, S., Mahadevan, P., Yiakoumis, Y., Sharma, P., Banerjee, S., McKeown, N.: Elastictree: Saving energy in data center networks. In: Nsdi, Vol. 10, pp. 249–264, (2010)
67. Shirayanagi, H., Yamada, H., Kono, K.: Honeyguide: a VM migration-aware network topology for saving energy consumption in data center networks. IEICE Trans. Inf. Syst. **96**(9), 2055–2064 (2013)
68. Curtis, A.R., Mogul, J.C., Tourrilhes, J., Yalagandula, P., Sharma, P., Banerjee, S.: DevoFlow: scaling flow management for high-performance networks. In: *Proceedings of the ACM SIGCOMM 2011 Conference*, pp. 254–265 (2011)
69. Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., Vahdat, A.: B4: experience with a globally-deployed software defined WAN. ACM SIGCOMM Comput. Commun. Rev. **43**(4), 3–14 (2013)
70. Bizanis, N., Kuipers, F.A.: SDN and virtualization solutions for the internet of things: a survey. IEEE Access **4**, 5591–5606 (2016)
71. Behera, A., Sagar Sahoo, K., Kumara Mishra, T., Nayyar, A., Bilal, M.: Enhancing DDoS detection in SDIoT through effective feature selection with SMOTE-ENN. PloS one **19**(10), e0309682 (2024)
72. Daniel, A., Krishnan, S.D., Sudha, D.N., Balusamy, B., Nayyar, A.: A novel optimized hybrid deep learning model for intrusion detection in software defined internet of things (SDIoT) environment. Clust. Comput. **28**(15), 981 (2025)
73. Cox, J.H., Chung, J., Donovan, S., Ivey, J., Clark, R.J., Riley, G., Owen, H.L.: Advancing software-defined networks: a survey. IEEE Access **5**, 25487–25526 (2017)
74. Bhatia, J., Dave, R., Bhayani, H., Tanwar, S., Nayyar, A.: SDN-based real-time urban traffic analysis in VANET environment. Comput. Commun. **149**, 162–175 (2020)

75. Nayak, R.P., Sethi, S., Bhoi, S.K., Sahoo, K.S., Nayyar, A.: Ml-mds: machine learning based misbehavior detection system for cognitive software-defined multimedia vanets (csdmv) in smart cities. Multimedia Tools Appl. **82**(3), 3931–3951 (2023)
76. Kumar, A., Krishnamurthi, R., Nayyar, A., Luhach, A.K., Khan, M.S., Singh, A.: A novel Software-Defined Drone Network (SDDN)-based collision avoidance strategies for on-road traffic monitoring and management. Veh. Commun. **28**, 100313 (2021)
77. Ku, I., Lu, Y., Gerla, M., Gomes, R. L., Ongaro, F., Cerqueira, E.: Towards software-defined VANET: architecture and services. In: 2014 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET), pp. 103–110. IEEE (2014)
78. Pan, J., Ma, L., Ravindran, R., TalebiFard, P.: HomeCloud: An edge cloud framework and testbed for new application delivery. In: 2016 23rd International Conference on Telecommunications (ICT), pp. 1–6. IEEE (2016)
79. Singh, P., Gupta, P., Jyoti, K., Nayyar, A.: Research on auto-scaling of web applications in cloud: survey, trends and future directions. Scalable Comput. Pract. Exp. **20**(2), 399–432 (2019)
80. Long, Q., Chen, Y., Zhang, H., Lei, X.: Software defined 5G and 6G networks: a survey. Mob. Netw. Appl. **27**(5), 1792–1812 (2022)
81. Yang, F.Y., Zhang, J.M., Xie, W.L., Wang, M., Wang, H.: Analysis of 5G cellular network architecture. Telecommun. Sci. **31**(5), 46–56 (2015)
82. Kukliński, S., Li, Y., Dinh, K.T.: Handover management in SDN-based mobile networks. In: 2014 IEEE Globecom Workshops (GC Wkshps), pp. 194–200. IEEE (2014)
83. Sahoo, K.S., Solanki, A., Mishra, S.K., Sahoo, B., Nayyar, A.: SDN-Supported Edge-Cloud Interplay for next generation internet of Things. CRC Press (2022)
84. Kaur, A., Singh, P., Nayyar, A.: Fog computing: building a road to IoT with fog analytics. In Fog Data Analytics for IoT Applications: Next Generation Process Model with State of the Art Technologies, pp. 59–78. Springer Singapore, Singapore (2020)
85. Basta, A., Kellerer, W., Hoffmann, M., Hoffmann, K., Schmidt, E.D.: A virtual SDN-enabled LTE EPC architecture: a case study for S-/P-gateways functions. In: 2013 IEEE SDN for future networks and services (SDN4FNS), pp. 1–7. IEEE (2013)

Chapter 10
SDN in Internet of Things (IoT)



Umesh Sahoo and Priyanka Kujur

Abstract The Internet of Things (IoT) is a rapidly growing technology that enables seamless connectivity between objects, people, and systems, improving efficiency and decision-making. Nearly every device nowadays is embedded with wireless sensors and communication devices creating a smarter environment using various technologies of IoT. Almost every device, including door locks, traffic sensors, manufacturing equipment sensors, smart devices, smart bulbs, and healthcare devices, generates and transmits data every second. Since there are billions of IoT devices using different types of networks and different protocols for communication, it becomes difficult to manage them together. Furthermore, it becomes easier for the hackers to seize devices and utilise them for other cyberattacks or as zombie devices. Since these devices are resources-constrained and often deployed in hostile environments they are also vulnerable to attacks. This is the stage where Software Defined Networking (SDN) demonstrates its significance. It centralizes the network layer of the IoT devices, enforcing an SDN Controller (like Open Daylight, Floodlight) over them. This controller authenticates and authorises the IoT devices for network access and enforces common networking protocols, allowing for secure communication between IoT devices. Furthermore, SDN offers IoT Security Applications and Policy-Based Security Applications to help with IoT implementation. This Chapter examines the drawbacks of traditional IoT Solutions, and the need of secure and centralized network controllers. It defines the need of introducing the role of SDN in IoT, and its benefits over traditional IoT networking solutions. Further it discusses the architectures of SDN-IoT Integration in detail and about the key components involved in this integration. This Chapter also dives deep to analyse the comparison between these architectures and how they overcome the issues in previously used IoT networking solutions. Further we will also highlight the scalability and security concerns with SDN-based IoTs for future research directions and next generation IoT solutions.

U. Sahoo · P. Kujur (✉)
Department of Computer Science, Ravenshaw University, Cuttack, India
e-mail: priyanka.k@ravenshawuniversity.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_10

10.1 Introduction

10.1.1 Overview of IoT

IoT (Internet of Things), is one of the vastly emerging techniques in today's world. It is a network of connected machines or devices that can interact with each other and also with the physical world. IoT doesn't have any specific definition that is accepted universally, but it is defined in different ways by different academics, businesses and industries. Some define IoT as it is nothing but interconnected things that actively take part in communication over the internet. Others define it as the IoT consists of two words, 'Internet' which is a global network, and another is 'Things' which refers to all the interconnected networked devices which share some common protocols to communicate with the physical world [1]. Some also gave a 4A's principle based on IoT which said that IoT connects people and devices "Anytime, Anyplace, Anything, Anyone" [2].

- Anytime—it referred to the property that IoT can ensure that data and services are available whenever needed
- Anyplace—it referred to the property that IoT can ensures that communication and connectivity to devices for data exchange is possible from anyplace
- Anything—this referred to the property that a wide range of physical 'things' can be connected to the Internet
- Anyone—this referred to the property of easy interaction and communication between people and devices

IoT targets to make physical devices recognizable digitally and make them intelligent so that they can share and collect data over real time. Industries define IoT as an interconnection of smart devices working together with minimal human intervention. Some definitions of IoT also say that it is a blend of the digital world with the physical world. It enables the objects of the physical world to sense, process, communicate and sometimes, act thus making the physical objects intelligent to interact with the environment and the surrounding system. Given below Fig. 10.1 shows IoT being a way to blend Physical world objects and the digital world. This diagram shows that IoT is a part of the physical world as well as the digital world.

10.1.2 Functional Blocks of IoT

To understand how IoT works, we will first need to know about the functional blocks of IoT which are the fundamental building blocks of IoT. These blocks work altogether from sensing the environment to processing information and providing services. There are six main functional blocks of IoT as follows:

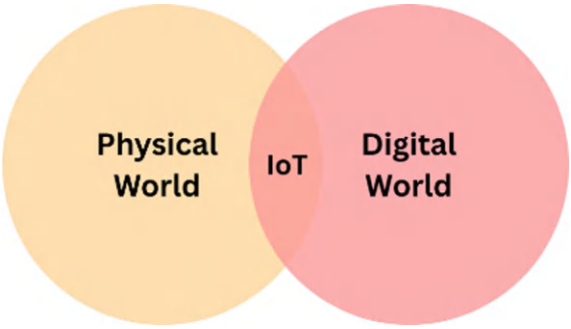


Fig. 10.1 IoT being a way to blend Physical world objects and the digital world

- Identification—There are multiple devices which communicate with each other through the internet. Since they are connected to the Internet, they must contain a uniquely identifiable address so that the device could be easily identified from the network. Traditionally we used the IPv4 addressing system and now we use the IPv6 addressing system. Since there are IoTs of devices emerging as IoT IPv4 is not sufficient to provide them IP for which IPv6 is used. The IoT devices are often lightweight and can have roles like sensor, actuator, gateway or relay. So due to their varied functionalities, only IP is not enough to identify these devices. For this reason there was a need for Service IDs/Tags like RFID (Radio Frequency Identification) EPC codes to describe their functionalities. For example, there is a smart watch which uses an IPv6 address. But the smart watch contains varied devices in it like step counter, heart monitor or notification relay which have varied roles. Thus, to identify devices only an IP address won't be sufficient for which these Service IDs are used for their identification [3].
- Sensing—The IoT devices sense and collect data from the environment and send them to the cloud to store them. This data is later used for decision making. In the case of Large-scale IoT nodes act collectively to sense and relay data. The data to be sensed depends on the application. These smart devices operate when they are triggered by some external environment like temperature, humidity, motion, pressure or some situation. But there is a challenge for these devices. Actually, these devices are energy constrained, which means they might not be able to sense for an unlimited amount of time. For this reason, they have to rely on smart sensing or energy harvesting. Most of the sensors are passive in nature i.e. they take energy from external sources as IoT devices are low cost and energy constrained. Thus, the sensors would require a hardware platform to process the input for which they rely upon SBCs(Single Board Computers) like Arduino, Raspberry Pi for processing information onboard which help processing the input [3].
- Communication—The IoT networks are a combination of heterogeneous smart devices, protocols, and communication technologies. These networks are mostly built upon WSN (Wireless Sensor Networks) where the nodes conserve energy. Thus there is a requirement of some Communication protocols that can handle data coding, flow control, sequencing and error correction. Major Communication Technology like NFC, ZigBee use low power, have a limited range and low

data rate which is generally used for short range communication. For medium range communication, technologies like Wi-Fi and WiMAX are used which have high data rate but they consume high energy. And for long range communication Technologies like LoRa WAN and Cellular(2G-5G) are used which provide an extended coverage and cost less than Wi-Fi [3].

- Computation—Since IoT devices have low cost and use low power processors, most IoTs sense and relay data to the cloud for heavy computation. Some cloud platforms like AWS IoT, Google Cloud IoT, IBM Watson, Azure IoT provide large scale computation, storage and analysis. Some devices have On-Board computation which provides some computation facilities but it is limited to cost, energy usage and complexity. This also required an OS which acted as the nervous system managing communication, processing power and services. Some of the Popular IoT OS are RIOT, TinyOS, Mbed, FreeRTOS. But with evolving technology, some IoT devices used Edge Computing for real time tasks making IoTs more efficient in computation [3].
- Services—This functional block of IoT can be understood as different layers of intelligence working together. At the base level works the Identity related services which include object sensing, identification and tracking things at real time. Once the objects are identified they collect and send data for information aggregation services in which all the data from sensors are combined for smooth transportation. Then there is collaborative aware service which enables decision making, where connected devices communicate and respond together.. And finally, there are ubiquitous services where everything becomes seamlessly intelligent and personalized. Thus, there are layers of connected devices that work together acting as a single device providing service. This is called the Service Oriented Architecture (SOA) that provides an abstraction layer to simplify device complexities and enable easier integration [3].
- Semantic and Analytics—Just like humans need language to communicate, IoT devices require semantic and analytics to understand the data generated by other IoT devices. The semantics acts like a dictionary which gives meaning to data so that devices can understand which data belongs to which object, what it conveys and how it is to be used. For example, for knowing whether the reading is from a fridge sensor or Air Conditioner sensor there is a need of analysing the semantics. Analytics works like the brain that processes information intelligently to find patterns and make predictions and smart decisions. These two together help IoT systems to go beyond just the raw numbers and build smarter systems transforming collected data into smart decisions [3].

Along with these functional blocks there are also some other crucial functional blocks such as management, security and the application layer. Management ensures remote control and monitoring of the IoT devices separating them from the traditional mechanical switch-based devices which require manual operation. This function also supports better decision making. Security is equally crucial part of IoT since the IoT devices are vulnerable to threats like denial of service (DoS), spoofing and eavesdropping. Thus, to safeguard these devices, mechanisms like content integrity,

message integrity, privacy, protection, authentication and authorization are provided by this functional block. And finally, the application layer which also acts as a functional block to provide an interface for users so that they can monitor and manage devices ensuring anytime, anywhere visibility and control. All these blocks together form a backbone of the IoT architecture making it secure, reliable and user centric.

10.1.3 Architecture of IoT

The IoT can be said to be a heterogenous system where information is collected, transmitted and processed using diverse technologies. Thus, to manage such complexities, researchers adopted a layered approach, which grouped similar functions together making it more of a structured model.

The most common among them is the three-layer architecture. This layer has three layers named as perception layer, network layer and the application layer. Given below Fig. 10.2 depicts the 3-layer architecture of IoT Systems. The perception layer is also called the physical layer and is the bottom-most layer as shown in Fig. 10.2. This layer is responsible for collecting data and converting them to digital signals using devices like sensors, actuators, RFID tags and cameras. Traditional sensors were integrated with microprocessors, to make smart sensors which can pre-process, filter and normalize data before transmission. Above the perception Layer there is the network layer which transfers this data from the physical layer to the application layer. It is the middle layer between the perception Layer and Application Layer as shown in Fig. 10.2. This layer transfers data using various communication protocols and channels. It is not the same as the Network Layer in OSI model since it goes beyond routing and includes network management tasks like data aggregation, encoding and synchronization. In places where there is no proper infrastructure available, wireless protocols like Wi-Fi, ZigBee, Bluetooth, 3G/4G/5G are used. These make IoT management flexible, cost-effective, and easy to install. But the wired mediums are still sometimes preferred due to their reliability, speed and security [4]. And finally at the top there is the Application layer which interacts with the end users. It receives and processes data received from the lower layers and transforms them into meaningful services such as real-time monitoring and decision making. It is designed according to the needs of the users. This layer is different for different areas like healthcare, industrial automation, transportation or smart homes.

Another proposed Architecture of IoT includes an extra middleware layer in between the network layer and the application layer as shown in Fig. 10.3a. This middleware layer acts as an intermediate software layer that standardizes communication between the connected heterogeneous devices. This layer simplifies development, enables interoperability and also supports integration of devices that are connected together. Some models also add a business layer on top of the application layer as shown in Fig. 10.3b. This layer is particularly based on the Service Oriented Architecture (SOA) approach. This layer allows applications to be shared, scaled and deployed across vendors and industries. Such Architecture also provides cross

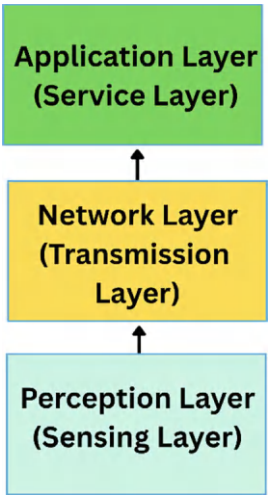


Fig. 10.2 Three-layer architecture of IoT

domain integration which is crucial for markets where IoT needs to be flexible and collaborative. Another widely recognized model for IoT is the five-layer IoT stack architecture as shown in Fig. 10.3c, which draws inspiration from the traditional IP Stack [3]. This model divides IoT functionalities into 5 distinct layers namely physical layer, data link layer, network layer, transport layer and application layer [4].

10.1.4 Growth and Applications of IoT

The growth in the global IoT market has been unmatchable. Today, businesses are spending on technologies that provide real-time data of users. Trends indicate that Industries have perceived the use of IoT not only for connection but have become drivers of efficiency. Industries such as the manufacturing industry, transportation sector, military, construction sector, waste management sectors, to name a few, have seen the immense advantages of adopting IoT [4].

The Internet of Medical Things (IoMT) has brought a significant transformation in the healthcare sector. It provides remote patient monitoring, predictive diagnosis, and improved treatment delivery. The smart sensors such as ECG monitors, oxygen level detectors, glucose testers, and wearable devices continuously collect vital signs like heart rate, blood pressure, respiration, and SpO_2. These devices enhance the quality of healthcare services, support early disease prediction and cost reduction through preventive measures. IoT also plays an important role in agriculture addressing the global challenges of population growth, climate change, and food security. Making use of IoT in agriculture makes farming precise with smart resource utilization. Automated systems such as soil-moisture-based irrigation and smart sprinklers optimize

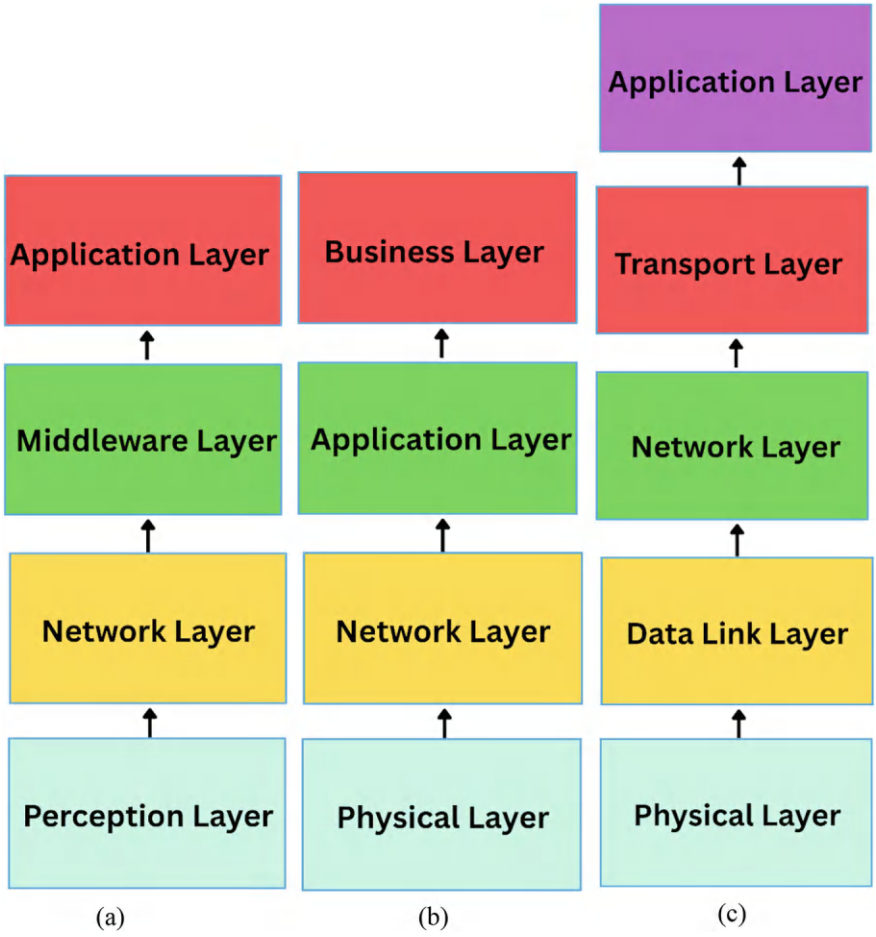


Fig. 10.3 **a** IoT architecture with middleware layer, **b** IoT architecture with business layer, and **c** five-layer IoT stack architecture

water usage. Some farm management platforms leverage real-time data for crop monitoring, weather forecasting, and supply chain efficiency. These applications result in higher productivity, reduced waste, and more sustainable farming practices [5].

Smart Homes is also another use of IoT, which provides home automation, management, and increased security within homes. Some devices such as smart speakers, sensor-enabled monitors, and other IoT enabled devices offer convenience and assistance to the elderly and people with disabilities. They offer services such as fall detection and assistance to the elderly and people with disabilities. The smart city vision is also supported by IoT technology, whose goal is to enhance the efficiency, sustainability, and quality of life of people living in cities. Traffic monitoring

systems using cameras and sensors aid in reducing traffic, and IoT-enabled trash bins also optimize their routes based on their fill levels and smell. Similarly, environmental monitoring systems employing edge computing forecast air quality, and anomaly detection tools provide safety features to detect unusual occurrences in a city. Industry 4.0 or the fourth industrial revolution is further marked by the integration of different IoT devices with different technologies like Artificial Intelligence (AI), Machine Learning (ML), Virtual and Augmented Reality (VR/AR), cloud systems, and collaborative robots or cobots. Examples of using IoT include predictive maintenance and real-time quality monitoring to smart supply chain and personnel training. These technologies result in improved efficiency, decreased operational expenses, and increased workplace security [4, 5]. Although IoT is extremely popular and widely used in different sectors, nevertheless, there exist different disadvantages or drawbacks of conventional IoT solutions, which shall be discussed in the upcoming sections.

10.2 Related Works

In this regard, the development of Software Defined Networking (SDN)-enabled IoT architectures has been the focal point of considerable research interest in the last ten years due to their potential for application in industries, smart homes, and edge computing. The early efforts were related to the adoption of SDN within IoT networks to provide flexibility, scalability, and centralized management. Theodorou et al. (2020) categorized SDN approaches over existing WSN protocols and clean-slate SDN designs [6]. Some parallel research enhanced IoT security using SDN, including those by Iqbal et al. (2020, 2021). He proposed lightweight authentication mechanisms like ALAM for smart homes and also presented centralized and decentralized controller frameworks to mitigate DoS/DDoS attacks. These works showed that SDN could enhance network management and privacy by removing bottlenecks and controller failures [7]. DoS/DDoS attack detection in SDN-based IoT has been extended further by researchers like Brajones et al.. They also showed that real potential for SDN in security management had vulnerabilities in controllers and interfaces [8]. Wani et al. extended his theory by integrating deep learning into intrusion detection systems in SDN-based environments. This improved adaptability and detection accuracy, but there still remained a challenge for handling large-scale IoT data [9].

As a result, the integration of AI and reinforcement learning became a way to overcome SDN-IoT limitations in routing and QoS management. Guo et al. proposed a deep reinforcement learning-based secure routing protocol that also optimized network performance and security, filling the gaps left by traditional traffic-aware or congestion-avoiding methods [10]. Likewise, Shafique et al. proposed application-aware SDN routing protocols like SADFIR. This application was utilized for real-time cluster management, integrating energy-efficient path selection. The key

improvements were demonstrated clearly over the legacy routing protocols, which lacked application-awareness.

The integration of SDN with blockchain technology and edge computing further enhanced the management of IoT. Turner et al. [11] presented the survey of SDN-Blockchain-IoT solutions, depicting the effectiveness of blockchain in overcoming the problem of SPFs. He also highlighted the role of blockchain technology in building trust and secure access control. However, the performance overheads were still a concern [12]. Yazdinejad et al. in 2020 focused on the energy efficiency of SDN controllers while providing security through blockchain. He filled the gap between the constraints of IoT and decentralized management [13]. Rahman et al. in 2021 presented SmartBlock-SDN for optimal resource management in IoT. He also focused on the role of blockchain in secure communication. Nonetheless, the issues of flow table saturation attacks and DDoS attacks were still present [14].

The integration of edge computing with SDN is also a significant area that many researchers have concentrated upon. Many researchers, such as Jazaeri et al. and Kamarudin et al., have performed systematic reviews of architectures that include SDN, Edge Computing, and IoT. They described the use of various architectures, such as SDIoT, SDFogIoT, and SHIoT, that allow overcoming the issues of latency, energy consumption, and scalability in the network [6, 15]. It was found that the multilayer edge computing architecture is very useful for offloading processing and controller placement. There was also the limitation of practical implementation due to the issues of memory, energy, and uncertainty of traffic.

There are also industrial and domain-specific SDN-IoT architectures that have also evolved over the past few years. Nteziriza et al. tackled the architectures of Industrial IoT and came up with hierarchical architectures for SDN-IIoT, aiming for the incorporation of AI, DHT routing, and AdaptFlow protocols for optimized communication and fault tolerance [16]. EdgeSDN-I4COVID was also developed by Rahman et al. to specifically manage industrial IoT in the context of the pandemic, focusing on the ideals of scalability, QoS, and security [17]. Likewise, a low-cost SDN architecture for smart libraries was developed by Xu et al. and showed how the benefits of SDN apply to IoT architecture and how it benefits the IoT's performance, adaptability, and security despite [18].

Finally, containerization and orchestration methods have improved the flexibility of SDN-IoT. A study by Bedhief et al. discussed different SDN-Docker designs that enhance mechanisms for handling heterogeneity in the network communication model, as well as the manageability of the control units of the edge resources [19]. With the help of software definitions such as Aloe, ImuLab, or SDRM, these designs facilitate dynamic control operations; nevertheless, these require a combination with NFV, fog computing, or blockchain for SDN by themselves to offer a complete solution.

10.3 Drawbacks of Traditional IoT Solutions

10.3.1 Scalability Issues

Conventional IoT networks experience substantial scalability issues as the number of IoT devices grows exponentially. Ideally, the present technology is capable of handling a maximum of 200,000 connections in a square kilometer in which the server-client frameworks find difficulty in handling a huge number of devices at one time. As IoT extends to smart cities, healthcare, transport, industries, and elsewhere, there is a vast need for a scalable IoT Architecture. There is a need for a proper IoT Architecture that is capable of handling a large number of devices without affecting performance [2].

10.3.2 Heterogeneity of Devices and Protocols

The other major concern in the networking of IoT is heterogeneity. The IoT environment is composed of various devices that have diverse capabilities, such as processing power, storage, energy, and communications. For example, the needs of a pacemaker are quite different from the needs of a sensor in a home automation system. Therefore, making heterogeneity manageable requires the design of frameworks that satisfy the conflicting requirements while ensuring reliability and efficiency. Therefore, heterogeneity in IoT networking requires the design of frameworks that satisfy the conflicting requirements while ensuring reliability and efficiency [2, 20].

10.3.3 Resource Constraints

Currently, IoT devices have resource constraints. They have been deployed in areas where battery replacement is expensive or impossible. Devices such as pacemakers or hearing aids do not have ultra-low power consumption but play a very important role in enhancing the usage safety of the user. Energy efficiency is a core consideration offered by these devices. Wireless power delivery, low power communication systems, and sleep cycles also play a very important role in extending the lifespan of these IoT devices, which is a key consideration in designing these devices. Balancing power is also a consideration for us if we apply an edge update [2, 3].

10.3.4 Performance Limitations

Various IoT devices produce a large amount of data from sensors. This data is processed in real time for its application in fields such as healthcare self driving cars and other domains. The delay produced by traditional and cloud centric models is a major drawback since the devices send all the data to the cloud for processing. This makes these models unsuitable for critical environments since even a few milliseconds of latency is a major concern for IoT systems. A small amount of uncertainty can lead to a significant risk in critical decision making. To address these limitations of cloud centric models approaches such as edge computing and in memory analytics have been introduced to reduce delay and enable faster and more accurate decision making. However these approaches still suffer from synchronization and consistency issues within large scale IoT networks [2].

10.3.5 Security Issues

One of the major challenges in traditional IoT networks is security. Since IoT relies a lot on wireless communication, data are mostly sent over a broadcast medium, hence it is more susceptible to any sort of interception and exploitation. The major concerns include poor authentication, encryption missing, insecure interfaces, outdated firmware, and device hijacking to carry out botnets. Moreover, several IoT devices have limited resources; therefore, deploying heavy cryptographic techniques is not possible, hence there is an urgent need for lightweight yet robust security mechanisms. The fundamental security needs that need to be met at every level and layer within the IoT environment, from the device level to the even higher level in the cloud, include confidentiality, integrity, authentication, and non-repudiation. IoT needs to provide continuous connectivity for the availability needed in the services offered to ensure the support of applications that require connectivity at any time and from anywhere. Connectivity at any time and from anywhere creates challenges in maintaining the continuity of service delivery based on the IoT device vulnerabilities related to device mobility, dynamic topological networks, and network failures. Availability and Security are significantly important in the context of applications related to health monitoring IoT, industries and other IoT applications, because the interruption in the service delivery creates life-threatening constraints in such applications [2, 4].

10.3.6 Interoperability and Standardization

There are different types or technologies used in communication between IoT devices. They include LoRa, ZigBee, Bluetooth Low Energy (BLE), Z-Wave,

Thread, NB-IoT, NFC, SigFox, among others. It is even due to this existence of different communication technologies that the problem of interoperability arises. The different technologies are designed to enable communication between devices within their scope or application. In view of the nonexistence of a standardized protocol to communicate, problems are often encountered when integrating devices from different companies or sources when communication between them is needed. Interoperability is closely followed by the problem of Standardization. This is because there are no universally accepted standards within IoTs, which causes hindrances to innovation, mobility, and mass adoption. The existence of different devices, vendors, and communication standards makes it challenging to ensure that uniformity is achieved in IoT design. Consequently, there is a need to make group decisions on consensus to design standards to be applied in designing IoTs [4].

With all such drawbacks of Traditional IoT Solutions, it is unrealistic to execute IoT on a large scale in a secure manner. Therefore, to overcome all such difficulties, SDN, which stands for Software Defined Networking, has come up as a hopeful solution. It distinguishes between the control plane and data plane, which makes it simpler to deal with IoT devices. Moreover, it provides centralized intelligence, as well as customizable network behavior, which is discussed in the following sections that would describe the basic concept of SDN and understand how SDN-IoT would be able to perform seamlessly as if there are no drawbacks of Traditional IoT networking solutions.

10.4 Introducing SDN to IoT

10.4.1 Fundamentals of SDN

Software-Defined Networking (SDN) is a paradigm shift in which modern networks are designed, managed, and operated. SDN separates the network's planes into distinct layers, which could be easily managed. These layers are distinct and have a well-defined responsibility. At the bottom there is the data plane as shown in Fig. 10.4. This plane is responsible for forwarding packets across the network. This functionality is carried out by SDN-enabled switches, which acts like a forwarding device which has no embedded intelligence for routing decisions. Then, above the data plane, lies the control plane, which is logically centralized in the form of the SDN controller. This SDN controller serves as the brain of the network, making decisions about traffic management, routing, and policies. On top of the control plane there exists the management plane or the application plane where the SDN applications run on the controller. These applications are basically high-level business policies which are translated to low-level configurations that the controller enforces through the switches. There is also an admin plane which communicates with all other planes.

The interaction between the controller in the control plane and the switches in the data plane is governed by flow rules, which typically consist of three components.

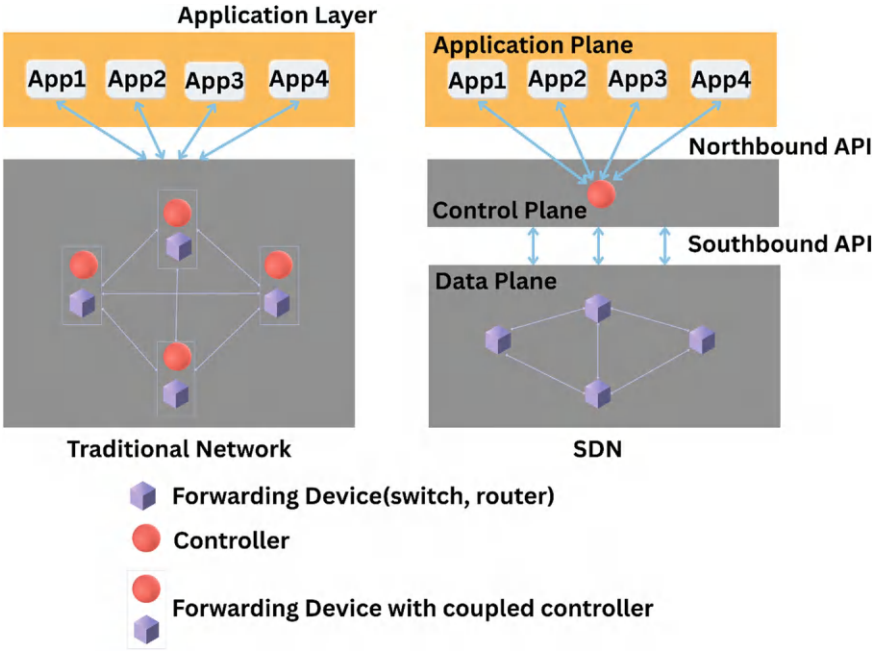


Fig. 10.4 Traditional network versus SDN

First, there is a match criteria that defines the type of packets and their flow, then an action such as forwarding, dropping, or changing packets, and lastly, statistics counters to measure statistics and performance outputs. These rules control each switch in an SDN network to ensure programmability to a level that allows each switch to be altered in terms of functions without any human intervention to match or meet network requirements without human reconfigurations. Therefore, SDN is quite different from a network system through its potential to easily manage a network compared to other network systems.

10.4.2 Architecture of SDN

Software-Defined Networking consists of many related elements which work and function together as a system that enables the management of the network programmatically and dynamically. These components are mainly divided into three categories: Application Plane (Management Plane), Control Plane, and finally the Data Plane.

The Application Plane, or management plane, supports a variety of network applications and services in order to utilize SDN services and capabilities. Various network-related applications and services include firewalls, Intrusion Detection &

Prevention Systems (IDS/IPS) solutions, routing services, load balancing services, and network virtualization services. The Application Layer communicates with Control Layer through Northbound APIs, which in turn dynamically affect network actions according to demands on a real-time basis. Figure 10.4 depicts the link between Application Plane and Control Plane and Northbound APIs [21, 22].

The Control Plane is the core component within the SDN paradigm and has been represented by the second layer in Fig. 10.4. It has a mediator nature in the context of the Application and Data Plane. All communication in the Application and Data Plane is handled by an SDN controller, defining the Control Plane. The controller exposes northbound APIs for applications at application plane and southbound APIs for switches and routers at data plane, enabling programmability and abstraction. The controllers can be categorized into centralized, flat distributed, or hierarchical distributed architectures, and each of them have their distinct advantages and trade-offs in terms of scalability, fault tolerance, and complexity. Some of the popular SDN controllers include POX (Python), Floodlight (Java), Beacon, Ryu, OpenDaylight, McNettle, and FlowVisor [21, 22].

The Data Plane, also called the forwarding plane, consists of the network devices which are actually responsible for moving packets. This layer is shown as the bottom most layer in Fig. 10.4. The packet movement is based on instructions from the controller at the Control Plane. The devices in the data plane, such as switches and routers, operate using flow tables that store rules defining packet handling. These devices can be stateless, where packets are processed independently, or stateful, where devices maintain information about previous flows to optimize forwarding decisions. The typical tasks of the data plane includes packet forwarding, topology discovery, and host tracking services (HTS). This is achieved by using protocols such as Link Layer Discovery Protocol (LLDP), OpenFlow Discovery Protocol (OFDP), Broadcast Domain Discovery Protocol (BDDP). This also offers services such as Link Discovery Service (LDS) or HTS for necessary information regarding topologies and location of host devices [21, 22].

Communication between the planes is supported through well-detailed interfaces. The Southbound Interface that is between the control plane and the data plane of the network, assists in managing the flow labels and rules into the switches. Some of the common Southbound protocols are OpenFlow, ForCES, LISP, MPLS, and NETCONF. The Southbound Interface can work in either in-band mode where the control traffic passes in the same path as the data, or in the out-of-band mode that uses some additional channels for the control messages. The Northbound Interface that connects the application plane and the controllers, supports the interaction between the applications and the network through the use of high-level abstractions. Such Interfaces typically employ the use of REST APIs, in addition to utilizing other tools such as Onix or Java-based libraries, to create programs capable of monitoring the status, policy management, and traffic optimization on such networks. Whenever the architecture in question regards physical distribution of the control plane, other interfaces such as Eastbound Interfaces (EBI) and Westbound Interfaces (WBI) also apply. Eastbound Interfaces make possible communication between various controllers, while Westbound Interfaces assist in various interactions between both

SDN and other networks; examples of EBI include: SDNi, HyperFlow, and ONIX, while examples of WBI include: RouteFlow, and SDN-IP [21, 22].

Data plane switches are not mere simple forwarding tools but are equipped with advanced functionality in the form of OpenFlow Agents (OFA), which facilitate direct communication and control of switches by an SDN controller themselves. This makes them capable of performing a variety of functions based upon the defined policy downloaded within them and ranging from a simple packet-forwarding device to a sophisticated device like a router, firewall, and even a traffic analyzer or a sniffer. Of these software switches options, Open vSwitch (OVS) is a popular open-source software switch which enhances these features even more. It also uses features such as caching and optimized flow table searches that increase speeds and efficiently use memory. The software-defined networking switches integrate the concepts of flow tables that are programmable, real-time controller interaction, and application logic to make networking much more flexible and dynamic compared to traditional technologies that rely on hardware components [21, 22]. The modification of the Traditional Network to a Software Defined Network (SDN) is shown in the figure below: Fig. 10.4: Modification of Traditional Network to Software Defined Network (SDN).

10.4.3 Advantages of SDN Over Traditional Networking

In traditional networks, decision-making is distributed, that means each device is like a router or switch which depends on its own routing tables and protocols to handle traffic and forward packets. While in the case of SDN, there is centralized decision-making using a centralized controller. This controller configures all network devices and this centralization ensures consistent policy enforcement and simplifies overall network management. The Traditional device-centric networks required manual setup, while SDN allows programmable, policy-driven configuration via controller APIs. SDN administrators can manage the entire network using this single controller. The policies defined once at the controller are automatically applied across all devices. This reduces the need for manual configuration of all the devices. This enables automation and dynamic reconfiguration of the network as requirements change [21].

By separating the control plane from the data plane, SDN allows networks to adjust dynamically. Moreover, the services can be easily customized through software application tools available in the controller. For a conventional network, adding more networks will make the architecture more complex. However, through SDN and since there is central control of the network it will be easier to implement new networks through the data plane without much need for configuration. Although there may be a need to restrict the network due to geographical or capacity limitations when a controller is solely implemented; it will be less of a concern when central control is ensured through distributed networks. The SDN networks enable effective scalability of resources depending upon the need [21].

Traditional networks support vendors and their protocols, making interoperability a major issue. In contrast to traditional networks, SDN favors non-vendor-dependent communication using protocols such as OpenFlow. Therefore, SDN is capable of supporting vendors easily by connecting new devices, minimizing dependencies on a single vendor. Interoperability in traditional networks is achieved by keeping the monitoring devices separately. While the SDN controllers can collect detailed flow data and statistics enabling better monitoring, performance optimization, and faster troubleshooting. While both SDN and traditional networks face security threats, SDN provides centralized, and better policy enforcement. It allows rapid threat detection and response, consistent application of security policies across the network, and dynamic adaptation of security mechanisms to protect against emerging threats [21, 23].

10.4.4 Role of SDN in IoT

Software Defined Networking is an important area for the improvement of the IoT environment by providing centralized control and programmatic management of the network. Unlike traditional networking, which relies on autonomous devices to manage routing as well as policies, Software Defined Networking enables a central controller to control the entire network of the IoT environment. It also enables efficiency in the routing process, load balancing, as well as the implementation of security policies. The integration of SDNs and IoT frameworks equally helps in adapting dynamically to the increasing workload and differences in network conditions. Since IoT devices have unpredictable traffic, SDNs help in efficiently managing all resources so that the network can perform well. Another feature of SDNs, automation, is made possible by their programmability, leading to rapid deployment of new services based on IoT [22, 23].

This leads to enhanced observation, control of traffic, recovery from failure, as well as quality of service, while dealing with diverse IoT devices and their respective protocols. With SDN, the scalability restrictions, compatibility, as well as maintaining the bandwidth, latency, and reliability of IoT, which are difficult to accomplish, can be achieved by IoT networks. It should be noted that this SDN also enables the flexibility, security, and scalability characteristics for IoT ecosystems. With the control and programming capabilities, SDN is able to provide optimized network control and support efficient IoT solutions in large and heterogeneous networks. It is worth noticing that the SDN acts as the basis for the future needs and requirements related to the IoT networks [21].

SDN is much more beneficial compared to the conventional network; however, its actual impact will be understood if it is implemented in a complex environment of the IoT. The flaws mentioned above highlight the need for a change in the architecture of the earlier design of the IoT. It is necessary to recognize the different SDN-IoT Integration Architectures and the manner in which they address the flaws of the

architecture of the earlier design of the IoT and its limitations. This section will discuss the principal SDN-IoTs and their functioning.

10.5 Architectures of SDN IoT Integration

10.5.1 Centralized SDN-IoT Architecture

The centralized software-defined network architecture for the Internet of Things (IoT) is a fundamental architecture based on Software-Defined Networking (SDN). It is created to improve network management, scalability, and security by splitting the control and data planes. The concept of splitting the two planes in a network makes the network intelligently controlled by an intelligent controller. Among centralized SDN-IoT deployments, two common architectural models are widely discussed, they are the classic 3-layer architecture and the 4-layer architecture [24, 25].

Centralized SDN-IoT architecture is generally organized into three fundamental layers—data plane, control plane, and application plane as shown in Fig. 10.5a. In some extended models, an additional middleware or service abstraction layer is included that forms a four-layer architecture as shown in Fig. 10.5b. This architecture further enhances programmability and interoperability. Together, these layers enable efficient management, control, and utilization of IoT resources under a software-defined networking environment.

(1) The infrastructure layer is the foundation of this architecture, better known as the data plane. It is the very lowest layer in this architecture. It is responsible for the forwarding of data packets across the network. It contains various network devices ranging from routers, physical and virtual switches, firewalls, and storage units. These forwarding devices are typically connected through an SDN-enabled gateway that bridges to the IoT infrastructure with the controller. It captures all the transmission data sent from IoT devices and routes it to the control layer for further processing. Communication between this layer and the control plane is established through the OpenFlow protocol [24].

(2) After that comes the control plane which is the spine of an SDN system and is also a centralized controller for the IoT devices. It plays an integral part in making decisions about data forwarding policy. The most popular controllers are ONOS (Open Network Operating System) and OpenDaylight controllers, which are used in this layer in abundance. This layer communicates with other layers using interfaces that are defined as southbound interfaces and northbound interfaces. The Southbound interface is used by the control plane to communicate with the data plane layer, and the Northbound interface is used by the control plane to communicate with the application plane layer. There comes another interface named eastbound interface that is used for intercontroller coordination. This concept makes systems more stable and secure by providing simplicity in load balancing and reliability [24].

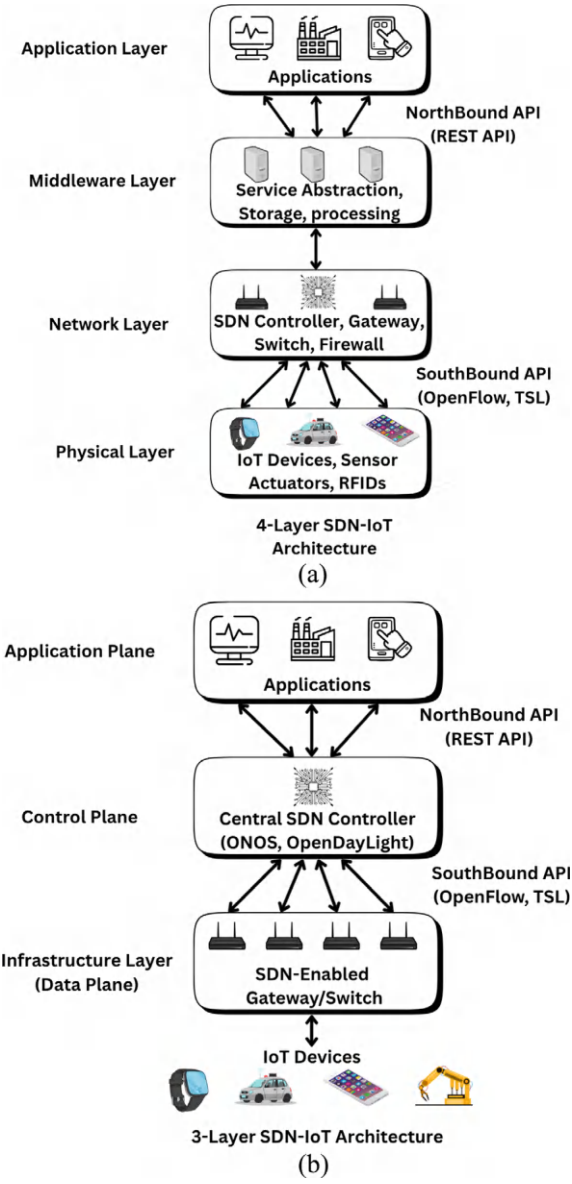


Fig. 10.5 **a** 3-layered SDN-IoT architecture and **b** 4-layered SDN-IoT architecture

(3) At the top is the application plane, which offers and performs the services demanded in various IoT application domains. This plane makes use of the SDN framework for updating the flow rules in the data plane in a dynamic manner. To achieve flexibility and efficient use of resources, the use of NFV is also supported in this plane, which makes possible execution, networking, and

other processes in a virtual environment without using physical infrastructure [24].

Though it is very simple and resource-effective for a small or medium-sized IoT network, it could have limitations in a heterogeneous or a large-scale network. To address these limitations, there comes the need for a new model, which involves additional layers for more complex applications. The 4-layer architecture includes a middleware layer, in addition to the already described sensing or physical medium layer, network layer, and application layer. It usually consists of the sensing or physical medium layer, network layer, middleware layer, and application layer. It includes end-user services. This allows a standardized API or abstraction to be established to facilitate interaction between heterogeneous IoT applications and the SDN controller. This middleware layer further improves scalability, security, and interoperability in this context. This further helps in effectively managing a network involving complex applications and a variety of IoT devices.

10.5.2 Distributed SDN-IoT Architecture

Distributed SDN-IoT architectures are designed to overcome these limitations of centralized SDN-IoT. This architecture distributes network control, intelligence, and security mechanisms across various controllers or network segments. Issues such as latency, resilience, and heterogeneous device management arise in large-scale systems. This approach overcomes those issues, enhancing general network performance [26].

The distributed SDN-IoT architecture embeds distributed Software-Defined Networking in a two-tier system. It operates based on four clear-cut layers or planes: the application layer, the control plane, the data plane, and the machine-to-machine device layer. Of these, the control, data, and M2M layers operate with the majority at the network edge. This reduces latency and offers better scalability for dynamic IoT environments [27].

At the top lies the application layer as shown in Fig. 10.6. This layer hosts typical network functions and services such as intrusion detection systems, firewalls, and load balancing mechanisms. In this architecture, these applications are not implemented directly at the device level but are instead managed and controlled by the SDN controllers. The controller decides the forwarding behaviour of the data plane [27]. Then below it there is a control plane which serves as the core decision-making component of the architecture. This model distributes the control plane across multiple controllers, each responsible for a local domain of the IoT network. These controllers maintain partial views of the topology and continuously synchronize with one another to achieve a holistic or global perspective. Below the control plane there is a data plane. This plane is responsible for forwarding packets based on instructions from the control plane. At the bottom-most as shown in Fig. 10.6, there is a M2M device layer that consists of the end devices—such as sensors, actuators, and smart

appliances—that generate and consume IoT data. These devices interact directly with the data plane via gateways, ensuring that raw data is efficiently captured, processed, and transmitted [27].

The architecture of this model is focused on modularity, scalability, and high performance. Each of the SDN controllers observes its local domain, manages link state information, and communicates this information to other controllers. This is made possible by the Publication / Subscription protocol (MQTT), which helps in efficient communication among the controllers. The IoT gateways in the data plane have dual functionality. They can handle data transmission. They can further be used as processing units to provide intelligence. They act as worker nodes in the federated learning model. Here, computations happen at the local level. This helps to minimize use of the cloud. It helps in maintaining the privacy of data while keeping the latency of data processing low [27].

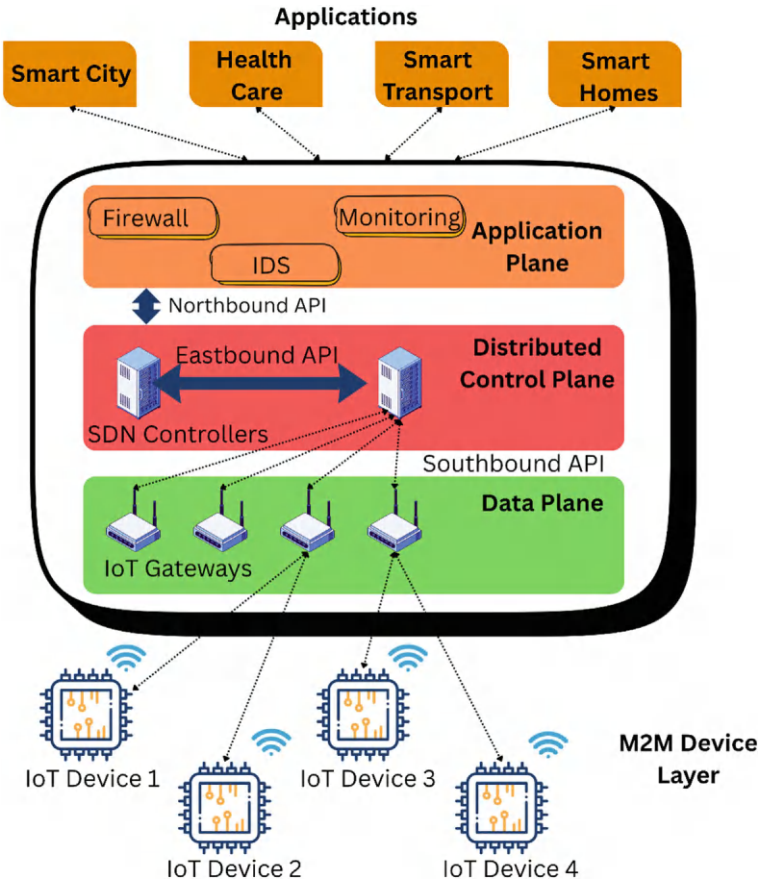


Fig. 10.6 Architecture of distributed SDN-IoT

One of the major benefits that come with the distributed SDN-IoT is the ability to scale and be flexible. The reason is that the distributed SDN-IoT breaks the network into domains or clusters and thus each cluster is controlled by a local SDN controller. This makes the network less congested and allows for better load balancing. Thus, the network is able to scale in a better manner as the number of IoT devices increases. Also, this is because there is no point of failure and the network is able to function even if one controller fails [27].

Security is another significant advantage of distributed architectures in SDN-IOTs. These architectures apply blockchain technologies for securing flow rule management. These systems promote privacy and data integrity and provide an element of protection against attacks. Furthermore, the cluster and Multi-controller design of this architecture provide segregation in security domains and make attacking easier by just focusing on localized responses to threats. Various architectures in distributed SDN-IOTs, for example, DistBlockNet and SDBlock-IOTs, combine blockchains and software-defined networking to provide real-time detection capabilities and a secure flow control service. Some technologies, for instance, SoftEdgeNet and Edge-SDN combine edge computing and software-defined networking to provide scalability and fast real-time data processing and analysis capabilities. Other architectures, for example, DistBlackNet and SDoT-NFV, combine software-defined networking and Network Function Virtualization along with clustering to provide privacy and energy efficiency and load optimization features [26, 27].

10.5.3 Cloud Based SDN-IoT Architecture

Cloud-based SDN-Iot architectures leverage the best features of SDN and cloud computing to efficiently handle and secure massive IoT networks. With the help of this architecture, we will be able to leverage the scalability and compute resources of the cloud. This architectural design is capable of dealing with resource management and end-to-end security issues in massive IoT networks. These architectures are often complemented by or built on top of network function virtualization and blockchain technology to increase agility and security [28, 29].

A primary cloud-based architecture for SDN-IoT relies on control virtualization. All elements in this architecture view the network as mere forwarding elements. These elements use commercial off-the-shelf (COTS) equipment. They do not contain any embedded intelligence that could perform control operations on their own to process traffic. For this purpose, they use the heart of this architecture that is the SDN controllers. These controllers serve as the brain of any network; they perform every control function of the network. These controllers operate in the clouds along with NFV. This approach also allows multiple service providers to share resources effectively, thus reducing the hardware and energy cost. The cloud also offers the ability to process and analyse the massive amounts of data generated by IoT devices; thus, it becomes more intelligent and responsive [28, 29].

The architecture is structured into three planes, each serving distinct roles as shown in Fig. 10.7. The data plane is the bottom most plane which comprises the physical IoT hardware which are generally COTS based. These hardware includes sensors, actuators, and gateways, whose primary responsibility is forwarding data packets. These hardware collect and forward data along with executing the instructions from the control plane. Above the data plane there lies the control plane within which lies the SDN controller. SDN Controller is responsible for device virtualization. This layer also includes a network hypervisor module as depicted in Fig. 10.7. This network hypervisor virtualizes COTS-based IoT devices and with it the network operating system helps it to manage the network as a single logical entity. Between the Data Plane and the Control Plane lies the Southbound API, which uses OpenFlow protocols to send forwarding rules downwards to the devices. At the top of this architecture there is the management plane which is a separate layer within the SDN controller. This layer ensures that administrative, monitoring, and configuration tasks are executed independently of the forwarding and control functions. In between the Control Plane and the Management, there lies the Northbound API which acts as an interface between them. This interface allows applications to request and configure network behaviour at the control planes [30].

Security remains a central concern for cloud-based SDN-IoT systems. To this there are multiple solutions that have been proposed to enhance protection. Blockchain

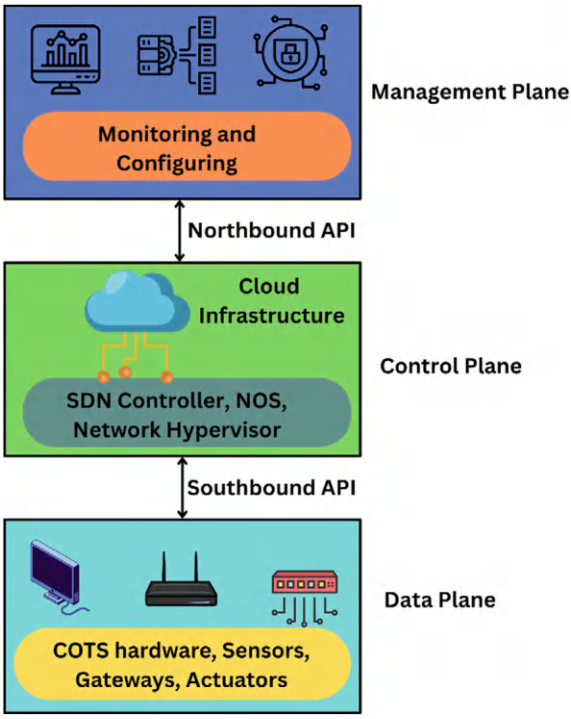


Fig. 10.7 Cloud based architecture of SDN-IoT

integration is popularly adopted to ensure data integrity, privacy, and resilience against attacks. This secures the flow management, access control, and audit trails through distributed ledger mechanisms. It also provides tamper-proof records and enhances overall trust. Many intelligent security systems, such as machine learning-based anomaly detection and collaborative intrusion detection systems, are using these architectures [29, 30].

10.5.4 Edge/Fog-Enabled SDN-IoT Architecture

Edge and fog-enabled SDN-IoT architectures are some advanced architectures that integrate software-defined networking (SDN) with edge and fog computing. They address the challenges such as latency, scalability, and security in large-scale IoT deployments. The cloud-only models discussed beforehand rely on centralized processing. While these architectures bring the computation closer to the data source. This enables real-time analytics, efficient resource management, and improved Quality of Service (QoS) for time-sensitive applications [31].

A typical multi-layered design for edge/fog-enabled SDN-IoT consists of four main components as shown in Fig. 10.8. They are namely the sensing layer (IoT devices and sensors), the processing layer (fog and edge nodes), the control layer (SDN controllers), and the application layer (cloud and data centres). The Perception Layer is the bottom most layer in this architecture which consists of IoT devices and sensors. It is also called the sensing layer. This layer collects data and interacts with the physical world and offloads raw data to the processing layer above it as shown in Fig. 10.8. The processing consists of edge and fog nodes which enables local computation. They perform local preprocessing such as filtering, aggregation and quick decision making. Since in this setup, fog and edge nodes handle localized computation, this reduces dependence on distant cloud servers. Hence this architecture minimizes latency and distributes workloads more evenly across the network.

Above the processing layer there lies the Control Layer which contains an SDN controller. This SDN controller acts as the brain of the system. It maintains traffic flows, dynamically allocation of resources and traffic paths. The Southbound APIs are used for communication between the Control layer and the processing layer. After the Control Layer comes the Application Layer which is the highest layer represented in Fig. 10.8. This layer holds the servers for the applications as well as the data center. They offer services such as analysis, business logic, as well as the dashboard. The APIs are responsible for controlling the behavior pertaining to the network from the application layer[31, 32].

These architectures provide major performance benefits. With computations offloaded to fog and edge nodes, the architecture delivers lower latency, consumes bandwidth with efficiency, and reduces energy. This architecture fits better than purely cloud-based solutions for scenarios that require fast responses. SDN further optimizes resources through dynamic partitioning and distribution of tasks across

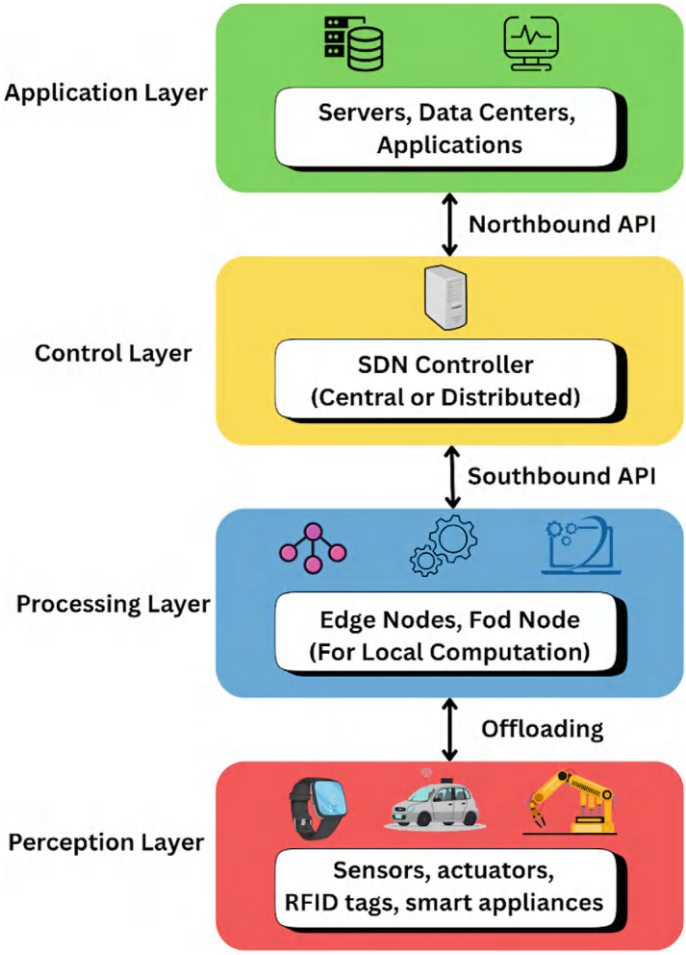


Fig. 10.8 Edge/Fog-enabled architecture of SDN-IoT

fog and edge nodes, ensuring higher throughput, load balancing, and utilization of available resources [32, 33].

It also enhances security and reliability. The integration of SDN with fog and edge computing enables programmable security functions. These include real-time analysis for detecting anomalies, encryption at the edge of the network, as well as secure data management through blockchain. Such elements promote secure networking for IoT networks against any form of break-ins or unauthorized access. The distributed architecture of fog or edge computing nodes also improves resilience to failure, which ensures that a local failure does not affect the system as a whole. It also ensures that core services are maintained even at critical moments [32, 33].

10.5.5 Hybrid SDN-IoT Architecture (Cloud + Fog + SDN)

A hybrid architecture for SDN-IoT combines the benefits of cloud computing, fog computing, and SDN networks in an effort to address the challenges faced by traditional IoT networks, which use cloud computing alone. Because the proposed architecture uses both the cloud and fog layers and orchestrates them using SDN networks, its main advantages include scalable, low-latency, and secure communication systems and resource management in IoT networks, which is very useful in various IoT operations requiring both low-latency and efficient central processing [34].

The architecture typically follows a multi-tier structure as shown in Fig. 10.9. This architecture consists of IoT devices, edge/fog nodes along with cloud servers for. The bottom layer is the sensing Layer which consists of IoT devices, sensors and RFID tags. These hardware devices are responsible for data collection from the physical environment. They forward data to the Fog Computing Layer above it as depicted in Fig. 10.9. The Fog Computing layers contain fog nodes which are close to users and devices. These fog nodes perform local processing and provide temporary storage for real time analytics. These reduce latency and avoid dependence on the cloud for every task. But since it cannot perform heavy processing load an extra layer called the Cloud Computing Layer is added above it. This layer contains cloud servers and large-scale data centres. This layer handles heavy computation, large scale analytics and long term storage. This layer is not always used but used when the task exceeds the fog-node capacity. The task moves between the sensing layer and Fog Computing Layer and the Cloud Computing Layer based on their computational needs and this is called Dynamic Task Offloading. The Fog Computing and Cloud Computing layers are unified under the control of SDN controllers in the Control Layer. This layer provides programmable management across both fog and cloud resources. This makes intelligent routing decisions, offloading decisions and efficient resource allocation. This enables dynamic traffic management and seamless integration of heterogeneous devices. The SDN controller in this layer can be centralized or distributed according to the application And next to it there is an Application Layer which contains real world applications that receives processed data through the SDN-managed network [34].

One of the defining features of hybrid SDN-IoT is dynamic task offloading. In this feature, time-critical operations are carried out at the fog nodes, thus allowing for reduced latency, while non-time critical operations are outsourced to the cloud platform for in-depth analysis and storage purposes, respectively. This is done by the SDN controller, which optimizes task distribution depending on real-time network conditions, thus ensuring an optimum balance between performance optimization and cost savings of up to three times reduced latency and twice the reduced network usage, respectively. Additionally, SDN supports optimization techniques with respect to resource usage, which encompasses fog node positioning, intelligent load balancing, and intelligence-driven allocation of bandwidth and computing resources [34, 35]. Security and high availability are also improved in Hybrid Fogy SDN architectures.

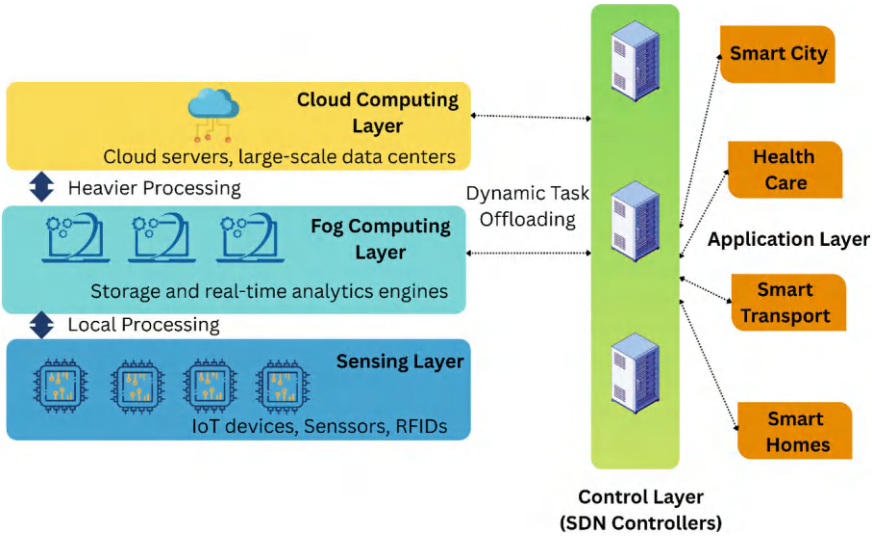


Fig. 10.9 Hybrid architecture of SDN-IoT

This includes support and inclusion of edge-level data encryption, blockchain technology, and distributed SDN controllers to improve data privacy and resistance to attacks. The blockchain technology supports an immutable record system in flow and access control, with high availability facilitated by fog and distributed SDN controllers even in cases where resources fail [34, 35].

10.5.6 Comparative Analysis of Architectures

Table 10.1 Comparative Analysis of Architectures

Aspect	Centralized SDN-IoT	Distributed SDN-IoT	Cloud based SDN-IoT	Fog/ Edge-enabled IoT	Hybrid SDN-IoT
Architectural layers	3-layer or 4-layer (with middleware between control & application layer)	Multiple controllers distributed across network	IoT devices with centralized Cloud service	IoT devices + Fog nodes for local processing	Multi-tiered IoT Architecture including IoT devices, Fog nodes, Cloud, orchestrated by SDN

(continued)

(continued)

Aspect	Centralized SDN-IoT	Distributed SDN-IoT	Cloud based SDN-IoT	Fog/ Edge-enabled IoT	Hybrid SDN-IoT
Control management	Single centralized SDN Controller	Multiple distributed SDN controllers	Centralized cloud server	Fog nodes handle local control	Unified SDN control over both fog & cloud resources
Performance	Moderate due to latency because of a single central controller	Improved then the centralized SDN-IoT	Moderate as there is latency due to distance to cloud	Hight as there is Low latency due to local processing	Very low latency since fog handles urgent tasks and cloud handles heavy tasks
Scalability	Limited	Better	High	Limited	High
Resource utilization	Static allocation, less adaptive	Load spread across controllers	High but not optimized for delay-sensitive tasks	Efficient locally, but limited globally	Optimal resource allocation since SDN manages fog placement and load balancing
Security and reliability	Vulnerable to single point of failure	Improved resilience, but complex synchronization	Depends on cloud security but vulnerable to DDoS, central attacks	Vulnerable to local attacks due to limited visibility	Strong security since it include blockchain, edge encryption, distributed SDN, fault-tolerance
Application	Small/ medium IoT systems	Mission-critical applications with fault tolerance	Big data analytics and delay-tolerant apps	Real-time local services (e.g., AR/VR, autonomous vehicles)	Large-scale, time-sensitive IoT (smart cities, healthcare, industrial IoT)

10.6 Demerits of SDN-Based IoT

Although SDN-based IoT designs ensure programmability and manageability, there are a number of inherent challenges that SDN-based IoT architectures have been enduring. Since SDN architecture has a centralized controller, there is a single-point-failure problem, which results in the whole network either becoming bogged down or an attack on the whole network. With an ever-increasing number of IoT network scales, having an SDN controller for their management is not an easier task anymore.

Although SDN provides enhanced security features, there is exposure to attacks as well. Especially, some of these malicious users attack the controller, flow tables, and control channels that can be affected by Distributed Denial of Service (DDoS), flow manipulation attacks, and unauthorized access. Moreover, the interaction of SDN with various IoT elements and legacy systems poses an added complexity. It presents difficulties in seamless integration, synchronization, and fault tolerance [36].

SDN's layered architecture further exposes it to specific vulnerabilities across different planes. In the application plane, malicious or hacked third-party applications can compromise security and cause information leakage or impersonation attacks. The control plane suffers from issues like unauthorized access, flow rule conflicts, and controller hijacking, which all threaten network stability. Similarly, the data plane is also susceptible to spoofing, new flow attacks, and metadata leakage due to insecure communication channels. The risk of DDoS attacks cuts across all planes, leveraging the central nature of SDN to seriously threaten the availability of the network [37].

Various mechanisms have been suggested to overcome these issues. Honeypot based methods that can deflect malicious network traffic to honeypot systems for further analysis to secure major systems. Moving Target Defense (MTD) methods that can reconfigure networks to increase ambiguity for malicious users, and blockchain based methods that use distributed trust systems to remove singular vulnerabilities and authenticate secure flow rules. Learning based advanced machine learning systems (LEDEM) using AI algorithms to monitor network systems in real-time to forecast attacks based on changing tactics and technologies by malicious users. Lightweight algorithms using methods based on Change-point Detection that enable effective DDoS attack systems in resource-limited IoT devices. Also using Multi-controller systems, Role-based systems with secure encryption methods addressing major risks associated with network security [36–38].

10.7 Conclusion

In conclusion, there has been a sequential advancement of SDN-based IoT architectures in matters of control centralization, security, energy efficiency, and QoS. Initially, there were approaches involving the adoption of SDN and routing functionality. This followed optimization through AI algorithms. There also followed the integration of blockchain and the implementation of edge computing. However, there are still areas of concern regarding practical implementation, energy efficiency, control centralization, vulnerability to DDoS attacks, and the integration of AI, blockchain, and edge computing.

References

1. Mansour, M., Gamal, A., Ahmed, A., Said, L., Elbaz, A., Herencsar, N., Soltan, A.: Internet of things: a comprehensive overview on protocols, architectures, technologies, simulation tools, and future directions. Energies (2023). https://doi.org/10.3390/en16083465
2. Elksasy, M.: Understanding the Internet of Things (IoT) concepts, applications and standards: an overview. Delta Univ Sci J. https://doi.org/10.21608/dusj.2023.291047
3. Ali, O., Ishak, M., Bhatti, M., Khan, I., Kim, K.: A comprehensive review of internet of things: technology stack, middlewares, and fog/edge computing interface. Sensors (Basel, Switzerland) **22** (2022). https://doi.org/10.3390/s22030995
4. Gerodimos, A., Maglaras, L., Ayres, N.: IoT: communication protocols and security threats. Int Things and Cyber-Physic Syst (2021). https://doi.org/10.20944/preprints202111.0214.v1
5. Chataut, R., Phoummalayvane, A., Akl, R.: Unleashing the power of IoT: a comprehensive review of iot applications and future prospects in healthcare, agriculture, smart homes, smart cities, and industry 4.0. sensors (Basel, Switzerland) **23** (2023). https://doi.org/10.3390/s23167194
6. Iqbal, W., Abbas, H., Deng, P., Wan, J., Rauf, B., Abbas, Y., Rashid, I.: ALAM: anonymous lightweight authentication mechanism for SDN-enabled smart homes. IEEE Int. Things J. **8**, 9622–9633 (2021). https://doi.org/10.1109/JIOT.2020.3024058
7. Galeano-Brajones, J., Carmona-Murillo, J., Valenzuela-Valdés, J., Luna-Valero, F.: Detection and Mitigation of DoS and DDoS Attacks in IoT-Based Stateful SDN: an experimental approach. Sensors (Basel, Switzerland) **20** (2020). https://doi.org/10.3390/s20030816
8. Wani, A.S.R., Khaliq, R.: SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL). CAAI Trans. Intell. Technol. **6**, 281–290 (2021). https://doi.org/10.1049/CIT2.12003
9. Guo, X., Lin, H., Li, Z., Peng, M.: Deep-reinforcement-learning-based QoS-aware secure routing for SDN-IoT. IEEE Int. Things J. **7**, 6242–6251 (2020). https://doi.org/10.1109/JIOT.2019.2960033
10. Shafique, A., Cao, G., Aslam, M., Asad, M., Ye, D.: Application-aware sdn-based iterative reconfigurable routing protocol for internet of things (IoT). Sensors (Basel, Switzerland) **20** (2020). https://doi.org/10.3390/s20123521
11. Turner, S., Karakus, M., Guler, E., Uludag, S.: A promising integration of SDN and blockchain for IoT networks: a survey. IEEE Access **11**, 29800–29822 (2023). https://doi.org/10.1109/ACCESS.2023.3260777
12. Yazdinejad, A., Parizi, R., Dehghantanha, A., Zhang, Q., Choo, K.: An Energy-efficient SDN controller architecture for IoT networks with blockchain-based security. IEEE Trans. Serv. Comput. **13**, 625–638 (2020). https://doi.org/10.1109/tsc.2020.2966970
13. Rahman, A., Islam, M., Montieri, A., Nasir, M., Reza, M., Band, S., Pescapé, A., Hasan, M., Sookhak, M., Mosavi, A.: SmartBlock-SDN: an optimized blockchain-SDN framework for resource management in IoT. IEEE Access **9**, 28361–28376 (2021). https://doi.org/10.1109/ACCESS.2021.3058244
14. Kamarudin, I., Ameedeen, M., Razak, M., Zabidi, A.: Integrating edge computing and software defined networking in internet of things: a systematic review. Iraqi J. Comput. Sci. Math. https://doi.org/10.52866/ijcsm.2023.04.04.011
15. Jazaeri, S., Jabbehdari, S., Asghari, P., Javadi, H.: Edge computing in SDN-IoT networks: a systematic review of issues, challenges and solutions. Clust. Comput. **24**, 3187–3228 (2021). https://doi.org/10.1007/s10586-021-03311-6
16. Rahman, A., Chakraborty, C., Anwar, A., Karim, M., Islam, M., Kundu, D., Rahman, Z., Band, S.: SDN–IoT empowered intelligent framework for industry 4.0 applications during COVID-19 pandemic. Clust. Comput. **25**, 2351–2368 (2021). https://doi.org/10.1007/s10586-021-03367-4
17. Xu, H., Liu, W., Li, L., Zhou, Q.: An IoT-based low-cost architecture for smart libraries using SDN. Sci. Rep. **14** (2024). https://doi.org/10.1038/s41598-024-57484-2

18. Josbert, N., Wei, M., Wang, P., Rafiq, A.: A look into smart factory for Industrial IoT driven by SDN technology: a comprehensive survey of taxonomy, architectures, issues and future research orientations. J. King Saud Univ. Comput. Inf. Sci. **36**, 102069 (2024). https://doi.org/10.1016/j.jksuci.2024.102069
19. Bedhief, I., Kassar, M., Aguili, T.: Empowering SDN-docker based architecture for internet of things heterogeneity. J. Netw. Syst. Manag. **31** (2022). https://doi.org/10.1007/s10922-022-09702-3
20. Isyaku, B., Abu Bakar, K. bin, Yusuf, N.M., Abaker, M., Abdelmaboud, A., Nagmeldin, W.: Software defined wireless sensor load balancing routing for Internet of Things applications: Review of approaches. Heliyon **10**(9), e29965 (2024). https://doi.org/10.1016/j.heliyon.2024.e29965.
21. Aljughaiman, A., Almarri, S.: The pivotal role of software-defined networks to safeguard against cyber attacks: a comprehensive review. PeerJ Comput. Sci. **11**, e2814 (2025). https://doi.org/10.7717/peerj-cs.2814
22. Anand, N., Saifulla, M.A., Ponnuru, R.B., Alavalapati, G.R., Patan, R., Gandomi, A.H.: Securing software-defined networks: a comprehensive analysis of approaches, applications, and future strategies against DoS attacks. IEEE Access **13**, 64473–64515 (2025). https://doi.org/10.1109/ACCESS.2024.3520478
23. Moorkattil, X.: Software defined networking (SDN). SSRN (2022). https://doi.org/10.2139/ssrn.4230813
24. Rahman, A., Chakraborty, C., Anwar, A., et al.: SDN–IoT empowered intelligent framework for Industry 4.0 applications during the COVID-19 pandemic. Clust. Comput. **25**, 2351–2368 (2022). https://doi.org/10.1007/s10586-021-03367-4
25. Alenezi, M., Almustafa, K., Meerja, K.A.: Cloud-based SDN and NFV architectures for IoT infrastructure. Egyptian Inform. J. **20**(1), 1–10 (2019). https://doi.org/10.1016/j.eij.2018.03.004
26. Firouzi, R., Rahmani, R.: A distributed SDN controller for distributed IoT. IEEE Access **10**, 42873–42882 (2022). https://doi.org/10.1109/ACCESS.2022.3168299
27. Islam, M.J., et al.: Blockchain-SDN-based energy-aware and distributed secure architecture for IoT in smart cities. IEEE Int. Things J. **9**(5), 3850–3864 (2022). https://doi.org/10.1109/JIOT.2021.3100797
28. Ray, P.P., Kumar, N.: SDN/NFV architectures for edge-cloud oriented IoT: a systematic review. Comput. Commun. **169**, 129–153 (2021). https://doi.org/10.1016/j.comcom.2021.01.018
29. Rahman, A., Islam, M.J., Islam, R., Aziz, A., Kundu, D., Sazzad, S., … Band, S.S.: Enhancing Data Security for cloud Computing Applications Through Distributed Blockchain-Based SDN Architecture in IoT Networks (2022). arXiv. https://arxiv.org/abs/2211.15013
30. Apat, H.K., Sahoo, B., Madapathi, P., Maiti, P.: Software-Defined Networking for Future Internet Technology (1st ed.). Apple Academic Press (2021). https://doi.org/10.1201/9781003145721-8
31. Singh, J., Singh, P., Amhoud, E.M., Hedabou, M.: Energy-efficient and secure load balancing technique for SDN-enabled fog computing. Sustainability **14**(19), 12951 (2022). https://doi.org/10.3390/su141912951
32. Sreekanth, G. R., Ahmed, S., Ahmed, N., Šarac, M., Strumberger, I., Bacanin, N., Zivkovic, M.L.: Mobile fog computing by using SDN/NFV on 5G edge nodes. Comput. Syst. Sci. Eng. **41**, 751–765 (2021). https://doi.org/10.32604/csse.2022.020534
33. Ahammad, I., Khan, M.A., Salehin, Z.: Advancement of IoT system QoS by integrating cloud, fog, roof, and dew computing assisted by SDN. Int. J. Amb. Comput. Intell. **12**, 132–153 (2021). https://doi.org/10.4018/IJACI.2021100108
34. Kar, H.A., Rather, G.M.: Multilayer Software defined networking architecture for the internet of things. Int. J. Comput. Digit. Syst. **9**(4) (2020). https://doi.org/10.12785/IJCDS/090420
35. Singh, C., Jain, A.K.: A comprehensive survey on DDoS attacks detection & mitigation in SDN-IoT network. e-Prime. Adv. Electric. Eng. Electron. Energy **8**, 100543 (2024). https://doi.org/10.1016/j.prime.2024.100543

36. Bhayo, J., Jafaq, R., Ahmed, A., Hameed, S., Shah, S.A.: A time-efficient approach toward DDoS attack detection in IoT network using SDN. IEEE Int. Things J. **9**(5), 3612–3630 (2021). https://doi.org/10.1109/JIOT.2021.3098029
37. Rahdari, A., Jalili, A., Esnaashari, M., Gheisari, M., Vorobeva, A.A., Fang, Z., Sun, P., Korzhuk, V.M., Popov, I., Wu, Z., Tahaei, H.: Security and privacy challenges in SDN-enabled IoT systems: causes, proposed solutions, and future directions. Comput. Mater. Continua **80**(2), 2511–2533 (2024). https://doi.org/10.32604/cmc.2024.052994
38. Theodorou, T., Mamatas, L.: A versatile out-of-band software-defined networking solution for the Internet of Things. IEEE Access **8**, 103710–103733 (2020). https://doi.org/10.1109/ACCESS.2020.2999087

Chapter 11
Policy-Driven Intent-Based Networking in SDN-Enabled IoT: A Paradigm Shift in Autonomous Network Management



Ankur Sisodia, Swati Vishnoi, and Ajay Kumar Yadav

Abstract The rapid growth of the Internet of Things (IoT) has introduced an incomparable level of complexity in the management of networks through the devices, protocol, and application diversity. Traditional Software-Defined Networking (SDN) solutions, although exhibiting great potential, still rely on rigid and static rules that are manually prepared by network admins. This results in the bottleneck of scalability, adaptability, and responsiveness especially in dynamic IoT environments where requirements are ever-changing. A new conceptual framework that integrates Intent-Based Networking (IBN) and SDN-based IoT systems have been put forth in this chapter to serve as a facilitator of such autonomous, context-sensitive, and human-aware network control. The system will automatically translate high-level intents, such as prioritizing emergency data streams, or dictating ethical data routing policies, expressed by the users or applications into executable SDN flow rules, will enforce them dynamically, and manage policy conflicts in real-time. The proposed IBN-SDN-IoT architecture integrates natural language processing, policy-based domain engines, and self-verifying tools for reducing the challenges of network orchestration and boosting the performance without requiring human intervention. This revolutionary approach transforms IoT network management from a low-level configuration to a high-level goal statement, marking the beginning of a new era of adaptive, ethical, and policy-driven automation. The chapter delves into architectural components, intent lifecycle, security issues, ethical enforcement, and future research directions, laying the foundation for the next generation of intention-aware IoT infrastructures.

A. Sisodia (✉)
Department of Computer Science Engineering and IoT, Noida Institute of Engineering and Technology, Greater Noida, India
e-mail: ankur22887@gmail.com

S. Vishnoi
Department of Computer Science and Engineering, Noida Institute of Engineering and Technology, Greater Noida, India

A. K. Yadav
School of Cyber Security and Digital Forensics, National Forensic Sciences University, Bhopal, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_11

Keywords IBN · SDN · IoT · CBR · Ethical policy enforcement

11.1 Introduction

The Internet of Things (IoT) is the main revolutionary cause, powering up the smart environments such as intelligent systems, smart homes, healthcare and industrial automation. These environments are characterized by the most extreme diversity of devices, communication protocols, and applications, where each has distinct needs for bandwidth, latency, reliability, and security. The challenge of handling such a diverse ecosystem in an unifying and scalable way is now regarded as one of the main issues of networking in the modern world.

Software-Defined Networking (SDN) emerged as the key enabler of programmability and centralized management, allowing network operators to dynamically shape the traffic and enforce the policies on the scattered nodes. In the IoT context, SDN is an important enabler of constrained devices, Quality of Service (QoS), and communication simplification among different types of devices. The classical SDN method relies heavily on the manual definitions of rules, which are typically in the form of expert knowledge for comprehending application requirements and translating them into low-level network commands.

The above-mentioned static and admin-specified control logic is what limits the system considerably:

- One of these various limitations is the scalability limit, as managing rule sets gets more and more difficult in large-scale dynamic networks;
- The second limitation deals with responsiveness, as policies are often not context-sensitive but rather reactive;
- Human dependence is a situation where configuration mistakes or decision delays can negatively impact either the performance or the safety of the system;
- Policy rigidity is the main reason why the system is unable to quickly adapt to new applications or changing ethical standards.

The limitations mentioned above are addressed in this chapter of the book, by introducing a new method that results from the integration of IBN concept and SDN-powered IoT management.

11.1.1 Need for Intent-Based Networking

IBN is a game-changer as it moves away from the traditional approach of specifying the network configuration and instead focuses on the desired outcome. Users or applications declare high-level "intents"—declarative objectives like these—rather than writing explicit rules for SDN controllers:

- "Give priority to ICU camera feeds when the patient's vitals exceed the thresholds."
- "Divert all educational IoT traffic through regional fog nodes during exam times."
- "Disallow any outside data flow from street sensors in alert-level weather conditions."

The Intent Engine interprets these intents, converts them into policies, and the SDN controller applies them through dynamically created flow rules. Thus, it turns out to be a closed-loop system that continuously adapts the network in real-time without any human intervention.

11.1.2 Scope and Vision of the Chapter

This chapter proposes a multi-layer IBN-SDN-IoT framework and elaborates on:

- The lifespan of intents from inception to application;
- The components of the architecture which include the Intent Engine, Policy Translator, and Verifier;
- The mechanisms for conflict detection and resolution when multiple intents coexist;
- The integration of ethical boundaries and privacy-first intents;
- The exploration of future possibilities like AI-assisted intent learning and distributed orchestration of intents.

Rather than producing a simulation model, the chapter focuses on conceptual breakthrough and the transition of SDN from a programmable platform to an autonomous intent-aware control system for the IoT era which is extremely necessary.

11.2 Literature Review

The application of Software-Defined Networking (SDN) in the Internet of Things (IoT) has undergone extensive inquiry to find solutions to traditional network architecture limitations which are not programmable, not adaptable and do not respond to real-time demands. Early contributions like [1, 2] revealed potential of using SDN in handling issues like heterogeneity, dynamic topologies, and scalability in IoT. With the growth in the area, researchers such as [3, 4] brought SDN into the framework of Network Function Virtualization (NFV) for real-time edge-cloud orchestration. Intent-Based Networking (IBN) was henceforth created as an interpreter for converting high-level intentions into low-level settings. The industrial players like Cisco and Apstra were quick to adopt this trend, and soon academic research followed. The authors in [5, 6] demonstrated systems that convert user-defined intents into network actions by means of validation and compilation techniques. However,

even though these models are very effective in classic enterprise networks, they cannot be applied easily to the IoT context due to its diverse nature, restrictions of context and device diversity, and ethical concerns across multiple domains. In response to rule-based system limitations, Case-Based Reasoning (CBR) became a cognitive approach to intelligent decision-making by reusing experienced cases to solve new problems. CBR has found its way in SDN for applications such as intrusion detection and fault diagnosis [7, 8], yet its combination with intent translation in IoT environments remains to be explored. The case-based model lets the semantic learning, contextual flexibility, and explainable reasoning aspects to shine through, which are certainly advantageous in the SDN-IoT systems where network reactions must keep up with changing user goals and moral norms. Ethics of policy enforcement and policy management across areas present another major research issue. With more attention being paid to data privacy, user consent, and compliance with regulations (e.g., GDPR, HIPAA), IoT systems must not only be efficient but also operate justly and transparently as well. In [9, 10], authors demonstrated frameworks that convert user-intents into network actions through validation and compilation, which have also proposed access control and secure interoperability models for IoT that, however, keep on fortifying a static ethical milieu and lacking the situational intelligence required for the resolution of policy disagreement in such sensitive areas as healthcare, transport, and industry. The mentioned shortcomings signal a dire need for a network-level ethical decision-making system that can keep pace with shifting policies and specific governance models prevalent in various domains.

This part of the work concentrates on recognizing some limitations of control based on SDN, while a more thorough discussion is offered in Sect. 3. The current chapter is unique in that it suggests a new combination of CBR in an intent-based SDN-IoT framework as indicated in Table 11.1. This approach not only leverages past instances of policy enforcement but also resolves ethical as well as domain-specific policy disputes in real-time, all without the requirement of simulation-based evaluation or static system modeling.

11.3 Limitations of Manual Policy and Rule-Based Control in SDN-IoT

The rollout of Software-Defined Networking (SDN) has made it possible to manage IoT ecosystems through rule-based policies, which, however, remain largely manual, static, and reactive. This process typically involves the network admin translating application-level requirements—e.g., latency sensitivity, bandwidth, or device priority—into low-level flow rules through controller interfaces like OpenFlow. While it is still manageable for small or quite stable networks, the situation gets worse in the case of large-scale IoT systems where it becomes totally impractical.

Table 11.1 State of art for SDN

Ref. no.	Key findings	Research gap	Comparison with our work
[11]	Proposed SDN integration to handle IoT complexity and scalability	Lacked semantic intent translation and user-centric adaptability	Our work introduces intent abstraction and adaptive CBR-based reasoning for user-oriented control
[12]	Discussed SDN for dynamic IoT resource management	Did not consider high-level automation or ethical policy handling	We go beyond dynamic flow control to include ethics-aware and cross-domain policy management
[13]	Combined SDN and NFV for IoT service orchestration	Focused on technical orchestration without semantic or ethical integration	Our framework incorporates semantic learning and ethical reasoning within SDN orchestration
[14]	Introduced intent compilers to map user needs into network rules	Applicable to enterprise networks, not designed for IoT heterogeneity or dynamic policies	We target heterogeneous IoT domains and support domain-specific and evolving intents
[15]	Defined high-level intent interfaces for IBN	Did not address conflict resolution in multi-domain environments	Our work uses CBR to resolve domain and ethical conflicts based on past decisions
[16]	Applied CBR in SDN for anomaly detection	Limited to security contexts; no application in policy or intent reasoning	We repurpose CBR for intent interpretation and ethical, domain-aware automation
[17]	Used CBR for fault diagnosis in SDN control plane	Lacked integration with IBN or support for user-defined policy resolution	We integrate CBR directly into the IBN engine to enable adaptive policy enforcement
[18]	Developed access control mechanisms for ethical IoT	Focused on static rules; did not support contextual ethics or evolving norms	We provide dynamic, case-driven ethical policy adaptation based on application context
[19]	Presented IoT interoperability and privacy frameworks	Did not provide programmable network-level enforcement of privacy preferences	Our model enables network-level enforcement of user privacy using intent and CBR mechanisms

11.3.1 Static Configuration in a Dynamic World

IoT networks are always changing and full of surprises: new devices are added or removed from the network, and the data rates in the data stream may go up or down, and environmental or contextual factors might change so fast that one can hardly notice. The static flow rules, once established, cannot cope with real-time changes because they are not flexible enough. For example:

- The traffic camera feed in a smart city network might need to be given utmost priority during rush hour or emergencies.
- The reliability of the network in an agricultural IoT system may need to be increased during the critical irrigation periods.

In rule-based SDN, the behavior is to be implemented by predefining all the possible contexts and writing numerous condition-specific rules—an approach that quickly becomes unscalable and error-prone.

11.3.2 Human Dependency and Operational Bottlenecks

The conventional SDN control was very much a manual effort on the part of the administrators who had to:

- Deliver the application requirements,
- Define flow rules for the application,
- Conduct tests and put the rules in place,
- Keep track of the performance, and if needed, do the necessary updates.
- The reliance on humans brings in quite a few risks:
- In emergencies, there will be delay in making decisions,
- There will be inconsistency in policies across network areas,
- Configurations will be prone to errors and could possibly risk security or degrade performance.

In IoT scenarios like healthcare or relief operations, where milliseconds are the determining factor, human control is not only unproductive but might even cost lives.

11.3.3 Policy Complexity and Rule Conflicts

With the expansion of networks, administrators are confronted with a more complex set of policies. Conflicts between rules might arise as a result of:

- Simultaneous definitions of flows,
- Quality-of-service assignments that exclude each other,
- Incompatible security or ethical policies.

Standard SDN lacks a built-in mechanism for the automatic resolution of such conflicts. For instance, should the camera footage from a public park during a festival be prioritized or should the bandwidth be reserved for emergency responders? In a rules-based system, these options have to be pre-programmed and pre-defined, without any possibility of dynamic negotiation or prioritization.

11.3.4 Protocol Heterogeneity

The IoT networks consist of a multitude of devices communicating through different protocols like CoAP, MQTT, HTTP, ZigBee, BLE, etc. In such a diverse environment, manual rule setup does not work well. Flow rules usually have to take into account the protocol-specific characteristics (for example, header formats, reliability models), which adds cognitive and operational burden for network managers. To sum up, traditional rule-based SDN, though powerful in theory, cannot bear the load of changing, diverse and complicated IoT ecosystems. The limitations create an opportunity for a more self-governing and declarative IBN paradigm.

11.4 Intent-Based Networking

11.4.1 Conceptual Overview

Intent-Based Networking (IBN) represents a drastic change in network management where the system interprets and applies the intentions (high-level goals) of the administrator, application, or user autonomously without the need for manual intervention. IBN is not the same as imperative models ("forward packets from A to B on port X"). It is not hard to understand that an intent is there, it is quite a goal-oriented declaration that the system needs to work on:

- Understanding the intent
- Converting the intent into policies that can be enforced
- Determining if the network state matches the intent
- And, if not, then automatically adjusting it.

The aforementioned model is the most suitable one for IoT where the priorities of the changing devices need to be contextually adapted in a very fast manner. The Intent-Based SDN-IoT Framework's High-Level Architecture (summarized in Fig. 11.1) demonstrates the interplay between users, intelligent decision-making systems, and the IoT infrastructure that is programmable at the bottom layer. By utilizing Software-Defined Networking (SDN), this architecture intends to human-centric, goal-directed and ethics-compliant control of complex IoT environments. In the outer layer, end-users or apps state their high-level intentions or preferred actions in an understandable way, e.g. "Emergency health data to have priority" or "Traffic from certain devices to be blocked during office hours." These kinds of intentions should be understood not as low-level commands but rather as high-level expectations.

The intentions are passed on to the Intent Management Engine, which consists of three main modules:

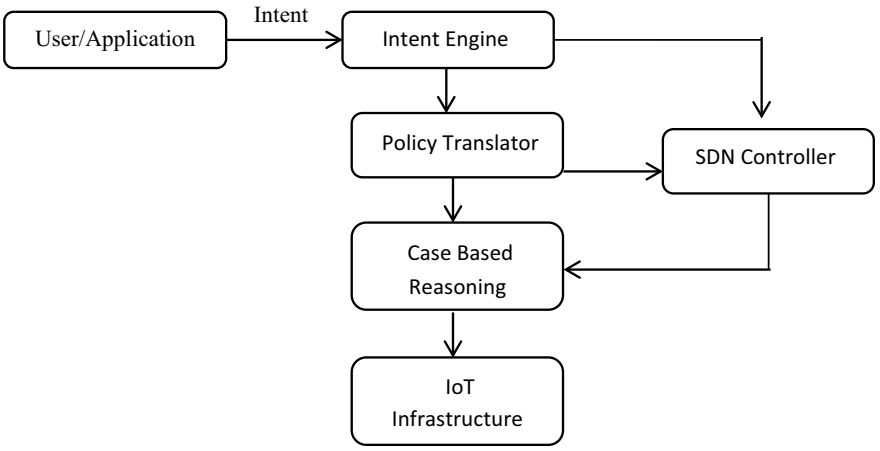


Fig. 11.1 High-level architecture of intent based SDN-IoT framework

Natural Language Processing (NLP) Interface: It decodes the human input by parsing its semantic structure and extracting the actionable intent components (subject, constraints, preferences).

Case-Based Reasoning (CBR) Module: It retrieves earlier cases (intents with similar goals and contexts) from a knowledge base, reuses the solution if it is applicable, or adapts it to the present situation. It promotes dynamic learning, contextual awareness, and reusability of previous network configurations.

Intent Translator and Conflict Resolver: It converts the intended intent into an executable policy, e.g., OpenFlow commands or northbound API requests. In case of conflicting intents (e.g., two intents requiring contradictory actions on one data flow), this module resolves based on a context.

The policy that has been translated is sent to the SDN Controller, which is the network's brain. It converses with the southbound interfaces to establish the IoT infrastructure—sensors, actuators, gateways, and fog/cloud nodes—and thus the network behavior is in sync with the user intention. A Policy Enforcement Module is present in the controller which actively sends flow rules and routing decisions to the infrastructure. It constantly monitors for feedback from the network and can trigger re-optimization using a feedback loop, which is then fed into the CBR module for continuous learning and improvement. The IoT infrastructure layer contains various devices from different domains such as healthcare, agriculture, smart cities, etc. The separating data plane and control plane— a feature of SDN— make the devices programmable and adaptable in real-time without the need for manual reconfiguration. An architecture of this kind consequently creates a responsive, intelligent, and ethically responsible network, where user intention steers network behavior instead of rigid configurations, thus making the network scalable to different IoT setups and able to cope with shifting real-world demands.

11.4.2 Life Cycle of an IBN System

An IBN-capable SDN-IoT life cycle most commonly comprises the following elements as indicated in Fig. 11.2.

11.4.2.1 Intent Engine

The interface that the user interacts with, where intents are submitted. It can be:

- A graphical one for administrators,
- A natural language one for intelligent applications,
- A machine-readable API for automated systems.

11.4.2.2 Intent Parser

Interprets the submitted intent and transforms it into a structured middle format (e.g., JSON, YAML, or domain-specific language). For instance:

Input intent:

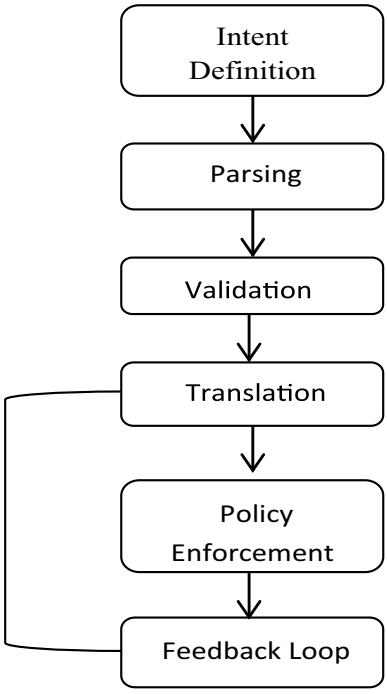


Fig. 11.2 Intent life cycle in SDN-IoT

"Ensure all patient monitors have less than 100 ms latency to the hospital cloud server."
Parsed format:
json
```
{
  "application": "health-monitoring",
  "latency_threshold": "100 ms",
  "destination": "cloud_gateway",
  "priority": "high"
}
```

11.4.2.3 Policy Translator

Converts the parsed intent into controller-specific policies and flow rules. It determines:

- Match fields (source, destination, application tags),
- Action (forward, drop, mirror),
- QoS queue assignments,
- Routing paths.

11.4.2.4 Policy Verifier

Continuously monitors network performance to check whether intents are being met. When deviations come up (e.g., the latency drifts above the threshold), the verifier invokes:

- Re-installs a rule,
- Switches a path,
- Reassigns bandwidth.

11.4.2.5 Remediation Engine

The remediation engine automatically reconfigures the network to regain the declared intent using dynamic control in SDN.

11.4.3 Benefits of IBN in SDN-IoT Environments

- **Abstraction:** The administrators of the network declare their desires while the system takes care of the implementation.
- **Agility:** The system is flexible in changing its operation according to environmental alterations, emergencies, or new priorities.

- **Scalability:** Human effort required for the management of complex rule sets is decreased to the lowest level.
- **Security and Compliance:** The intents can be supplemented with ethical or legal constraints (e.g., "no minor location data").
- **Interoperability:** Single intent schema is used across different protocols notwithstanding the differences in protocols.

11.4.4 Example of Use Cases

Table 11.2 Example of Use Cases

Intent	Realization
"Block unknown IoT devices from sending video feeds"	Flow rules match unregistered MAC/IP and drop packets
"Prioritize all environmental sensors during wildfire alerts"	SDN controller adjusts queues and allocates more bandwidth to relevant flows
"Ensure data encryption for all wearable devices"	Intent triggers TLS enforcement or encrypted tunnel setup

11.5 The IBN-SDN-IoT Triad: Towards Autonomous Policy-Driven IoT Networks

The incorporation of IBN with SDN in IoT networks results in a mutual dependency between the three different layers of network control, intents, and IoT service context. This mixture creates a smart, self-tuning environment that, without human intervention in the loop, dynamically manages connectivity, performance, and trust. The SDN controller is in charge of the translation and enforcement layer in this setup. It interprets intents that are submitted at a higher level by applications or administrators and establishes the underlying data plane accordingly. This enables a revolutionary move from imperative control logic to autonomous orchestration. The integration of IBN-SDN is goal-oriented and contextually aware, in sharp contrast to the traditional SDN implementations that involve flow rule scripting explicitly.

As an example, consider a smart hospital network. In normal circumstances, sensor traffic from the different wards gets distributed uniformly across the network. On the other hand, if a life-threatening patient is transferred or monitored from a distance, the management system will automatically create an intention like "give priority channel for ICU patient telemetry." This is done by the patient management system. The IBN engine decodes the intention, detects the related QoS requirements, and commands the SDN controller to either reroute or increase bandwidth. The

controller, having received the patient's signal that he has stabilized, will at once return the resources that were set aside, thus maximizing energy and throughput.

This mechanism guarantees a shift in IoT ecosystems from reactive to proactive, context-aware operations. The intent engine takes advantage of the learning from past scenarios, the tracking of real-time metrics, and the optimization of routing and resource utilization on the fly. Besides, such a method significantly reduces human error and latency in decision-making, which is critical in time-sensitive sectors such as industrial automation, emergency response, or smart grid control.

The architecture is also supportive of policy layering where global policies (e.g., "no video outside secure zones") and contextual intents (e.g., "stream to medical expert during tele-consultation") are resolved hierarchically. Conflicts are solved through the incorporation of prioritization logic in the intent parser and policy engine. This also distinctively removes the fragility and rigidity of conventional rule-chains that are a bane of traditional SDN deployments in IoT.

At the intent level, security and compliance are also embedded. The business can set out high-level ethics-based policies like "prevent unauthorized data exchange from wearable devices" or "only certified gateways can communicate with industrial PLCs". Continuous validation of these intents is done by monitoring agents, and the violations can trigger automatic quarantining or re-routing without the need for manual handling.

The end-to-end architecture does not merely impart intelligence to SDN it also re-characterizes IoT networks as automatically-managing systems based on goals, not commands. "What" and "how" separation imparts previously unimagined flexibility in a domain where change is the rule, not the exception.

11.6 Intent Definition and Semantic Parsing: Bridging Human Goals with Network Behavior

One of the main challenges in the development of intent-based SDN-IoT is inter-changing natural, high-level descriptions to executable network configurations. This exciting task is performed through the intent parser and semantic engine, both of them being the cognitive backbone of the system.

Human operator (via a dashboard), a higher-order application (such as an AI-driven analytics module) or even another intent (composite or nested intent) can be different sources of an intent. Each intent is declarative, expressing objectives in human or domain-specific language. For example, the following intents may be considered as examples:

- "Provide less than 200 ms latency for all autonomous vehicle feeds in zone 7."
- "Limit bandwidth use for entertainment devices within office hours."
- "Segregate any sensor indicating anomalous traffic patterns from the industrial subnet."

The semantic parser processes such intents through natural language processing (NLP) models or structured formats (YAML/JSON), depending on the interface. After parsing, the system generates an intermediate representation with the next fields:

- Usage area (e.g., automotive data transfer, industrial monitoring)
- Limitations for the system (e.g., maximum delay 200 ms)
- Mapping of the entities (e.g., area 7, device category = "self-driving car")
- Policy type (e.g., QoS, security, routing, access control)

After this intermediate representation, the Policy Translation Engine takes over and the generation of SDN rules happens. These rules might come in the form of OpenFlow entries, controller plugin invocations, or API requests to security or QoS modules that are external. The main difference with more traditional policy deployment is the goal-based abstraction—the system is not instructed in what way to reach the result, but rather, what the result is.

To ensure correctness and compliance, the policy translated is checked before the implementation. When the present network status is unable to support the intent (because it is limited in capacity or is under conflicting global policies), the engine either:

- Proposes running realizability ("latency cannot be guaranteed on existing path, rerouting through less busy node"), or
- Intents are escalated for authorization (in the situation of highly impacting requests or conflicting global policies)

Along with this, the system also acts as a translator and parser at both ends. To make it easier to understand, the system communicates the following:

Intent Status: Fulfilled.

Presently, all the sensor feeds in zone 7 are providing a latency of ≤ 185 ms.

Policy implemented through controller rte- $\times$ 1, priority queue Q2, and path [Node23 $\rightarrow$ Node45 $\rightarrow$ Core].

Such interpretability and feedback are essential in regulated environments where transparency, auditability, and trust in automation are expected. Moreover, intents can be compounded thereby creating higher-level abstractions. As an illustration, consider:

- Highest intent: "Keep operational safety for plant section A."
- Developed intents:
  - "Stop sending too much power to motor controller X."
  - "Get fire sensor group response time < 100 ms."

This facilitates the assignment of high-level business or safety goals to network operators who can then have these goals automatically broken down into network policies. Therefore, the intent layer turns out to be the new link connecting digital business goals and physical network infrastructure, and SDN orchestrating the whole process.

11.7 Conflict Resolution and Ethical Intent Enforcement

One of the most difficult problems with intent-based SDN in IoT networks is managing conflicting intents and enforcing ethical limits. As various users, applications, or subsystems emit intents—either explicitly or autonomously—the need emerges to consider and mediate among competing priorities. Unresolved conflicts can cause poor performance, unsafe choices, or even inadvertent breaches of trust and compliance.

11.7.1 Conflict Scenarios

The following are examples of conflicting intents.

- Intent A: "Allocate maximum bandwidth for industrial surveillance feed at Gate 4."
- Intent B: "Save energy consumption among all IoT nodes between 10 PM and 6 AM."

These intents can come from various stakeholders—the former from security, the latter from energy management. In an SDN-managed setup, simultaneous satisfaction may not be feasible based on bandwidth or power constraints. An intent-aware resolution mechanism needs to consider:

- Temporal urgency (Is the surveillance evoked by a real-time event?)
- Policy precedence (Does safety take precedence over energy savings?)
- Impact assessment (What's the impact of downgrading either intent?)

The intent arbitration engine, which resides in the SDN controller or orchestration plane, employs a decision matrix comprised of priority classes (emergency > real-time > routine), intent category, source credibility, and even past conflict patterns. Technical resolution is only half the aim; alignment with business and moral principles is also on the agenda.

Ethical and Regulatory Intents.

One of the key advantages of IBN-integrated SDN is its ability to natively incorporate ethical and compliance logic within the intent framework. For instance:

- "Do not send patient data through nodes which have not been verified."
- "Obey GDPR data locality regulations for all sensor logs."
- "Avoid autonomous actions in case of contradictory safety inputs."

These are not technical optimizations but non-negotiable ethical intentions. They are unlike traditional SDN policies that aim at performance and can be overridden by any QoS or throughput optimization if there is a conflict. The controller is required to:

- Validate if a suggested action breaks any ethical limitation.

- Absolutely abort the action or propose a conforming alternative.
- Log the decision and reasoning for auditability.

This is realized by incorporating intent contracts, which are immutable statements of moral limits. These contracts are used as a reference point in every stage of translation and deployment, so that even automated choices are kept in check within legal and ethical standards.

Further, contemporary intent engines may include trust scores, with IoT nodes, data sources, or users being dynamically graded. A low-trust node, whereas demanding a high-priority path, might be kept out or diverted through a sandboxed virtual network segment.

By integrating ethics as first-class citizens into the policy fabric, intent-driven SDN evolves from a reactive optimizer to a value-aligned decision-maker, allowing for trusted and responsible automation of heterogeneous IoT systems.

11.8 Practical Implementation: Lightweight Prototypes Without Full Network Simulation

The fallacy that complicated simulation frameworks or real network environments are necessary for the establishment of SDN-IBN architectures for IoT is still present. However, it is feasible to prototype the cognitive and semantic core of intent-based SDN using light-weight modular building blocks which makes this approach very suitable for conceptual, ethical, and preliminary validation research.

11.8.1 Domain-Specific Languages (DSLs) for Intent Definition

One of the most accessible ways to explore intent-based networking is to come up with a DSL that captures the high-level semantics of the goals. A simple YAML or JSON-based DSL can let users define intents such as:

```
yaml
intent:
  id: secure_data_flow
  goal: “Ensure encrypted data transfer from wearable sensors”
  constraints:
    –encryption: “AES-256”
    –destination: “certified_health_gateway”
    –max_latency: “100 ms”
```

This DSL can be interpreted with a Python-based engine that checks syntax, derives constraints, and maps them to the respective network action or API invocation (e.g., set up a virtual switch, invoke a security module, fire off a monitoring alert).

Such intent definitions can be tested in mock mode without an underlying network setup. By inputting various intents and observing parsing correctness, constraint conflict, and resolution routes, developers can assess:

- The extent to which the system is able to understand high-level policies.
- Whether conflicts are identified and ranked.
- What alternative actions are proposed in constrained scenarios.

11.8.2 Controller Logic via Microservices

Rather than employing a complete OpenFlow switch and Mininet simulation, microservices can emulate the control layer's behavior. For instance:

- A REST API service that takes parsed intents.
- A policy engine that enforces constraints and priorities.
- A rules dispatcher that "simulates" applying actions (e.g., prints the flow rule or logs it for validation).

These services can be containerized and execute on a local machine or cloud VM. This method not only accelerates development but also creates an educational tool to demonstrate how intent parsing, ethical enforcement, and arbitration are performed—without touching actual hardware or simulation tools.

11.8.3 Integration with No-Code/Low-Code Tools

To render intent systems democratized, low-code environments like Node-RED or Retool can be incorporated to:

- Enable graphical definition of intents (e.g., drag-and-drop flow for "in the event that device A breaks, re-route to device B").
- Initiate webhook-based controller responses.
- Display arbitration results and feedback reports.

This openness encourages wider experimentation, even for researchers or practitioners with minimal experience with SDN or network programming. Additionally, it makes it possible for multidisciplinary collaboration to emerge, where domain specialists (e.g., healthcare, manufacturing, law) can specify intents using natural language while the backend interprets and enforces them.

11.9 Use Case Scenarios and Case-Based Reasoning in IBN-SDN-IoT

Intent-based SDN architectures, while being a programmable underlying support for IoT, really shine in complex, multiple-objective situations where traditional rule-based control would break down. Case-Based Reasoning (CBR) is the mentioned paradigms that come into play in such scenarios: instead of starting from scratch, the system utilizes past intents, decisions, and outcomes to recommend better actions in similar situations.

11.9.1 Urban Smart Grid Management

In the smart city's distributed power grid, IoT edge sensors are monitoring power consumption, voltage dips, and possible grid failures. There is a contention between:

- Intent A: "Redirect the power supply to residential area Z because of the outage."
- Intent B: "Do not cut off power to the industrial area for the next night shift."

Using CBR, the SDN controller explores previous instances, as load-shedding situations closely similar to this one happened already. For instance, if public dissatisfaction or critical weather warnings had previously forced the system to prioritize the residence's stabilization, the system could recall that as a good result and apply that logic again on-the-fly, thus changing in real-time.

Consequently, this will not only get rid of the just-in-time recalculating of decisions, but also reduce the delay, and pave the way for a knowledge-based, self-upgrading network that can resolve complex situations without human intervention.

11.9.2 Healthcare IoT Monitoring

Imagine a hospital that uses wearable sensors on patients. At times when the hospital is busiest:

- The transmission of critical signals might be postponed due to network congestion.
- Several devices may vie for the high-priority channels.
- The IBN model processes intents such as:
- "Keep ECG data from high-risk patients uninterrupted"
- "Encrypt and log all telemetry data every 60 s"

Are handled by the SDN controller. Drawing from previous occurrences, the system is capable of recalling that directing the traffic to a reserved emergency link and logging the telemetry in a compressed form provided not only low latency but also compliance with regulations. Thus, it employs and alters that pattern.

The application of case-based methods is consequently to the benefit of the network as it goes from reactive behavior to smart reuse of policies, hence, making the system increasingly intelligent over time without the need for complete reprogramming.

11.9.3 Disaster Response and Search Missions

In connectivity zones that have become unstable because of disasters, IoT nodes, which include UAVs, sensors, and mobile gateways, are deployed very quickly. The intention is to "Keep constant link from search drone to control centre." The SDN controller consequently has to handle the following issues:

- Changing topologies,
- Node availability that is not certain, and
- Ethical limitations (for example, privacy during household scanning).

The system, relying on past emergency deployments, can easily recall successful routes, acceptable compression techniques, and predefined ethical exclusions, such as the no storage of private video streams. This experience-driven SDN-IoT provides significant benefits for applications that are time-critical and where predetermined network schedules are not applicable and real-time simulation is impossible.

11.9.4 Conclusion

The combination of Software-Defined Networking (SDN) and the Internet of Things (IoT) has been a major breakthrough in the management and control of very dynamic and diverse ecosystems by human programmers and network operators alike. However, traditional SDN architectures still face the challenge of interpreting and enforcing complex, abstract user intentions in different IoT applications. In this chapter, we introduced a new approach—Intent-Based Networking (IBN) combined with Case-Based Reasoning (CBR)—which aims to bring the SDN-IoT integration to cognitive policy flexibility. Our framework is based not on low-level, static configurations or extensive simulations, but rather on semantic understanding, experiential learning, and ethical decision-making. It equips the SDN controller to deduce high-level human intents, balance conflicting objectives across domains, and apply successful decision-making patterns in context-situated cases like healthcare, smart grids, and disaster recovery. Furthermore, by separating the dependence on simulation or real-time deployment, this approach enables lightweight prototyping, explainability, and continuous system evolution—qualities that are indispensable in areas where legal, ethical, or practical constraints are very strict.

In the long run, the combination of predictive analytics, large language models, and ethical digital twins will provide the necessary tools for intent-based SDN-IoT

infrastructures to evolve from being reactive to becoming autonomous, anticipatory, and transparent decision-makers. This change will probably alter the entire scenario of governing, optimizing, and trusting future IoT ecosystems—thus paving the way for resilient and ethically aware smart environments.

11.9.5 Future Outlook: From Reactive Policies to Cognitive Autonomy

The trajectory of SDN-IoT systems is going to change from a merely reactive policy enforcement to a fully aware system that understands, foresees, and reacts to the desires with the least involvement. Presently, the intent-based networking (IBN) area is restricted to transforming the user objectives into actionable network policies, but the future will undoubtedly surprise us with a more intelligent net where the SDN controllers will be equipped to foretell the needed actions, to derive the implied user directions, and to settle the disputes without prior permission. The most significant revolution will come from predictive intent modeling, where the controller will not have to sit and wait for pre-programmed policies but will actively create intents based on behavior modeling and environmental sensors. Suppose, for instance, there is a sudden spike in temperature and the surveillance cameras are being activated at the same time; then the system may proactively trigger a "fire risk mitigation" intent. The predictive capabilities will be complemented by temporal analytics that will spot intent patterns over time, such as traffic surges or emergency situations. Federated learning will be of prime importance, as it would facilitate privacy-preserving cross-domain learning, where machine-readable insights can be mined from the distributed IoT domains without accessing raw data. This transforms the controller into a smart partner, capable of making adaptive foresight and not merely a rule enforcer.

As the SDN infrastructures are expanding their capacities to support smart environments in all and any directions, multi-agent collaboration will be the main concern. Each SDN domain—for example, the healthcare, transport, and energy domains—will have its own agents that will be responsible for intent fulfillment, constraint negotiations, and uninterrupted service. Next-generation SDN may even have to rely on very large language models (LLMs) such as ChatGPT to further enhance the intent abstraction process. These models can take vague user goals as structured intent, explain system behavior in plain language, and gradually refine user preferences through interactive feedback loops. The combination of LLMs and SDN makes IBN a cognitive negotiation interface from the point of view of technological abstraction rather than programmability through APIs, but now it is done with language and reasoning.

The moral aspect of intent enforcement also refers to the need for intent-based digital twins that would not only mirror the real world but also anticipate how conflicting intentions would play out. These moral digital twins would be able to do counterfactuals, such as determining the outcome of the prioritization of the hospital's

emergency network over the public surveillance feed during a crisis. Furthermore, they could also consider and evaluate the ethical aspects of relaxing data localization regulations due to a cyber attack. These scenarios act as a principled sandbox for testing things in advance, thus providing not only technical resilience but also ethical responsibility in the case of SDN-IoT deployments. As the technology of cognition, automation, and intent-based policy comes together, the future of SDN-IoT is the establishment of a system where networks reason, learn, and evolve based on user values and contextual realities.

References

1. Jimmington, A., Thampi, S.M., Mukherjee, P.: Advancing connectivity and security in SDN-IoT networks through policy management. In: Securing the Connected World: Exploring Emerging Threats and Innovative Solutions, pp. 223–261. Springer Nature Switzerland, Cham (2025)
2. Khan, U.S., Mahboob, T.: Network softwarization and virtualization: management of qos in wireless and mobile networks. In: Quality of Service (QoS)-Challenges and Solutions. IntechOpen (2024)
3. Singh, B.: Software-Defined Data Centers: Innovations in Network Architecture for High Availability. Available at SSRN 5331661 (2023)
4. Aljughaiman, A., Almarri, S.: The pivotal role of software defined networks to safeguard against cyber attacks: a comprehensive review. PeerJ. Comput. Sci. **11**, e2814 (2025)
5. Sisodia, A., Vishnoi, S., Yadav, A.K.: To brace society 5.0: Enhanced reliability with a cost-effective protocol for underwater wireless sensor network. In: Sustainable Computing: Transforming Industry 4.0 to Society 5.0, pp. 171–185. Springer International Publishing, Cham (2023)
6. Biot, F.M., Fornes-Leal, A., Vaño, R., Simón, R.R., Lacalle, I., Guardiola, C., Palau, C.E.: A novel orchestrator architecture for deploying virtualized services in next-generation iot computing ecosystems. Sensors (Basel, Switzerland) **25**(3), 718 (2025)
7. Melis, A., Al Sadi, A., Berardi, D., Callegati, F., Prandini, M.: A systematic literature review of offensive and defensive security solutions with software defined network. IEEE Access **11**, 93431–93463 (2023)
8. Sisodia, A., Vishnoi, S., Upadhyay, S., Goswami, J.K., Khenwar, M., Yadav, A.K.: Enhancing energy efficiency: a protocol assessment in multi-hop mesh-based IOUT networks. Multimedia Tools Appl. **83**(37), 84999–85026 (2024)
9. Rahdari, A., Keshavarz, E., Nowroozi, E., Taheri, R., Hajizadeh, M., Mohammadi, M., ... & Bauschert, T. (2025). A survey on privacy and security in distributed cloud computing: exploring federated learning and beyond. IEEE Open J. Commun. Soc. **6**, 3710–3744 (2025). https://doi.org/10.1109/OJCOMS.2025.3560034
10. Sun, P., Wan, Y., Wu, Z., Fang, Z., Li, Q.: A survey on privacy and security issues in IoT-based environments: technologies, protection measures and future directions. Comput. Secur. **148**, 104097 (2025)
11. Sisodia, A., Yadav, A.K.: Deep dive into underwater sensor networks (UWSNs): architecture, protocols, challenges, applications, and security. In: Wireless Ad-hoc and Sensor Networks, pp. 354–378. CRC Press
12. Leivadeas, A., Falkner, M.: A survey on intent-based networking. IEEE Commun. Surv. Tut. **25**(1), 625–655 (2022)
13. Mahmud, R., Toosi, A. N., Rodriguez, M. A., Madanapalli, S. C., Sivaraman, V., Sciacca, L., ... Buyya, R.: Software-defined multi-domain tactical networks: foundations and future

directions. In Mobile Edge Computing, pp. 183–227. Springer International Publishing, Cham (2021)
14. Pokhrel, S.R.: Software defined internet of vehicles for automation and orchestration. IEEE Trans. Intell. Transp. Syst. **22**(6), 3890–3899 (2021)
15. Attebury, G., Babik, M., Campanella, M., Capone, V., Ibarra, J., Lapacz, R., ... Networking, C.: HEPiX Network Functions Virtualisation Working Group Report
16. Afolabi, I., Taleb, T., Samdanis, K., Ksentini, A., Flinck, H.: Network slicing and softwarization: a survey on principles, enabling technologies, and solutions. IEEE Commun. Surv. Tut. **20**(3), 2429–2453 (2018). https://doi.org/10.1109/COMST.2018.2815638
17. Sahoo, B.P., Mohapatra, D.P., Sahu, S.: A novel intent-based QoS-aware SDN controller using reinforcement learning. Int. J. Commun Syst **31**(14), e3745 (2018). https://doi.org/10.1002/dac.3745
18. Yoon, J., Kim, D., Lee, J.: An intent-based policy framework for network resource management. Comput. Netw. **122**, 78–95 (2017). https://doi.org/10.1016/j.comnet.2017.05.003
19. Mijumbi, R., Serrat, J., Gorricho, J.L., Bouten, N., De Turck, F., Boutaba, R.: Network function virtualization: State-of-the-art and research challenges. IEEE Commun. Surv. Tut. **18**(1), 236–262 (2016). https://doi.org/10.1109/COMST.2015.2477041

Chapter 12
SDN in Cloud Computing



Akanksha Tandon, Kiran Nagare, and Sanjeev Patel

Abstract Achieving a fair and sustainable future requires meeting Sustainable Development Goal 9 (SDG 9) through developing clean and efficient operations that assist organizations with enhancing their existing infrastructure, as well as stimulating innovation in the process. Each country will have to re-examine its own development plans based on numerous changes (digitalization, climate change, and a complicated global supply chain) that have occurred recently. This chapter illustrates the way various nations (both developed and developing) are addressing the effects of these changes at an international level. We will present an analysis of what works and what does not work through actual cases from the field. We have also created an Integrated SDG 9 Impact Model that provides a standard framework to discuss key themes relating to SDG 9 such as: innovation; resilient infrastructure; digital inclusion; and long-term sustainability. Finally, we will provide current data to identify the required investments and provide examples of successful implementation so that individuals including: investors; corporate leaders; and public policy can take action to solve the problems identified herein. The overarching goal of this chapter is to increase economic development through worldwide community development, environmental stewardship, and sustainable economic growth.

Keywords Cloud computing · Software-defined networking · Virtual machines · Computing data centers · Quality of services · Resource allocation

A. Tandon (✉) · K. Nagare
Dr. Vishwanath Karad MIT World Peace University, Pune, India
e-mail: akankshatandon97@gmail.com

K. Nagare
e-mail: kirannagare21@gmail.com

S. Patel
Department of Computer Science and Engineering, National Institute of Technology Rourkela, Rourkela, Odisha, India
e-mail: patels@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_12

12.1 Introduction

Think of cloud computing (CC) as the modern era's digital utility. It offers a shared, on-demand pool of computational capacity that you may access and release as needed, just like how you utilize energy. According to the National Institute of Standards and Technology (NIST) [1], it is a concept that provides straightforward, on-demand networking use of a collective collection of customized computing resources (e.g., networks, servers, storage, applications, and services). Giants like Google Cloud, Amazon Web Services (AWS), and Microsoft Azure rent out this digital real estate. In only a few hours, individuals and companies can set up servers, launch apps, and rent equipment with a pay-as-you-go model, only paying for what they use. NIST [1] divides cloud computing into key service models, all these service offer different control levels:

- IaaS is held on a plot of land and the raw building materials. You can deploy and manage virtual machines without worrying about the underlying physical hardware.
- Platform-as-a-Service (PaaS): Provides a different platform for developers. It provides them with an environment to build and deploy applications without the need to set up physical servers or manage core infrastructure.
- Software-as-a-Service (SaaS): Provied fully finished product, delivered directly over the internet.

These offerings can be deployed in public, private, hybrid, or community clouds, each offering a unique combination of price, adaptability, and confidentiality to meet a variety of needs. A significant part of the massive servers that run the clouds in the background is virtualized [2]. This technique significantly reduces costs and electrical consumption by enabling multiple applications to run safely on fewer physical servers. Nevertheless, management of networks is challenging with this collaborative resource strategy. Traditional routing methods, such Open Shortest Path First (OSPF) [3] and Routing Information Protocol (RIP) [4], are comparatively inflexible and static. In the dynamic cloud environment, they often lead to wasteful resource use and high operating costs.

12.1.1 Software-Defined Networking (SDN): A Solution for the Cloud

Software-Defined Networking (SDN) can help in this situation because separate control plane and data plane allow SDN to provide a flexible and agile network that may be adjusted quickly; thus, allowing a controller to be centralized and allowing networks to be modified accordingly with changes [5]. The SDN controller can create intelligent policies for routing and traffic management using OpenFlow, and thus, allows SDN to see the whole network and provide the entirety of the data center with

a virtualized network that includes hardware and software for a cloud computing environment, allowing clouds to function more effectively; thus enabling dynamic traffic control for virtual machines.

The use of Network Function Virtualization (NFV) with SDN increases the capabilities of both SDN and NFV. By combining both SDN and NFV, a new type of network emerges, called Connection Sliced Networking, which defines physical networks as multiple virtual self-contained segments that meet the needs of different applications âŁ“ this is especially important for use in a 5G network, as NFV and SDN are key enablers of providing wireless networks with specific high-performance parameters, such as response time and available bandwidth. Despite improvements in SDN and NFV, proper resource allocation with California Computer Data Center Networks (CCDCN) continues to be a major issue.

Networking segmentation, a technique that divides the entire network into simulated segments to meet diverse application requirements, is enabled by the combination of SDN with the virtualization of network functions (NFV), which provides greater flexibility. In the context of 5G networks, where SDN and NFV enable the creation of diverse services while meeting latency and capacity requirements, this capacity is particularly useful. Distribution of resources in cloud-based Cloud Data Center Networks (CCDCNs) remains a significant challenge despite these developments. QoS, SLA compliance, and energy usage are all directly affected by issues such as connection congestion, uneven network load, and ineffective routing. Additionally, erratic user demands make allocation of resources more difficult, requiring the adoption of SDN to put effective balance of load, congestion management, and forwarding regulations in place.

Think of a modern cloud network as a city's transportation system. According to a special "coding language" (the standard), every single intelligent traffic indicator (the networking switching) can interface with the centralized traffic management station (the SDN administrator). The prior approach, having often led to congestion, was comparable to allowing each intersection to make independent judgments. SDN modifies the regulations. It moves the "rule book" from the particular illumination to centralized management. In modern times, intersections just follow orders, becoming simple, effective information administrators. The framework is simplified by removing "the brains" (control) and "bones and tendons, muscular" (data movement). From a bird's-eye perspective, the centralized controller receives information about every signalized intersection, calculates the best aggregate approaches to prevent gridlock, and then sends updated directions back to each traffic light. This improves both the ability to expand the system and to deal with unanticipated increases in traffic–by making the physical infrastructure as a whole more agile, as most effectively utilizing its resources, and ultimately operating far more efficiently than the previous unconnected manner. The purpose of the Cloud Computing Model is to appropriately allocate resources such as computing power and internet connections.

Figure 12.1 illustrates this difference by showing how SDN's centralized and adaptable methodology surpasses the rigid framework of conventional networking.

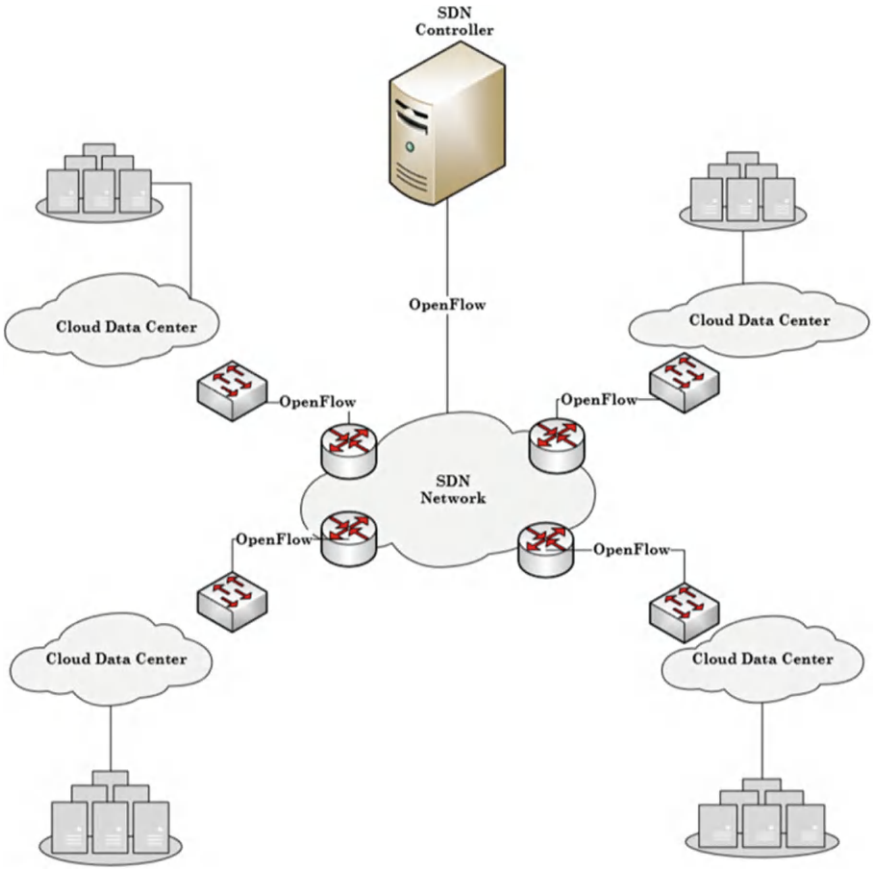


Fig. 12.1 Cloud computing based on SDN architecture

12.1.2 Key Benefits of SDN in Cloud Computing

This distinction is illustrated in Fig. 12.1 shows how SDN's centralized and flexible architecture outperforms the rigid infrastructure of traditional networking. Because SDN's greatest asset is its modularity, it acts as a universal translator between private cloud services and leading cloud administrative systems such as OpenStack and Kubernetes. Therefore, this collaboration allows leaders to concentrate on higher-level activities while completing essential yet time-consuming functions like assigning work, executing disaster recovery, and scaling infrastructure faster through automation.

SDN is also very good at creating secure virtual networks within real networks. A multi-tenancy cloud, in which multiple clients use the identical server environment, is built on this underlying structure. As noted by Zhang et al. [6], SDN serves as

an encrypted condominium construction, ensuring that every individual's "networking communication" is completely isolated from other residents while constantly upholding distinct security rules. The general stability and safety of services offered via the cloud are greatly enhanced by this. SDN also has a "green" advantage. It constantly monitors the internet connection and can automatically redistribute data over the most energy-efficient pathways or shut down underused computers, much like a thermostat with artificial intelligence for an information center. This continuous modification reduces electrical expenses and minimizes wasteful resource use.

12.2 Related Work

SDNÂ and cloud-based computing have attracted considerable attention lately as academics seek to address several of the greatest challenges posed by the advent of the cloud. Where would you build an internet connection that is easily expandable? Where could companies handle finances with greater caution and use electricity more effectively? The following part discusses the significant progress made in answering these problems. We'll look at the creative new methods researchers have proposed for three important domains: workload distribution across network components, task scheduling, and resource allocation. We'll point out each study's strengths and weaknesses, highlighting areas that still need work.

To help make understanding about this complicated subject, investigators like Banumathi et al. [7] have divided the utilization of resource techniques into three major groups: passive, active, and sophisticated. This classification offers a straightforward overview of how each approach operates and in which cases they fall short. In a similar manner, Asha et al. [8] examined these strategies in greater detail and how they affected systems running in the cloud. According to their investigations, four main factors contribute to the challenge of resource distribution: customer demand unpredictability, dynamic network topologies, changing settings, and the need to analyze large amounts of data simultaneously. Although this fundamental study does a great job of summarizing the significance, benefits, and drawbacks of each approach, it also makes it evident that there are still several significant issues that need to be resolved. Jennings et al. [9] helped simplify the discipline of resource allocation by organizing the present study into three primary categories. These fundamentally described the audience's attempts to address a number of important concerns, such as establishing portable cloud computing, simulating the financial aspects of cloud pricing, achieving genuine flexibility, administering cloud computing infrastructure from a single, unified control panel, and ensuring the predictable performance of applications.

As the monetary and ecological costs of running huge computing facilities becomes increasingly evident, current study has concentrated on identifying a remedy to the electricity consumption problem. Zomaya et al. [10] claim that minimizing energy consumption is crucial for durability and for lowering operational expenses, not just for being "green." Their work served as a comprehensive guidebook, evalu-

Table 12.1 Performance comparison between virtual machines and containers

Performance aspect	Novelty/Focus	Advantages	Disadvantages
Startup time	Time required to initialize and prepare the runtime environment	Containers start rapidly (2–5 s) as they share the host kernel and do not require full OS boot	VMs take longer (30–90 s) due to complete OS initialization, leading to higher boot latency
CPU benchmark (7-Zip)	Evaluation of computational throughput under intensive workloads	Containers achieve 95–98% of native CPU performance with minimal virtualization overhead	VMs reach only 85–90% of native performance because of hypervisor translation and context switching
Memory benchmark (STREAM)	Comparison of memory bandwidth and access latency	Containers sustain 92–95% of native bandwidth since memory access occurs directly via shared kernel	VMs achieve 85–88% of native performance due to additional address translation layers
Disk benchmark (IOzone)	Sequential and random I/O performance analysis	Containers exhibit 15–20% faster sequential read/write throughput because they bypass virtual disk emulation	VMs face reduced I/O speed caused by virtual storage abstraction and hypervisor overhead
Network benchmark (Netperf)	Latency and throughput performance over virtualized network stacks	Containers deliver 10–15% lower latency and higher throughput through streamlined network path and fewer TCP layers	VMs incur additional latency due to vNIC abstraction and extended network stack traversal
Energy efficiency	Assessment of resource utilization and idle power consumption	Containers consume 25–30% less energy owing to shared kernel processes and reduced CPU cycles	VMs are less energy-efficient due to redundant OS instances and hypervisor context switching
Ideal use case	Target deployment environments for each technology	Containers are best suited for scalable, cloud-native, microservice, and CI/CD workloads	VMs are preferred for security-intensive, compliance-driven, and legacy applications requiring full isolation

ating current energy-saving methods, identifying their flaws, and outlining a research strategy to develop more effective solutions in the cloud. Tables 12.1 and 12.2 show the summary of related work.

Table 12.2 Summary of related work

Author(s)	Novelty	Advantages	Disadvantages
Hamdy et al. [11]	Addressed parameters influencing resource allocation policies	Provided insights into policy impact on resource allocation	Lack of detailed exploration of specific resource allocation strategies
Muhammad et al. [12]	Focused on resource allocation and monitoring categorized by cloud definitions and deployment models	Identified challenges such as SLA violations and energy consumption	Limited focus on implementation and performance evaluation
Jennings et al. [9]	Organized cloud management processes into five key focus areas	Provided a global perspective on cloud management and scalability	Did not address specific resource allocation techniques or their performance in detail
Yousafzai et al. [13]	Proposed a taxonomy for optimizing resource allocation strategies	Highlighted optimization opportunities and categorized literature effectively	Limited exploration of emerging technologies and their integration into resource allocation
Alnajdi et al. [14]	Categorized dynamic resource allocation methods (use-based, market-based, SLA-based)	Highlighted strengths and limitations of dynamic strategies	Future research directions were identified but lacked in-depth solutions
Wang et al. [15]	Explored resource allocation and scheduling in data center networks	Classified algorithms based on scheduling granularity and fault tolerance	Did not fully address challenges related to fault tolerance and scalability
Akhter et al. [16]	Focused on reducing energy consumption and sustainable data center operations	Proposed efficient VM customization and cost-reduction strategies	Limited exploration of other resource allocation dimensions beyond energy
Hameed et al. [17]	Classified resource allocation challenges by policy, methods, and objectives	Comprehensive classification of resource allocation challenges and dimensions	Focused primarily on energy consumption without addressing broader cloud management challenges

12.3 Resource Management in Cloud Computing with SDN Integration

Managing resources in the cloud can be broken down into three fundamental steps: provisioning, allocation, and scheduling. You can think of it like running a highly efficient kitchen.

In reality, distributing resources is a comprehensive, organized process that includes identifying readily accessible materials, supplying them, organizing how apps will use them, and continually monitoring it all. planning and continuing administration of applications. virtual or tangible, are housed in massive information centers and utilized by many customers, comparable to a common resource. This suggests that finances ought to be regularly distributed and reallocated as necessary. An individual may need greater computing power for an intricate equation during one moment and more connectivity for an animation transmission afterwards. The end result is an adaptable architecture that avoids both extremes, which include either permitting assets lie stagnant or using them to their fullest potential. However, it is difficult to reach a precise balance, and one of the main challenges is to fairly distribute network resources in a data centre. The challenge of this issue is continuing to grow. With the proliferation of smartphones and the launch of multimedia-rich broadband applications, most notably the exponential growth of IoT, there is continuing growth in demand for unexpected levels of consumption, and we're creating more productive workers. Our information infrastructure also needs to be more intelligent and allow for more expansion. The rapidly changing environment of modern infrastructures has become too much for the outdated telecommunications model. Under that model, every gateway and connection made independent decisions. Now, at an ever-accelerating rate, computers are being built, relocated and reconfigured, creating an explosion of data that traditional technologies were not designed to handle. This is why we need an entirely new telecommunications system that is efficient, adaptable, and truly agile.

Through its powerful capabilities, Software Defined Networking (SDN) has fundamentally altered the way the market looks at the fundamental architecture needed by current cloud-computing facilities. This is done using one of SDNs basic principles (The separation of The Control Plane from The Data Plane) which enables networks to be managed by programming practices, allowing organizations to manage bandwidth, virtual machines and other critical assets without being impacted or limited by the physical architecture of their existing networks; SDN also offers the ability to manage the distribution of bandwidth, machine virtualization and other critical assets, allowing for significant reductions in time and effort on complex tasks and added layers of abstractions which provide more logical and easier to view, networked infrastructures, allowing for greater agility of distributed computing infrastructures.

Table 12.3 Resource allocation categories and percentages

Category	Percentage (%)
Bandwidth	24
Power	20
Response time	11
SLA	9
RTT	6
Packet loss	6
Throughput	6
Latency	4
Link utilization	4
Switch utilization	4
Memory	2
Cost	2

12.4 Types of Resources in Cloud Data Centers (CDCs)

CDCs (Cloud Data Centers) are the basis of cloud-based computing today, providing many different services (computation, data storage, etc.), under different models–a key component in order to provide effective, flexible and performance-driven cloud solutions and utilized as services (SaaS, PaaS, and IaaS) in multiple service categories. The types of resources available from a CDC are explained in detail below, to include a description of their role, challenges faced, and overall importance.

12.4.1 Energy Resources

Consider electricity as grows, this energy demand increases center (CDC); it is essential to the operation of the world we live in, however it additionally has a substantial cost and negative ecological impact (Jones, 2023). These organization server power based on demandable energy to run something avoid the numerous cooling techniques required to keep their equipment from overheated to the hundreds of whirling workstations. Optimization is becoming increasingly important as our dependability on online services increases. Information centers are become increasingly sophisticated to address this.

Table 12.3 demonstrates the way service level obligationsÂ essentially official agreements amongst suppliers and customers, function as a crucial structure to govern the distribution of resources in systems in the cloud. These agreements rely on certain qualitative of Service (QoS) attributes in order to measure effectiveness consistently.

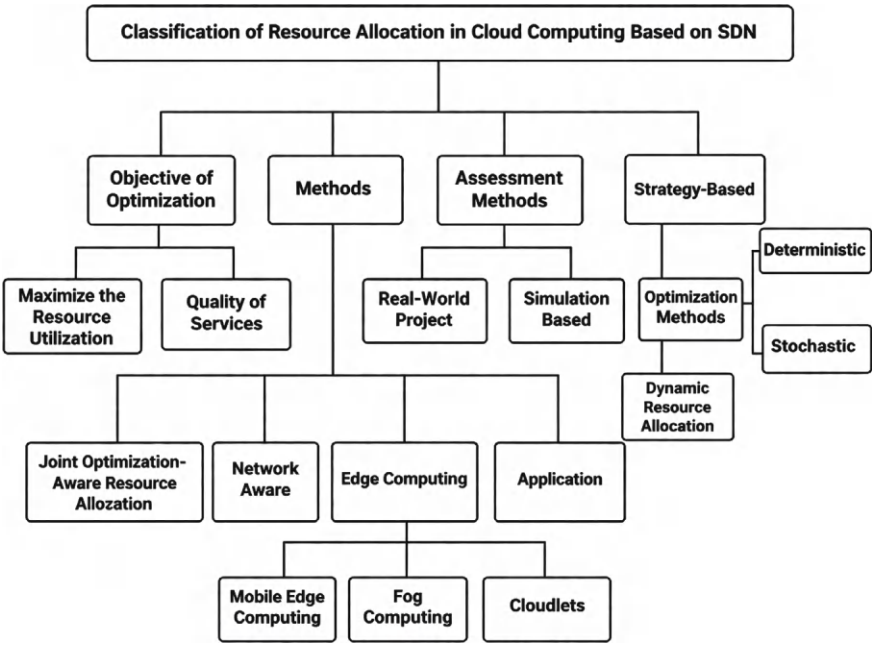


Fig. 12.2 Taxonomy of resource allocation in cloud based on SDN.drawio.png

The above variable way of ensuring it meets your demands for reliability resources procedure of modern software-generated environments.

Figure 12.2 acts as systematically understand the many approaches, objectives, and assessment strategies that SDN-enhanced data centers use ware-Defined Networking (SDN) framework. We can systematically understand the various methods, goals, and evaluation techniques that automated information centers use for successfully handling their physical assets thanks to this nomenclature. Chen and Wang (2021) state that there are three primary fundamental variations on capacity allocation, which include passive, dynamic, and a combination which combines the two. It is simpler to comprehend if resources are used in a combination strategy, assigned in a predetermined way, or changed on the fly thanks to this basic divide.

- Standardized management algorithms that distribute substances in a predetermined way are what define permanent the distribution of resources. Although with stable, predictable resource demands; however, it can lead to underutilization or overprovisioning when demand fluctuat esemand or burden in immediate terms. Although it can result in inadequate utilization or excessive provision whenever oscillations happen, stationary distribution is better suited for situations where the demands on resources are steady and anticipated.
- In contrast, dynamically utilization of resources adjusts actually in time as conditions modifications. This method continually alters resources in response to com-

munication circumstances, behavior of users, and business expectations. Adaptive deployment promotes improved adaptation, minimizes waste, and maximizes the usage of resources by responding to environmental modifications. fortunately in order to operate efficiently, complex algorithms and monitoring systems are needed.

- Hybrid Resource Allocation is combinatiuses static methods for baseline resource allocation while incorporating dynamic techniques to handle unexpected demand spikes or dropsnamic techniques to handle unexpected spikes or drops in demand. This hybrid approach balances predictability and adaptability, making it ideal for complex and fluctuating cloud environments.

Cloud administration, like running an prevent wasted resources clear strategy to guarantee ensure no assets are squandered. The major goals that guide this approach are to maximize customer satisfaction while preserving cost and environmental Cost-effectiveness suppliers. The key component is the expense of effectiveness. Every person wants to preserve assets, to put it simply. The providers of cloud services work to lower their overhead expenses in order to give their customers with affordable prices. It suggests that consumers only pay for the things they actually use. By preventing waste of resources, you may lower your costs despite sacrificing the requirements you have. It's the computerized version of turning out the illumination while you leaving a certain area. Furthermore, conserving energy has grown in significance. Think about how much energy is needed to run massive information centers. By using more clever adopting more intelligent methods to reduce power consumption not only save income on their monthly energy costs but also considerably benefit the surroundings. Everything gains from it, and the industry as everything moves closer to a more environmentally friendly future.

inevitably expenditures and energy savings are unable to achieved at the expense of effectiveness. In this particular instance, Quality of Service (QoS) is helpful. In this instant's enjoyment is the primary concern. Features like availability, rapidity, and dependability are usually guaranteed by companies in their terms of service commitments. Achieving all of these goals guarantees that your programs work effectively, the documents are constantly accessible, and your internet presence publishes swiftly, preserving individuals happy and the organization operating as guaranteed. Finally, a system that uses cloud computing needs to be expandable. Since the digital world is always changing, an internet service needs to be equipped to readily grow and shrink in alignment with requirements.

12.5 Resource Allocation in Cloud Computing Based on SDN

Figure 12.3 demonstrates the strategic equilibrium required for resource administration in a cloud context boosted by SDN. Several main streams of philosophy are depicted in the illustratconsumption and operational costs, and QoS-Based Alloca-

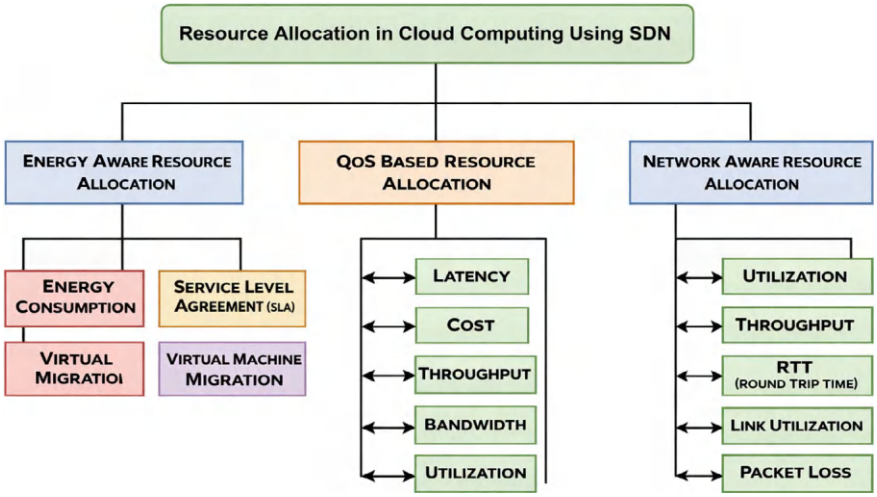


Fig. 12.3 Allocating resources in cloud computing based on SDN

tion, which focuses on guaranteeing performance, reliability, and end-user speeder, and Energy-Considering Optimization, whose seeks to maximize effectiveness for reduced electricity expenditure and costs associated with operation.

12.5.1 Energy-Aware Resource Allocation

The goal of this approach is to lower the electrical consumption neither sacrificing the dependability or functionality of the cloud infrastructure. Consider the enormous size of contemporary computing facilities. Theto trimcal consumption is not simply costly but it additionally has a substantial negative influence on the environment. We can strategically divide computation jobs in an arrangement that reduces overall electricity expenditure by putting an emphasis on reducing energy consumption. This benefits both business and the environment by reducing operational expenses and leaving a less environmental impact.

- Power Utilization: Reducing the amount of electrical power used by computer systems, networking gear, and other information technology equipment is one of the maaccounts for SLA compliance technique. To accomplish this, strategies including ecologically conscious schedules, effective distribution of duties, and adjustable capability allocation are frequently used.
- Service Level Agreement (SLA): Allocating resources in an energy-conscious manner also takes SLA compliance into account. Contenabling workload consolidation rformance that the cloud service provider must fulfill, such as accessibility

and quick responses. One major problem involves guaranteeing an energy economy while upholding SLA promises.

- In this situation, virtual machine (VM) relocation is an essential approach thatsuch as Habibi et al. [18] proposed a solution, the QRVE mechanism, to balance es, this aids in shutting downward inactive networks.

In a comparable manner, Son et al. [19] used multiple commoditto efficiently allocates the issues of minimizing SLA violations and optimizing the use of energy. Under the direction of statistical correlation analysis and real-time demand surveillance, they implemented variable overloading strategies for effective hosting and networking allocation of assets. Their strategy made it easier to put virtual machines (VMs) and consolidate traffic by utilizing virtualization and Software-Defined Networking (SDN), especially in uniform data center configurations. To improve effectiveness in SDN-compatible cloud computing environments, Lin et al. [20] introduced a complete system that integrates flow migration with virtual machine transition. Their method incorporates a Resource-and-Topology-Aware VM Migration (ETAVMM) strategy to maximize network availability and electrical consumption with traffic-relevant movement relocation using a dynamic rerouting algorithm (DENTIST) within DENDIST-FM for SDN. The effectiveness of their approach was validated using the Network Simulator (NS2) version 2.34 and CloudSim version 3.0.

Notwithstanding these developments, communication among virtual machines (VMs) is not sufficiently addressed in numerous research efforts, which mostly concentrate on VM deployment. Jin et al. [21] reinterpreted the VM deployment problem as a network improvement effort in order to close this gap in knowledge. To improve the administration of enormous virtual machines in DCNs, their approach included an automatic clustering mechanism. To find the best routes for hosting and network components and guarantee timely amalgamation, they created a first-depth best-suited scheduling mechanism. Their method was evaluated using an OpenFlow communication system combined with an adaptive fat-tree architecture in both simulated and real-world settings. In a similar vein, Yu et al. [22] developed a heuristic algorithm based on the Data Field concept to investigate VM management and interconnectivity.

Zheng et al. [6] presented the PowerNetS, a system that analyzes relationships among host demands and network activity in order in order to maximize the consolidation of servers. Their plan eliminated latency within the network, decreased power usage, and decreased cross-server traffic. Nevertheless, the investigation excluded VM deployment techniques and therefore restricted to homogeneity computing environments. The SAVE technique, that offers an SDN-assisted embedding solution for Virtual Data Centers (VDCs) and was put forward by Han et al. [23], is additional noteworthy accomplishment. SAVE combines adaptive traffic design, VDC implementation, and live virtual machine relocation predicated on the Identifying and Locating segregation Protocol (LISP). This method maximizes provider revenue while improving environmental sustainability by optimizing VDC layout of components and networking channels spanning cloud computing platefficient resource

utilization to deliver high-performance cloud services while meeting. The summary of resource allocation mechanism is depicted in Table 12.4.

12.5.2 Network Resource Allocation

In order to achieve maximum efficiency and effectiveness inside data centers located in the cloud (CDCs), allocation of network resources in cloud computing—improved by Software-Defined Networking (SDN)—is an essential operation. By integrating SDN, a customizable and centrally managed system is introduced that permits the dynamic distribution of network capacities to satisfy particular application needs, enhance Quality of Service (QoS), and reduce energy consumption.

12.5.3 Edge-Aware Resource Allocation

In edge computing systems, minimizing energy consumption and achieving Quality of Service (QoS) requirements depend on effective distribution of resources. Numerous approaches have been investigated to address these issues. Amara et al.'s workload splitting architecture [26] is one such strategy intended to handle massive applications involving data. In order to efficiently manage interconnected data center (inter-DC) conversions and guarantee seamless communication while lowering consumption of energy, the above architecture makes use of Software-Defined Networking (SDN) controls.

Under the supervision of an SDN administrator, a Metropolitan Optics Network architecture has been used to handle Infrastructure-as-a-Service (IaaS) queries. In order to find the best placement for the SDN controller's firmware while enhancing network bandwidth distribution, this framework computes significant variables. The Hierarchical Edge-Cloud SDN (HECSDN) paradigm was developed by Lin et al. [27] to address the latency problems brought on by high data flow congestion and constrained computational resources in SDN controllers. The theoretical framework was assessed using MATLAB's optimization toolbox, and the results show that it successfully improves SDN network management, which raises QoS and efficiency.

The distribution of resources algorithms in edges-cloud computers have to take into account a number of variables, such as the user's location, the wireless network connecting them to the perimeter hardware, and the link among the perimeter and the cloud.

Table 12.4 Energy-aware and QoS resource allocation mechanisms

Author	Algorithm	Advantages	Disadvantages	Parameters
Habibi et al. [18]	QRVE Mechanism with Distributed ODL Controller	Balances power saving and QoS requirements; avoids network congestion using elephant flow technique	Lacks migration techniques; applied only in emulator environments	Power consumption, SLA, Network congestion
Son et al. [19]	Dynamic Overbooking Policies and Workload Monitoring	Efficient VM placement; reduces SLA violations; integrates virtualization and SDN for traffic consolidation	Focuses on homogeneous data centers; addresses SLA and power reduction separately	SLA, Power reduction, Workload utilization
Lin et al. [20]	Unified Solution with DENTIST-FM and ETAVMM	Combines flow migration and VM migration; improves energy efficiency and network performance	Limited to simulation environments (NS2 and CloudSim); lacks scalability details for larger DCNs	Energy efficiency, Network performance
Jin et al. [21]	Clustering Mechanism with Depth-First Best-Fit Routing Algorithm	Optimizes VM placement and flow routing; fast completion of large-scale VM tasks; validated on fat-tree topology	Traffic congestion and VM migration are not addressed	Completion time, Flow consolidation
Yu et al. [22]	Heuristic Algorithm Based on Data Field	Efficient VM allocation and interconnection; provides heuristic optimization	Does not address VM migration or traffic congestion	VM allocation, Interconnection efficiency
Zheng et al. [6]	PowerNetS with Server Consolidation	Reduces cross-server traffic, power consumption, and network delays	Uses homogeneous servers; lacks VM placement considerations	Power consumption, Network delays
Han et al. [23]	VDC Embedding, Dynamic Traffic Engineering, and LISP-Based VM Migration	Optimizes VDC mappings and routing paths; reduces power consumption; increases provider revenues	Focused on heuristic algorithms; lacks broader validation across diverse configurations	VDC mapping, Power usage, Provider revenue
De et al. [24]	Mixed Integer Program (MIP) with Heuristic Techniques	Reduces latency; satisfies QoS constraints; addresses SDN-specific resource allocation challenges	Does not include VM placement or migration strategies	Latency, QoS constraints
Cziva et al. [25]	Server-Network Framework with Live VM Migration	Minimizes connectivity costs; alleviates network congestion; integrates SDN in DCNs	Limited focus on network-wide scalability; lacks detailed implementation for heterogeneous environments	Connectivity costs, Congestion reduction

12.6 Conclusion and Future Work

In the past few decades, studies examining the effective use of available assets in CDCs has gained prominence. The goal of the present study was to examine the utilization of resources and offer an elementary structure for furthering SDN-based CC investigations. The information offered here is meant to direct students studying in this field and aid in the creation and application of resource planning techniques. The deployment of resources in CC setups with SDN integration was investigated in this paper. Novel taxonomy that classify variables, methods, and strategies for optimization were revealed via a thorough investigation. These came from a thorough analysis of current methods in the literature, weighing their advantages and disadvantages. The study also examined the larger context of cloud computing, highlighting the difficulties and problems that make SDNs the best option for improving cloud.

Additionally, the study looked at the benefits and drawbacks of choosing resources techniques, offering conclusions about how well they work in various situations. The research investigation recognized unresolved issues and suggested prospective paths for improving resource allocation in SDN-enabled cloud computing environments by combining observations from both mathematical and practical viewpoints [28].

References

1. Mell, P.: The NIST definition of cloud computing. In: Recommendations of the National Institute of Standards and Technology (2011)
2. Semnanian, A.A., Pham, J., Englert, B., Wu, X.: Virtualization technology and its impact on computer hardware architecture. In: 2011 Eighth International Conference on Information Technology: New Generations, pp. 719–724. IEEE, USA (2011)
3. Sultan, N.T., Jamieson, D.D., Simpson, V.A.: Policy-based forwarding in open shortest path first (OSPF) networks. US Patent 7,831,733 (2010)
4. Open Networking Fundation: Software-defined networking: the new norm for networks. ONF white paper **2**(2–6), 11 (2012)
5. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: Openflow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
6. Zheng, K., Wang, X., Li, L., Wang, X.: Joint power optimization of data center network and servers with correlation analysis. In: IEEE INFOCOM 2014-IEEE Conference on Computer Communications, pp. 2598–2606. IEEE, USA (2014)
7. Banumathi, A., Prabakaran, M.: A survey on resource allocation policies in cloud computing environment
8. Asha,N., Rao, G.R.: A review on various resource allocation strategies in cloud computing. Int. J. Emerg. Technol. Adv. Eng. (IJETAE) **3**(7) (2013)
9. Jennings, B., Stadler, R.: Resource management in clouds: survey and research challenges. J. Netw. Syst. Manage. **23**, 567–619 (2015)
10. Zomaya, A., Balaji, P., Hameed, A., Jayaraman, P.P., Khan, S., Khoshkbarforoushha, A., Kolodziej, J., Malluhi, Q., Ranjan, R., Tziritas, N., et al.: A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems (2016)
11. Hamdy, N., Elsayed, A., ElHaggar, N., Mostafa-Sami, M.: Resource allocation strategies in cloud computing: overview. Int. J. Comput. Appl. **177**(4), 18–22 (2017)

12. Mohamaddiah, M.H., Abdullah, A., Subramaniam, S., Hussin, M.: A survey on resource allocation and monitoring in cloud computing. Int. J. Mach. Learn. Comput. **4**(1), 31 (2014)
13. Yousafzai, A., Gani, A., Noor, R.M., Sookhak, M., Talebian, H., Shiraz, M., Khan, M.K.: Cloud resource allocation schemes: review, taxonomy, and opportunities. Knowl. Inf. Syst. **50**, 347–381 (2017)
14. Alnajdi, S., Dogan, M., Al-Qahtani, E.: A survey on resource allocation in cloud computing. Int. J. Cloud Comput. Serv. Arch. (IJCCSA) **6**(5) (2016)
15. Wang, K., Zhou, Q., Guo, S., Luo, J.: Cluster frameworks for efficient scheduling and resource allocation in data center networks: a survey. IEEE Commun. Surv. Tutor. **20**(4), 3560–3580 (2018)
16. Akhter, N., Othman, M.: Energy aware resource allocation of cloud data center: review and open issues. Clust. Comput. **19**, 1163–1182 (2016)
17. Hameed, A., Khoshkbarforoushha, A., Ranjan, R., Jayaraman, P.P., Kolodziej, J., Balaji, P., Zeadally, S., Malluhi, Q.M., Tziritas, N., Vishnu, A., et al.: A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. Computing **98**, 751–774 (2016)
18. Habibi, P., Mokhtari, M., Sabaei, M.: QRVE: QOS-aware routing and energy-efficient VM placement for software-defined datacenter networks. In: 2016 8th International Symposium on Telecommunications (IST), pp. 533–539. IEEE, USA (2016)
19. Son, J., Dastjerdi, A.V., Calheiros, R.N., Buyya, R.: SLA-aware and energy-efficient dynamic overbooking in SDN-based cloud data centers. IEEE Trans. Sustain. Comput. **2**(2), 76–89 (2017)
20. Lin, W.-C., Liao, C.-H., Kuo, K.-T., Wen, C.H.: Flow-and-VM migration for optimizing throughput and energy in SDN-based cloud datacenter. In: 2013 IEEE 5th International Conference on Cloud Computing Technology and Science, vol. 1, pp. 206–211. IEEE, USA (2013)
21. Jin, H., Cheocherngngarn, T., Levy, D., Smith, A., Pan, D., Liu, J., Pissinou, N.: Joint host-network optimization for energy-efficient data center networking. In: 2013 IEEE 27th International Symposium on Parallel and Distributed Processing, pp. 623–634. IEEE, USA (2013)
22. Yu, B., Han, Y., Wen, X., Chen, X., Xu, Z.: An energy-aware algorithm for optimizing resource allocation in software defined network. In: 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–7. IEEE, USA (2016)
23. Han, Y., Li, J., Chung, J.-Y., Yoo, J.-H., Hong, J.W.-K.: Save: energy-aware virtual data center embedding and traffic engineering using SDN. In: Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), pp. 1–9. IEEE, USA (2015)
24. de Souza, F.R., Miers, C.C., Fiorese, A., de Assunção, M.D., Koslovski, G.P.: QVIA-SDN: Towards QOS-aware virtual infrastructure allocation on sdn-based clouds. J. Grid Comput. **17**, 447–472 (2019)
25. Cziva, R., Jouët, S., Stapleton, D., Tso, F.P., Pezaros, D.P.: SDN-based virtual machine management for cloud data centers. IEEE Trans. Netw. Serv. Manage. **13**(2), 212–225 (2016)
26. Amarasinghe, H., Karmouch, A.: SDN-based framework for infrastructure as a service clouds. In: 2016 IEEE 9th International Conference on Cloud Computing (CLOUD), pp. 782–789. IEEE, USA (2016)
27. Lin, F.P.-C., Tsai, Z.: Hierarchical edge-cloud SDN controller system with optimal adaptive resource allocation for load-balancing. IEEE Syst. J. **14**(1), 265–276 (2019)
28. Marcus, D.K., Zeigler-Hill, V.: A big tent of dark personality traits. Soc. Pers. Psychol. Compass **9**(8), 434–446 (2015)

Chapter 13
An Efficient Virtual Machine Migration Scheme for Load Balancing in Cloud-Fog Based IoT Systems



Sourav Kumar Bhoi and Kalyan Kumar Jena

Abstract The increasing number of requests per user in cloud-fog based Internet of Things (IoT) systems produces several problems which can be solved using load balancing mechanism. Basically, load balancing is done through load scheduling and virtual machine (VM) migration. There are many algorithms such as Shortest Job First (SJF), First Come First Serve (FCFS) etc. whose performance metrics are calculated using makespan, execution time, etc. In this work, a VM migration scheme is proposed for the balancing of load in cloud-fog based IoT systems. This work aims to produce an algorithm by focusing on adding a VM migration plan to achieve optimal performance. Based on the task length and processing power of the machine the tasks are allocated, thus no tasks have to wait longer for their turn. Hence, everyone gets fair chance and inclusion of the safety factor helps in proper load balancing in a single host as well as multi-host environment. The proposed method is compared with FCFS, SJF and MSJF methods. All these methods are compared in terms of makespan, execution time and load factor performance parameters. Results are obtained using randomly generated tasks in varied environments. The proposed method performs better as compared to FCFS, SJF and MSJF methods and it has 36, 55, 34% gains on an average in makespan, execution time and load factor performance parameters respectively.

Keywords Internet of Things · VM migration · Load balancing · Makespan · Execution time · Load factor

S. K. Bhoi (✉) · K. K. Jena
Department of Computer Science and Engineering, Parala Maharaja Engineering College, Berhampur, India
e-mail: sourav.cse@pmec.ac.in
URL: http://www.springer.com/lnc

K. K. Jena
e-mail: kalyan.cse@pmec.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_13

13.1 Introduction

Now-a-days, IoT devices play an important role in several sectors [1]. Currently, there are approximately 50 billion IoT devices available in the world wide scenario and the amount of data which is produced is very large. So, for the storage as well as processing of these data, there is the necessity of technologies like IoT, cloud computing, fog computing, etc. These technologies help in providing transient storage and processing facilities for time sensitive and low weight tasks.

With the ever-increasing demand and elasticity, cloud [2–11] becomes the part of all most all business environments. Basically, the cloud is a collection of heterogeneous hardware seen as a one whole thing. It delivers services based on the models such as IaaS, PaaS and SaaS. In this work, load balancing is done based on the IaaS model which deals with management of servers, data centers and VMs. Virtualization is behind the magic of combining multiple heterogeneous machines and abstraction of the idea of one cloud. It divides the data centers into hosts and hosts into VMs. At a particular time, when the demand is at peak, the machines can become overloaded and can cause the failure of the cloud. Also, under load can cause serious under performance of the machines. Thus, it becomes very important to balance these efficiently so as to maximize the performance and reduce failure rates. Due to more load at cloud now devices at the edge of the network provides services similar to cloud and reduces the response time and latency. This mechanism is called as fog computing [12–21] where IoT devices sends request for execution of tasks to the fog node in near proximity.

Load scheduling helps in load balancing [22–30] mechanism. Since, cloud is elastic in nature it can be scaled up and down based on the user requirements. Thus, load should be scheduled in such a manner so as to provide high availability and fault tolerance. There exist many such task scheduling algorithms such as SJF, FCFS, etc. but they are not that much effective in the resource utilization. This increases the overall execution time of the tasks. Load balancing is referred as certain set of rules and regulations for the allocation tasks to the machines. Each machine takes certain time to execute the tasks and thus reduce makespan and response time thus increasing resource utilization. But, the main focus is the parameter to use in order to describe the performance of the any load balancing technique. Many researches have been done towards it. So, the purpose of the research is to build a new approach for load balancing through task scheduling. The main contributions of this work are stated as follows.

- In this work, a load balancing approach is proposed in the IoT-cloud environment.
- The proposed load balancing approach addresses the issues of load migration in case of over load using the safety factor in a multi-tenant architecture.
- The proposed method performs better as compared with FCFS, SJF and MSJF methods in terms of makespan, execution time and load factor.
- It mainly reduces the makespan and execution time, and focuses on the improvement of load factor to much extent upto 36%, 55%, 34% respectively.

- Simulations of this work have been carried out using the python environment by the help of randomly generated tasks in different cases of the host and VMs to obtain the desired results.

In the rest sections, the work is presented as follows. Section 13.2 describes the related works. Section 13.3 discusses the methodology, Sect. 13.4 shows the results and discussion and al last Sect. 13.5 concludes the work.

13.2 Related Works

Different works have been carried out related to load balancing and scheduling [2–30]. Some of the works are described as follows. Wang et al. [2] proposed that load balancing is used to reduce the execution time for tasks and keep and every node busy and improves the load balancing algorithm which helps to increase the performance of the system. Naha et al. [3] proposed that response and processing time reduction is achieved because of the combination of both load balancing and broker techniques. Kumar et al. [4] proposed that efficient utilization of resources helps in reduction of Makespan. Banerjee et al. [5] proposed that efficient utilization of system resources reduces makespan by considering the quality of service. It gives balanced workload and lower completion time of couldlet Alworafi et al. [6] proposed a new SJF policy for cloud computing environment. Thomas et al. [7] proposed that same priority of every task can be resolved and it can help in reduce the Makespan. So, ensuring fair amount of priority for every task enabling equal resource allocation. Jindal et al. [8] focused on optimizing task scheduling using QoS metrics in cloud environment. Thus, improving the response time of tasks and improving Quality of service (QoS). Semmoud et al. [9] proposed that when there is increase in utilization rate of servers then the response time is decreasing, and the overall migration cost is reduced and the stable system is maintained. Polepally et al. [10] proposed that the decrease in number of task migration results in smooth CPU utilization and increase in performance of the system. Lakra et al. [11] proposed that reduction in the wastage of resources give improvement throughout the data center and reduce the violation in SLA. Xu et al. [12] proposed that it is difficult to analyze load balancing for the computation of nodes in Fog environment. For this challenge, DRAM is the dynamic resource allocation method has been used in the Fog environment. Yi et al. [13] focuses on the Fog computing and their similar concepts, and provides different application scenarios and analyzes various issues which are encountered when Fog computing is designing and implementing for computing systems. Yi et al. [14] focused on the Fog computing challenges and goals. Here the attempt has been made for the implementation and evaluation in the Fog computing platform. Stojmenovic et al. [15] focused on the motivation and advantages of Fog computing and analyze its application in the real scenario. Iorga et al. [16] focused on the Fog and mist computing and their connection in cloud based computing models for IoT. Atlam et al. [17] proposed that fog is the better platform for the IoT services and applications. Chen et al. [18] focused on the

issues that have been explored for computing opportunities, storage, networking and the mechanism for reducing the latency and enhancing the security. Bellendorf et al. [19] focused on the taxonomy of different optimization problems in Fog computing. Many such algorithms are presented in [20–30].

From the above works, it is found that many researchers have proposed different techniques for load balancing, VM migration, resource allocation, and task scheduling in cloud and fog environments. However, most of these studies focus either on improving resource utilization or reducing makespan, without jointly considering VM migration and load balancing. Very few works consider the dynamic nature of IoT workloads, where sudden changes in device activity can quickly overload fog nodes and degrade system performance. Therefore, there is still a need for an efficient and lightweight VM migration scheme that can balance the load between cloud and fog layers by maintaining low latency. The present work proposes an improved VM migration approach aimed at showing better load distribution and enhanced overall efficiency in cloud-fog based IoT systems.

13.3 Methodology

13.3.1 Problem Statement

Consider a data center with M (m1,,m2,...,mn) hosts having N (n1,n2,...nn) number of VMs each and at time t=k seconds arrives X (t1,t2,...,tn) number of tasks with y task length. Schedule, allocate resources to these tasks in such a way so as to maximize the performance of the cloud in terms of makespan, execution time and resource utilization with the constraints mentioned below. The cloud should be balanced, not overloaded or under loaded that means evenly distributed load to the nodes.

13.3.2 Network Model

In this work, a 3-layer network model is focused. This model is described in Fig. 13.1. The 3 layers are cloud layer, Fog layer and terminal nodes layer. The cloud layer can be used for performing required operations in cloud platform. It can be connected to the Fog layer with the help of Wi-Fi connectivity. The Fog layer is next to the terminal nodes layer and it is a complimentary layer to the cloud not the replacement. In the terminal nodes layer, huge amount of data is generated from wide variety of sensors from a wide variety of machines. Since the IoT devices produce huge amount of data, sending the raw data to the cloud directly makes little sense as it will require huge bandwidth and will have latency issues with the amount of data that is being generated over the time. To solve this issue, Fog layer is used. Fog nodes are the virtualized components that can be routers, embedded servers, switches etc. that

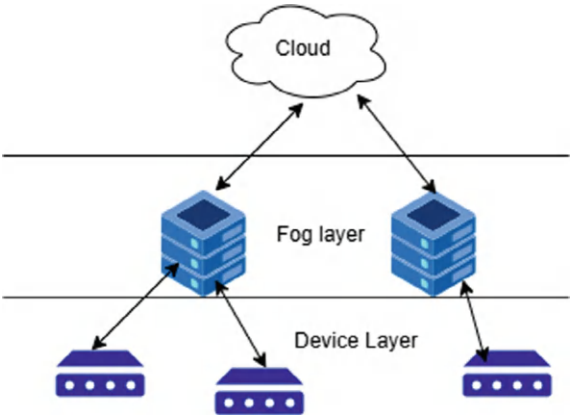


Fig. 13.1 Network model

provides a transient storage and processing for tasks enabling low operational costs, better security and also pre-processing of data for faster processing at the cloud. The data is sent using Wi-Fi technology. The low intensive and very time sensitive data can be processed in the Fog nodes and it helps in providing faster results in critical situations. However, the less time sensitive and heavy task are sent to the cloud for processing and the results are gathered after it is processed. In the Fog layer, Fog node coordinators are available to carry out Fog node coordination process. In this work, the Fog layer deals with Fog nodes sub layer with can interact with terminal nodes layer for several operations through Wi-Fi connectivity.

13.3.3 Proposed Load Balancing Scheme

The proposed load balancing model is described as follows.

13.3.3.1 Task Offloading

The IoT and edge devices generate data for processing and are at low intensive workload is processed in the Fog nodes and the highly intensive and middle level work are processed with the help of the cloud. The work load sent by the Fog nodes is served by the cloud broker. The working of the algorithm is black boxed into an entity and the clients across the world send their request to the cloud broker which then sends this to the load balancer that manages the resources in the backend consisting of multiple hosts with multiple VMs. As a result, the workload is evenly distributing across multiple VMs and hosts in the cloud environment. In the cloud, the tasks are kept in a queue and follow the M/M/1 queuing model for its smooth serving by the entity called cloud broker which acts as the server that means allocating the

required resources to the tasks and sending back the results. Here, the arrival rate of the tasks follows the Poisson's distribution λ and the server follows an exponential distribution μ. The number of tasks in the system is denoted by n and Pn denotes the probability of n customers in the system in steady state that is what is required for quicker allocation of the tasks. In the steady state, the arrival rate is equal to departure rate. The mathematical implications are mentioned in Eq. 13.1 to 13.3.

$$\mu P_1 = \lambda P_0 \tag{13.1}$$

$$\lambda P_0 + \mu P_2 = (\lambda + \mu) P_1 \tag{13.2}$$

$$\lambda P_{n-1} + \mu P_{n+1} = (\lambda + \mu) P_n \tag{13.3}$$

The steady state of the queue provides equilibrium for the arrival of the tasks from the fog nodes and the serving by the cloud broker in the data center. The policy for the segregation of the tasks for processing in the cloud is described in Algorithm 1.

Algorithm 1 Algorithm for Task Segregation

Require: Task Segregation(Task Size, Priority)
Ensure: α bytes for task
1: Start
2: Requests from IoT devices are sent to the nearest Fog nodes
3: **if** Task Size > γ% of α limit **and** Priority = Low **then**
4: Pass the task to the Cloud
5: **else**
6: Process the task at the Fog nodes
7: **end if**
8: End

13.3.3.2 Load Balancing

In the cloud, the proposed algorithm works when the tasks are handled over to the load balancer which then distributes over the physical machines for processing. The cloud architecture for load balancing is mentioned in Fig. 13.2. Firstly, it manages the allocation of the workload to a host among hosts to the available VMs based on the safety factor, violation of which will result in the migration of the tasks to the available hosts or postpone them through the ready queue. At time t, when the load balancer is flooded with tasks it first checks the host capacity based on its constituents VMs, refactor them to adjust to the safety factor and migrate the loads in case of a violation. If there is not enough capacity among all the hosts in the data center all the tasks are queued and will be executed later. The safety factor should be greater

than or equal to the threshold value δ as mentioned in Eq. 13.4.

$$\text{Safety factor} \geq \Delta\% \tag{13.4}$$

Secondly, it works to ensure a fair amount of priority to every task whether be it is the longest or the shortest task. The SJF is a pretty much good algorithm allocates shortest jobs to the machines first. Thus, it ensures better response time and execution of the shortest jobs but the longest jobs can have to wait for indefinite time. Thus, a modified version of the SJF [22] can be used where the length of the tasks and the processing power of the VMs in the hosts are sorted and their average is calculated using equation (13.5) and grouped accordingly that means the tasks with lower than average will be processed by machine having processing power lower than the average. That means, G1 and the tasks with greater than average length will be processed by the machine higher than the average that means G2. In this way, the makespan is highly optimized and the loads are evenly distributed among the VMs. But one demerit is that any shortest job can be assigned to the highest machine. To address this issue, a priority is given in an order that is they are sorted in a descending order so that the longest task is assigned to the machine with the highest processing power and the shortest task is assigned to the machine with the lowest processing power so as to maximize the performance. In equation (13.5), x_n is the processing power of a VM in host and n is the number of VMs.

$$\text{Average} = \frac{x_1 + x_2 + \cdots + x_n}{n} \tag{13.5}$$

At last, deals with the execution of the tasks based on the above policies in their respective VMs and hosts, and finally the makespan, execution time and resource utilization are calculated. Makespan represents the total time to schedule a cloudlet, execution time is the exact time to complete the tasks and resource utilization is the utilization of resources in performing the required operations in this scenario. The

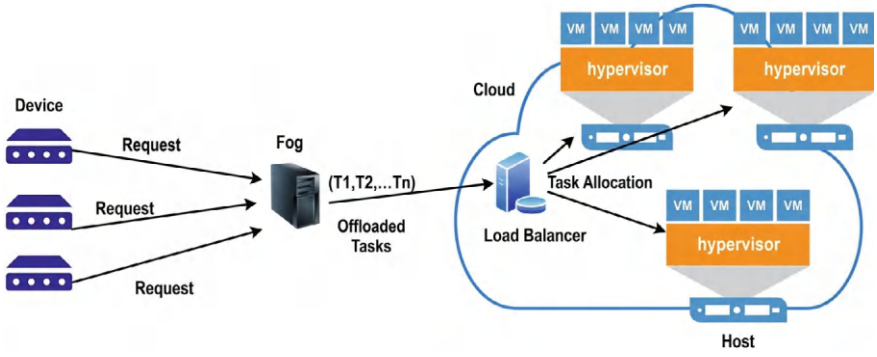


Fig. 13.2 Load balancing

workflow diagram of the proposed work is mentioned in Fig. 13.3 and the proposed algorithm is mentioned in Algorithm 2.

Algorithm 2 Proposed Work for Load Balancing

Require: V = list of VM MIPS values; T = list of task lengths; H = host capacity
Ensure: Makespan, Execution Time, Resource Utilisation
1: **if** $H \leq 85\% - 90\%$ **then**
2: **while** $H \leq 85\%$ **do**
3: **(a)** Compute

$$avg_vm_mips = \frac{\sum V}{|V|}$$

4: **(b)** Compute

$$avg_task_length = \frac{\sum T}{|T|}$$

5: **(c)** For each VM i in V:
6: **if** $V[i] > avg_vm_mips$ **then**
7: Put $V[i]$ into $G1$
8: **else**
9: Put $V[i]$ into $G2$
10: **end if**
11: **(d)** For each task i in T:
12: **if** $t_length[i] > avg_task_length$ **then**
13: Put $T[i]$ into $T1$
14: **else**
15: Put $T[i]$ into $T2$
16: **end if**
17: **(e)** Apply descending sort on $G1$, $G2$, $T1$, and $T2$
18: **(f)** Allocate tasks:
19: $T1 \rightarrow G1$
20: $T2 \rightarrow G2$
21: **end while**
22: **end if**
23: **3.** If $H > 85\%$:
24: Migrate workload
25: Repeat algorithm from step (a)
26: Compute average Makespan for each VM:

$$avg_makespan = \frac{CT(last\ task) - AT(first\ task)}{|V|}$$

27: Compute Average Execution Time
28: Compute Resource Utilisation:

$$Utilisation = \left(\frac{Total\ busy\ time}{Total\ available\ time} \right) \times 100$$

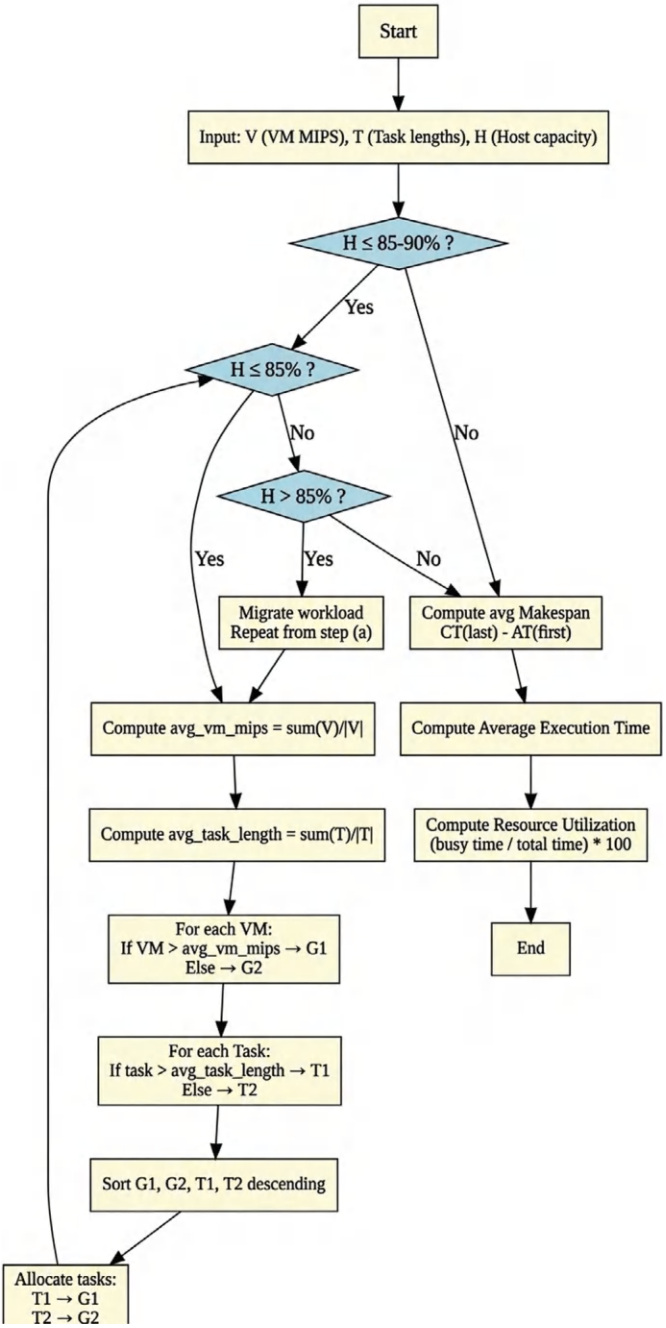


Fig. 13.3 Flowchart for load balancing

Table 13.1 Homogeneous host configuration

Host parameter	Values
No. of hosts	1–3 (10000 MIPS each)
No. of VMs	4, 6, 8
No. of tasks	20, 40, 60, 80, 100

Table 13.2 Heterogeneous host configuration

Host parameter	Values
No. of hosts	2, 3 (10000, 15000, 20000 MIPS)
No. of VMs	4, 6, 8
No. of tasks	20, 40, 60, 80, 100

Table 13.3 Makespan comparison for homogeneous hosts with reference to Fig. 13.4

Method	Fig 13.4a	Fig 13.4b	Fig 13.4c	Fig 13.4d	Fig 13.4e	Fig 13.4f	Fig 13.4g	Fig 13.4h	Fig 13.4i
FCFS	182.3	122.3	133.4	58.06	35.09	38.7	29.25	17.1	18
SJF	27	80	87.8	39.2	23.5	28.64	18.5	11.2	12
MSJF	23	73.5	68.07	32.6	22.8	21.8	13.6	11.1	11.5
Proposed method	21.3	67.8	52.15	29.1	17.8	15.5	11.8	8.4	8.5

13.4 Results and Discussion

The proposed work is implemented using the Python programming language and its facilitating tools and libraries such as Jupyter Notebooks, random library and matplotlib. This work has been carried out in a Hewlett-Packard (HP) G7 notebook PC with Intel i3-1005G1 processor 2 cores and 4 logical processors having 4GB of RAM. The notebook run 64-bit version of Windows 10. For performing this work Python distribution of Anaconda was used. The proposed load balancing algorithm has been compared with the algorithms such as FCFS, SJF and MSJF [22] in terms of makespan, execution time and load factor. The simulation is done for homogeneous and heterogeneous hosts with a particular number of VMs run for a wide range of random tasks upper limit to 20000 bytes over 100 tasks stepped 20 tasks to the mark. The homogeneous and heterogeneous configuration is mentioned in Tables 13.1 and 13.2 respectively.

Figure 13.4 and Table 13.3 describe the makespan results for homogeneous hosts. Here, Fig. 13.4 depicts the results of evaluation of makespan for combination of 1,2,3 homogenous host with 10000 MIPS each having been divided into 4,6,8 number of VMs each at once over the time run over a range over 100 randomly generated tasks of length upper limited at 20000 bytes. It can be clearly seen from the graphs that

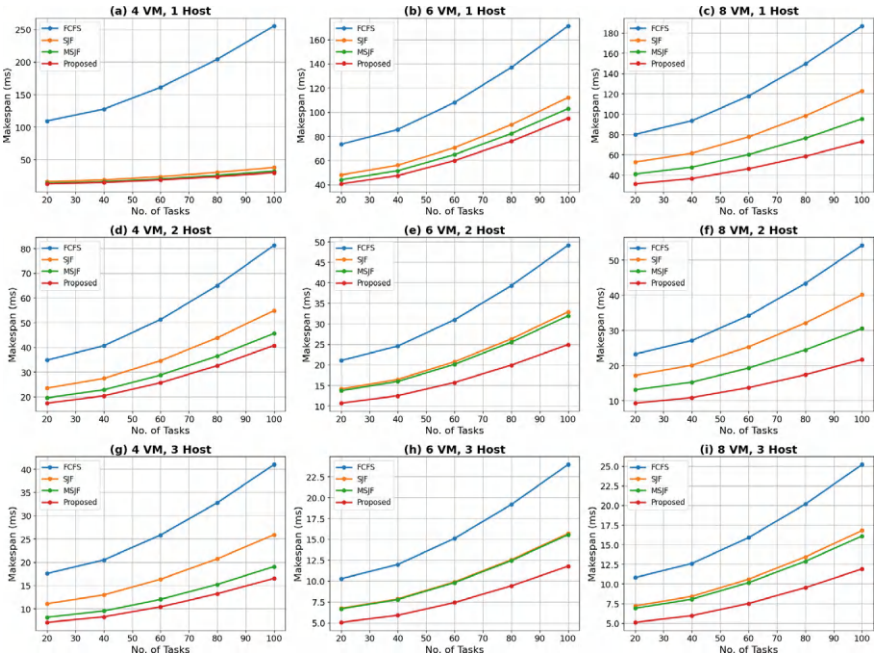


Fig. 13.4 Comparison of makespan for homogeneous hosts.

the proposed algorithm performs better than its peers in all the scenarios, the lower the make span the better the performance. Arranging tasks and VMs in descending order provides a fair priority thus this improvement in the result. For Fig. 13.4a, for high load and limited resource that is 100 tasks it can be seen 182.3, 27.1, 23.2, 21.3 ms are the makespan for FCFS, SJF , MSJF, Proposed method for 1 host which is divided into 4 VMs. Making a delightful choice for balancing load in a multi-tenant distributed cloud. Similarly, in Fig. 13.4e, for high load and medium resources that is 6 VM , 2 host configuration, it shows 122.3, 80, 73.5, 67.8 ms respectively makespan thus, achieving supremacy over others. In the similar manner, in Fig. 13.4i, for high load and high availability of resources 8VM, 3 host configuration, it achieves 17.1, 11.2, 11.1, 8.4 ms makespan respectively thus making it the better one. At last, it can be seen that it outperforms all the compared algorithms in the last case on average by 35%.

Figure 13.5 and Table 13.4 describe the execution time scenarios for homogeneous hosts. Here, Fig. 13.5 depicts the results of evaluation of execution time for combination of 1,2,3 homogeneous host with 10000 MIPS each having been divided into 4,6,8 number of VMs each at once over the time run over a range over 100 randomly generated tasks of length upper limited at 20000 bytes. It can be clearly seen from the graphs that the proposed algorithm performs better than its peers in all the scenarios, the lower the execution time the better the performance. Arranging tasks and VMs in descending order provides a fair priority thus this improvement in the result. For

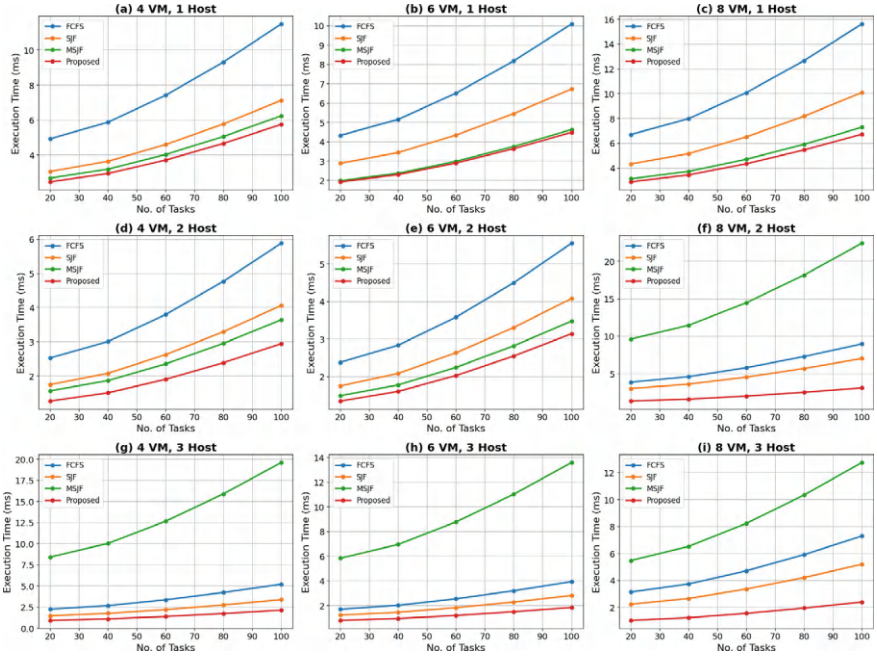


Fig. 13.5 Execution time comparison for homogeneous hosts

Table 13.4 Execution time for homogeneous hosts with reference to Fig. 13.5

Method	Fig 13.5a	Fig 13.5b	Fig 13.5c	Fig 13.5d	Fig 13.5e	Fig 13.5f	Fig 13.5g	Fig 13.5h	Fig 13.5i
FCFS	8.2	7.2	11.15	4.2	3.96	6.4	3.7	2.8	5.2
SJF	5.08	4.8	7.2	2.9	2.91	5	2.4	2	3.7
MSJF	4.45	3.3	5.2	2.6	2.48	16	14	9.7	9.1
Proposed method	4.1	3.2	4.8	2.1	2.24	2.2	1.5	1.3	1.7

Fig. 13.5a, for high load and limited resource that is 100 tasks it can be seen 8.02, 5.08, 4.45, 4.1 ms are the execution time for FCFS, SJF, MSJF, proposed method for 1 host which is divided into 4 VMs. Making a delightful choice for balancing load in a multi-tenant distributed cloud. Similarly, in Fig. 13.5e, for high load and medium resources that is 6 VM, 2 host configuration , it shows 3.96, 2.91, 2.48, 2.24 ms respectively execution time thus, achieving supremacy over others. In the same manner, in Fig. 13.5i, for high load and high availability of resources 8VM, 3 host configuration, it achieves 5.2, 3.7, 9.1, 1.7 ms execution time respectively thus making it the better one. At last, it can be seen that it beats all the compared algorithms in the last case on average by 65%.

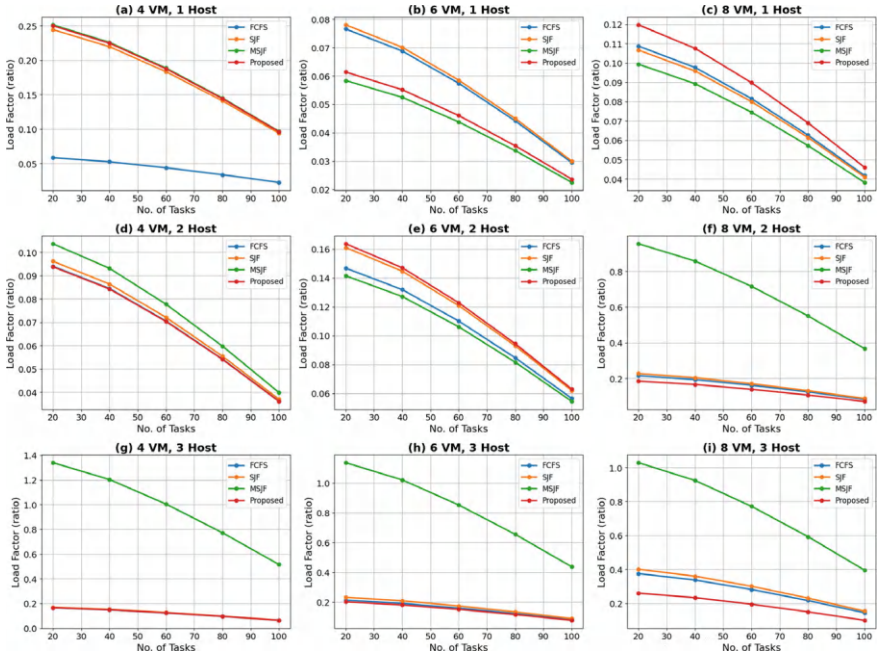


Fig. 13.6 Comparison of load factor for homogeneous hosts

Table 13.5 Load factor for homogeneous hosts with reference to Fig. 13.6

Method	Fig 13.6a	Fig 13.6b	Fig 13.6c	Fig 13.6d	Fig 13.6e	Fig 13.6f	Fig 13.6g	Fig 13.6h	Fig 13.6i
FCFS	0.044981	0.058872	0.083583	0.072339	0.112853	0.165375	0.126496	0.163743	0.288889
SJF	0.188148	0.060000	0.082005	0.073980	0.123830	0.174581	0.129730	0.178571	0.308333
MSJF	0.193478	0.044898	0.076392	0.079755	0.108772	0.733945	1.029412	0.873874	0.791304
Proposed Method	0.192488	0.047198	0.092042	0.072165	0.125843	0.141935	0.127119	0.154762	0.200000

Figure 13.6 and Table 13.5 describe the load factor scenarios for homogeneous hosts. Here, Fig. 13.6 depicts the results of evaluation of load factor for combination of 1,2,3 homogeneous host with 10000 MIPS each having been divided into 4,6,8 number of VMs each at once over the time run over a range over 100 randomly generated tasks of length upper limited at 20000 bytes. It can be clearly seen from the graphs that the proposed algorithm performs better than its peers in all the scenarios, the lower the load factor the better the performance, that means how well the load is distributed or with how much less resource the work can be done. Arranging tasks and VMs in descending order provides a fair priority thus this improvement in the result. For Fig. 13.6a, for high load and limited resource that is 100 tasks it can be seen 0.03,0.04,0.037,0.035 are the load factor for FCFS, SJF, MSJF, Proposed method for 1 host which is divided into 4 VMs. Making a delightful choice

Table 13.6 Makespan for heterogeneous hosts with reference to Fig. 13.7

Method	Fig 13.7a	Fig 13.7b	Fig 13.7c	Fig 13.7d	Fig 13.7e	Fig 13.7f
FCFS	22.2	32.9	36.7	20.0	14.0	13.8
SJF	16.5	21.0	24.6	12.1	9.0	8.7
MSJF	9.7	20.0	15.6	10.2	7.0	6.3
Proposed method	9.2	15.1	12.3	7.5	6.7	5.6

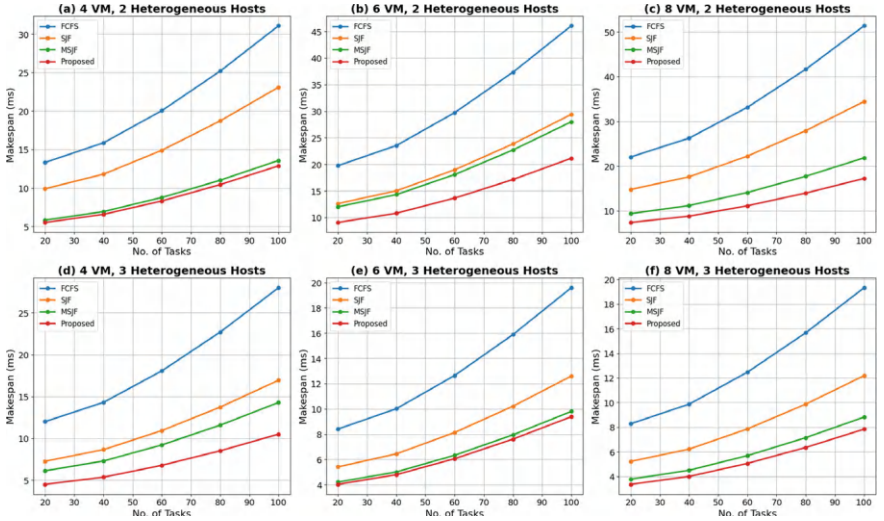


Fig. 13.7 Makespan comparison for heterogeneous host configurations

for balancing load in a multi- tenant distributed cloud. Similarly, in Fig. 13.6e, for high load and medium resources that means 6 VM, 2 Host configuration, it shows 0.11,0.12,0.11,0.10 respectively load factor thus, achieving supremacy over others. In the similar manner, in Fig. 13.7i, for high load and high availability of resources 8VM, 3 host configuration, it achieves 0.23, 0.220.93, 0.18 load factor, respectively thus making it the better one. At last, it can be seen that it beats all the compared algorithms in the last case on average by 20%.

Figure 13.7 and Table 13.6 describe the makespan scenarios for heterogeneous hosts. Here, Fig. 13.7 depicts the results of evaluation of makespan for combination of 2, 3 heterogeneous host with 10000 MIPS and 15000MIPS combination and with 10000MIPS, 15000 MIPS, 20000 MIPS configuration having been divided into 4,6,8 no of VMs each at once over the time run over a range over 100 randomly generated tasks of length upper limited at 20000 bytes. It can be clearly seen from the graphs that the proposed algorithm performs better than its peers in all the scenarios, the lower the make span the better the performance. Arranging tasks and VMs in descending order provides a fair priority thus this improvement in the result. For Fig. 13.7a, for

Table 13.7 Execution time for heterogeneous hosts with reference to Fig. 13.8

Method	Fig 13.8a	Fig 13.8b	Fig 13.8c	Fig 13.8d	Fig 13.8e	Fig 13.8f
FCFS	3.0	3.6	5.4	2.0	2.5	3.1
SJF	2.5	2.5	4.1	1.6	1.8	2.1
MSJF	24.0	20.0	16.0	8.8	7.5	5.7
Proposed method	1.8	1.5	2.1	1.0	1.1	1.5

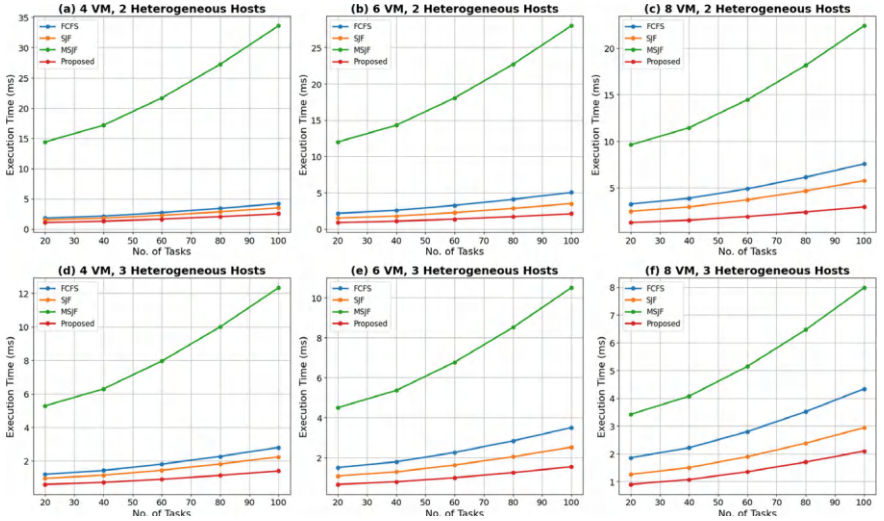


Fig. 13.8 Execution time comparison for heterogeneous host configurations

high load and limited resource that is 100 tasks it can be seen 22.2, 16.5, 9.7, 9.2 ms are the makespan for FCFS, SJF ,MSJF, Proposed method for 2 host which is divided into 4 VMs. Making a delightful choice for balancing load in a multi-tenant distributed cloud. Similarly, in Fig. 13.7c, for high load and medium resources that is 8VM , 2 Host configuration, it shows 36.7, 24.6, 15.6, 12.3 ms respectively makespan thus, achieving supremacy over others. In the similar manner in Fig. 13.7f, for high load and high availability of resources 8VM, 3 Host configuration, it achieves 13.8, 8.7, 6.3, 5.6 ms makespan respectively thus making it the better one. At last, it can be seen that it beats all the compared algorithms in the last case on average by 40%.

Figure 13.8 and Table 13.7 describe the execution time scenarios for heterogeneous hosts. Here, Fig. 13.8 depicts the results of evaluation of execution time for combination of 2,3 heterogeneous host with 10000 MIPS and 15000 MIPS combination and with 10000MIPS, 15000 MIPS, 20000 MIPS configuration having been divided into 4,6,8 number of VMs each at once over the time run over a range over 100 randomly generated tasks of length upper limited at 20000 bytes. It can be clearly seen from the graphs that the proposed algorithm performs better than its peers in

Table 13.8 Load factor for heterogeneous hosts with reference to Fig. 13.9

Method	Fig 13.9a	Fig 13.9b	Fig 13.9c	Fig 13.9d	Fig 13.9e	Fig 13.9f
FCFS	0.135	0.109	0.147	0.100	0.178	0.224
SJF	0.151	0.119	0.166	0.132	0.200	0.240
MSJF	2.700	1.000	1.020	0.860	0.930	0.900
Proposed method	0.195	0.099	0.170	0.133	0.164	0.267

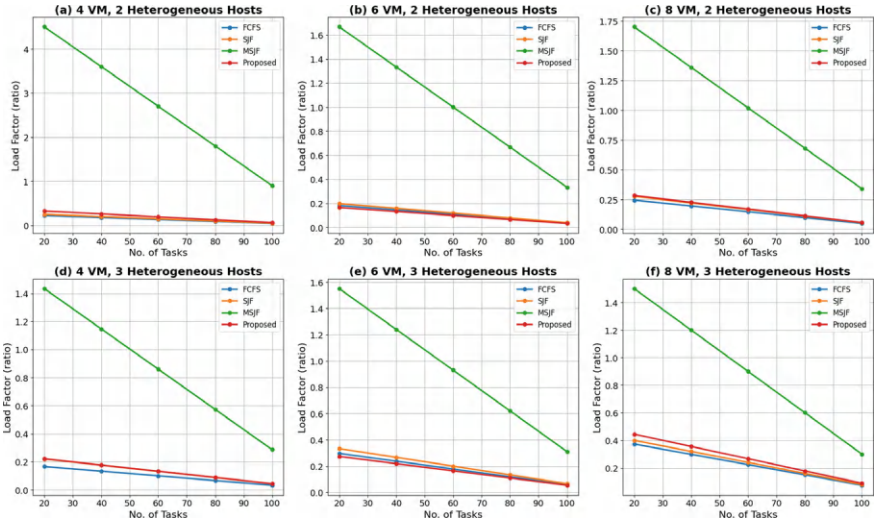


Fig. 13.9 Load factor comparison for heterogeneous host configurations

all the scenarios, the lower the execution time the better the performance. Arranging tasks and VMs in descending order provides a fair priority thus this improvement in the result. For Fig. 13.8a, for high load and limited resource that is 100 tasks it can be seen 8.3, 2.5, 24, 1.8 ms are the execution time for FCFS, SJF , MSJF, Proposed method for 2 host which is divided into 4 VMs. Making a delightful choice for balancing load in a multi-tenant distributed cloud. Similarly, in Fig. 13.8c, for high load and medium resources that is 8VM, 2 host configuration, it shows 5.4, 4.1, 16, 2.1 ms respectively makespan thus, achieving supremacy over others. In the similar manner, in Fig. 13.8f, for high load and high availability of resources 8VM, 3 host configuration, it achieves 3.1, 2.1, 5.6, 1.5 ms execution time respectively thus making it the champion by approximately 50%.

Figure 13.9 and Table 13.8 describe the load factor scenarios for heterogeneous hosts. Here, Fig. 13.9 depicts the results of evaluation of load factor for combination of 2,3 heterogeneous host with 10000 MIPS and 15000MIPS combination and with 10000MIPS, 15000 MIPS, 20000 MIPS configuration having been divided into 4,6,8 number of VMs each at once over the time run over a range over 100 randomly

generated tasks of length upper limited at 20000 bytes. It can be clearly seen from the graphs that the proposed algorithm performs better than its peers in all the scenarios, the lower the load factor the better the performance, i.e. how well the load is distributed or with how much less resource the work can be done. Arranging tasks and VMs in descending order provides a fair priority thus this improvement in the result. For Fig. 13.9a, for high load and limited resource that is 100 tasks it can be seen 0.73,0.73,0.79,0.71 are the load factor for FCFS, SJF, MSJF, Proposed method for 2 host which is divided into 4 VMs making a delightful choice for balancing load in a multi-tenant distributed cloud. Similarly, in Fig. 13.9c, for high load and medium resources that 8VM , 2 host configuration, it shows 0.14,0.17,1.02,0.17 load factor respectively thus, achieving supremacy over others. In the similar manner in Fig. 13.9f, for high load and high availability of resources 8VM, 3 host configuration, it achieves 0.22,0.21,0.90,0.24 load factor respectively thus making it the better choice by approx. 45%.

13.5 Conclusion

In this work, a load balancing method is proposed in the IoT-cloud environment. This work addresses the issues of load migration in case of over load using the safety factor in a multi-tenant architecture. The proposed method is compared with FCFS, SJF and MSJF methods in terms of makespan, execution time and load factor performance parameters. This work mainly reduces the makespan and execution time, and focuses on the improvement of load factor to much extent upto 36, 55, 34% respectively pegged over 100 randomly chosen tasks. From the results, it is observed that the proposed work performs better in balancing the load in several scenarios as compared to FCFS, SJF and MSJF methods. This work can be extended to compare the proposed method with other methods by considering different performance parameters. Also, the scalability can be studied by including more extensive simulations on larger benchmark datasets. Further, this method can be run on a real hardware for real time simulation.

References

1. Atzori, L., Iera, A., Morabito, G.: The Internet of Things: a survey. Comput. Netw. **54**(15), 2787–2805 (2010)
2. Wang, S.-C., Yan, K.-Q., Liao, W.-P., Wang, S.-S.: Towards a load balancing in a three-level cloud computing network. In: 3rd International Conference on Computer Science and Information Technology (ICCSIT), pp. 108–113 (2010). https://doi.org/10.1109/ICCSIT.2010.5563889
3. Naha, R.K., Othman, M.: Cost-aware service brokering and performance-sentient load balancing algorithms in the cloud. J. Netw. Comput. Appl. **75**, 47–57 (2016). https://doi.org/10.1016/j.jnca.2016.08.018

4. Kumar, M., Sharma, S.C.: Dynamic load balancing algorithm to minimize the makespan time and utilize the resources effectively in cloud environment. Int. J. Comput. Appl. **42**(1), 108–117 (2020). https://doi.org/10.1080/1206212X.2017.1404823
5. Banerjee, S., Adhikari, M., Kar, S., Biswas, U.: Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud. Arabian J. Sci. Eng. **40**(5), 1409–1425 (2015). https://doi.org/10.1007/s13369-015-1626-9
6. Alworafi, M.A., Dhari, A., Al-Hashmi, A.A., Darem, A.B.: An improved SJF scheduling algorithm in cloud computing environment. In: IEEE ICEECCOT, pp. 208–212 (2016). https://doi.org/10.1109/ICEECCOT.2016.7955216
7. Thomas, A., Krishnalal, G., Raj, V.P.J.: Credit based scheduling algorithm in cloud computing environment. Procedia Comput. Sci. **46**, 913–920 (2015). https://doi.org/10.1016/j.procs.2015.02.162
8. Jindal, A.: Optimization of task scheduling algorithm through QoS parameters for cloud computing. In: ICAET, vol. 57, pp. 1–4 (2016). https://doi.org/10.1051/matecconf/20165702009
9. Semmoud, A., Hakem, M., Benmammar, B., Charr, J.: Load balancing in cloud computing environments based on adaptive starvation threshold. Concurrency Comput. Pract. Exp. **32**(11), 1–14 (2020). https://doi.org/10.1002/cpe.5652
10. Polepally, V., Chatrapati, K.S.: Dragonfly optimization and constraint measure-based load balancing in cloud computing. Cluster Comput. **22**(S1), 1099–1111 (2019). https://doi.org/10.1007/s10586-017-1056-4
11. Lakra, A.V., Yadav, D.K.: Multi-objective tasks scheduling algorithm for cloud computing throughput optimization. Procedia Comput. Sci. **48**, 107–113 (2015). https://doi.org/10.1016/j.procs.2015.04.158
12. Xu, X., Fu, S., Cai, Q., Tian, W., Liu, W., Dou, W., Liu, A.X.: Dynamic resource allocation for load balancing in Fog environment. Wireless Commun. Mobile Comput. **2018**, 1–15 (2018)
13. Yi, S., Li, C., Li, Q.: A survey of Fog computing: concepts, applications and issues. In: Workshop on Mobile Big Data, pp. 37–42 (2015)
14. Yi, S., Hao, Z., Qin, Z., Li, Q.: Fog computing: platform and applications. In: 3rd IEEE Workshop on HotWeb, pp. 73–78 (2015)
15. Stojmenovic, I., Wen, S., Huang, X., Luan, H.: An overview of Fog computing and its security issues. Concurrency Comput. Pract. Exp. **28**(10), 2991–3005 (2016)
16. Iorga, M., Feldman, L., Barton, R., Martin, M.J., Goren, N.S., Mahmoudi, C.: Fog Computing Conceptual Model. NIST Special Publication (2018)
17. Atlam, H.F., Walters, R.J., Wills, G.B.: Fog computing and the Internet of Things: a review. Big Data Cogn. Comput. **2**(2), 10 (2018)
18. Chen, S., Zhang, T., Shi, W.: Fog computing. IEEE Int. Comput. **21**(2), 4–6 (2017)
19. Bellendorf, J., Mann, Z.A.: Classification of optimization problems in Fog computing. Future Gener. Comput. Syst. **107**, 158–176 (2020)
20. Zhang, G., Shen, F., Yang, Y., Qian, H., Yao, W.: Fair task offloading among Fog nodes in Fog computing networks. In: IEEE ICC, pp. 1–6 (2018)
21. Mahmud, R., Kotagiri, R., Buyya, R.: Fog computing: a taxonomy, survey and future directions. In: Internet of Everything, pp. 103–130. Springer, Singapore (2018)
22. Alworafi, M.A., Dhari, A., Al-Hashmi, A.A., Darem, A.B.: An improved SJF scheduling algorithm in cloud computing environment. In: IEEE ICEECCOT, pp. 208–212 (2016). https://doi.org/10.1109/ICEECCOT.2016.7955216
23. Al Nuaimi, K., Mohamed, N., Al Nuaimi, M., Al-Jaroodi, J.: A survey of load balancing in cloud computing: Challenges and algorithms. In: 2nd Symposium on Network Cloud Computing and Applications, pp. 137–142 (2012)
24. Mondal, B., Dasgupta, K., Dutta, P.: Load balancing in cloud computing using stochastic hill climbing. Procedia Technol. **4**, 783–789 (2012)
25. Handles, M., Lamb, D., Taleb-Bendiab, A.: A comparative study into distributed load balancing algorithms for cloud computing. In: IEEE AINA Workshops, pp. 551–556 (2010)
26. Shahrad, M., Fonseca, R., Goiri, Í., Chaudhry, G., Batum, P., Cooke, J., Bianchini, R.: Serverless in the wild: characterizing and optimizing the serverless workload at a large cloud provider. In: USENIX ATC, pp. 205–218 (2020)

27. Amelina, N., Fradkov, A., Jiang, Y., Vergados, D.J.: Approximate consensus in stochastic networks with application to load balancing. IEEE Trans. Inf. Theory **61**(4), 1739–1752 (2015)
28. Xiao, L., Boyd, S., Kim, S.J.: Distributed average consensus with least-mean-square deviation. J. Parallel Distrib. Comput. **67**(1), 33–46 (2007)
29. Yu, D., Ma, Z., Wang, R.: Efficient smart grid load balancing via fog and cloud computing. Math. Probl. Eng. **2022**(1), 3151249 (2022)
30. Vijarania, M., Gupta, S., Agrawal, A., Adigun, M.O., Ajagbe, S.A., Awotunde, J.B.: Energy efficient load-balancing mechanism in integrated IoT–fog–cloud environment. Electronics **12**(11), 2543 (2023)

Chapter 14
Adaptive Workload Management in Cloud and Edge Environments



D. Nandhini and M. Brindha

Abstract Cloud and edge computing together allow for scalable, low-latency, and resource-efficient execution of modern applications. As data-heavy workloads continue to increase, especially in areas like computer vision, adaptive workload management has become essential. Unlike traditional strategies that allocate resources statically, adaptive methods allocate, scale, and rebalance resources based on application needs, system variety, and real-time performance data. This flexibility is crucial for workloads like image classification and recognition, which involve high data volumes and intensive computation. They fit well with federated learning frameworks in distributed settings. This chapter looks at the principles, methods, and tools for adaptive workload management in cloud and edge ecosystems. It focuses on lightweight orchestration strategies, containerized deployment, and autoscaling tools like Kubernetes Vertical Pod Autoscaler (VPA). Special attention is given to minimal-cluster setups, which let distributed computer vision workloads run efficiently without needing large-scale infrastructure. By combining adaptive orchestration techniques with monitoring systems, the chapter shows how to balance workload demands in vision-based federated learning tasks across cloud and edge resources. The discussion also covers practical aspects, such as managing workload variability, improving energy use, and dealing with system diversity. Through this exploration, the chapter offers insights into how adaptive workload management can lead to performance-driven and cost-effective solutions for next-generation cloud and edge systems. It ensures that even compute-heavy applications like distributed image classification can be delivered reliably at scale.

D. Nandhini · M. Brindha (✉)
National Institute of Technology, Trichy, India
e-mail: brindham@nitt.edu

D. Nandhini
e-mail: mail2nandinid@gmail.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_14

14.1 Introduction

Cloud and edge computing have emerged as two complementary concepts in modern distributed systems [9, 16]. Cloud platforms provide almost unlimited computing power, while edge environments place computation closer to data sources. This setup reduces latency and allows for real-time [9, 18]. The combination of these two concepts is essential for meeting the growing demands of data-heavy applications. As Internet of Things (IoT) devices and artificial intelligence (AI) services rapidly expand, the volume of data generated at the edge has increased significantly [2, 17]. Applications in computer vision, such as image classification, object detection, surveillance, and healthcare imaging, impose significant computational and storage pressure on infrastructure [2, 7]. These workloads require not only powerful resources but also efficient scheduling mechanisms to maintain performance without incurring unnecessary overhead [9, 10]. Traditional workload management methods often rely on fixed resource allocation that rarely adapts to workload variations, leading to inefficiencies [5, 8]. While these methods are easy to implement, they usually lead to inefficiencies. Over-provisioning wastes resources and raises costs, while under-provisioning hampers application performance [3]. In contrast, adaptive workload management reallocates resources based on real-time performance metrics, system differences, and application needs. This flexibility is vital for balancing latency, cost efficiency, and reliability in distributed systems. Additionally, there is increasing interest in minimal-cluster strategies for cloudedge orchestration to reduce system complexity and overhead [9, 11]. Instead of using large-scale clusters, which can lead to high orchestration overhead and energy use, minimal clusters aim to utilize a smaller group of nodes with smart orchestration policies. These strategies are especially appealing in environments where controlling costs and conserving energy matter as much as scalability and accuracy. By integrating adaptive resource management with lightweight orchestration, cloud-edge systems can offer sustainable, scalable, and performance-driven services for future data-focused applications.

14.2 Principles of Adaptive Workload Management

Adaptive workload management involves adjusting computing resources and task allocation in real-time within distributed systems, especially in cloud and edge environments [10]. This method is different from static allocation strategies that set resource assignments ahead of time. Instead, adaptive approaches move workloads based on changing application needs, system conditions, and user requests.

The main goals of adaptive workload management include reducing latency to ensure that time-sensitive applications, such as healthcare monitoring or autonomous vehicles, respond without delay. It also aims to optimize resource use by preventing wasted resources and distributing workloads effectively to avoid underused or overloaded nodes [3, 6]. Additionally, it ensures fairness and quality of service so that no

single application or client runs out of resources [10, 11]. Finally, it works to minimize energy consumption and cost by lowering idle resource usage and supporting sustainable, cost-effective operations [6].

Through these objectives, adaptive workload management keeps cloud and edge systems scalable and responsive even in rapidly changing environments.

14.3 Static Versus Adaptive Resource Allocation

In traditional static allocation, resources are assigned based on predicted peak workloads rather than real-time demand [5, 8]. For example, a video analytics service might get a set number of CPUs, even if its workload varies significantly from day to night. This inflexible model often leads to wasted resources during low workloads and performance issues when demand exceeds the fixed amount. In contrast, adaptive resource allocation continuously adjusts resources by using real-time monitoring and prediction models. Instead of overprovisioning, Adaptive strategies monitor system metrics such as CPU utilization, memory consumption, network traffic, and latency to guide workload adjustment [12, 15]. For instance, in a smart surveillance system, adaptive allocation ensures that edge nodes use more GPU cycles during busy hours (e.g., traffic monitoring at rush hour) and reduce usage during quieter times, saving energy and cost. This dynamic approach enables adaptive systems to meet service-level objectives (SLOs) more effectively than static systems [10]. In our minimal-cluster adaptive workload management framework, we implement this concept using Kubernetes Vertical Pod Autoscaler (VPA) along with Prometheus-based real-time monitoring [15]. Static setups fix CPU and memory values for pods, no matter the workload changes. In contrast, our adaptive setup measures actual usage during federated training and adjusts resources on the go. For instance, during CIFAR-10 image classification, when training pods face spikes in computation from local epochs, VPA automatically adds more CPU or memory [7]. Once training stabilizes or aggregation happens, it reduces the resources to avoid waste.

14.4 Key Enablers

Adaptive workload management in cloud and edge ecosystems depends on several key technologies that work together to ensure efficient, responsive, and scalable operation [16]. These tools make up the foundation of any modern distributed system that seeks to balance performance with flexibility. This is especially important in federated or collaborative machine learning setups where workloads are dynamic and data-heavy.

14.4.1 Containerization

Containerization technologies, like Docker and Linux Containers (LXC), package applications with their dependencies into lightweight, portable units [18]. Containers ensure consistent performance across cloud servers and low-resource edge devices. They make it easier to deploy and scale workloads such as CIFAR-10 image classification. By isolating workloads, containers prevent conflicts and let multiple tasks run efficiently on the same hardware.

In our adaptive workload management framework, containerization was key to maintaining reproducibility and portability. Each training environment, whether on the cloud or an edge node, was wrapped in a Docker container that included Python libraries, the dataset loader, and the training scripts. This setup ensured identical runtime environments across different nodes. For instance, when edge nodes trained with subsets of CIFAR-10 data, they could operate in parallel without dependency conflicts or version issues. This lightweight packaging also cut deployment time, allowing containers to start in seconds rather than minutes, unlike traditional virtual machines.

Moreover, this approach allowed for smooth workload migration. If one edge node became overloaded or failed, its container could be quickly redeployed on another available node. This quick redeployment helped improve system resilience and reduced downtime [11] during federated learning rounds.

14.4.2 Orchestration

Managing containers at scale requires orchestration frameworks. Kubernetes offers strong scheduling, networking, and service discovery in cloud environments [16]. KubeEdge builds on these features for resource-limited edge nodes. Together, they allow containerized workloads to be deployed, moved, or scaled across different infrastructures. This reduces manual work and improves system efficiency.

In our case, Kubernetes handled the main orchestration logic. It made sure that every containerized training task was distributed effectively between the cloud and edge layers. The cloud node served as the control center, scheduling tasks, managing pod lifecycles, and tracking performance metrics. KubeEdge helped extend these orchestration features to remote devices, allowing containerized tasks to run close to the data source. This setup reduced latency and bandwidth use.

For instance, image classification tasks running on edge nodes (trained with CIFAR-10 subsets) were organized so that no two nodes processed the same data partition. Kubernetes took care of node health, job scheduling, and pod restarts automatically. This automation significantly reduced orchestration complexity, which is crucial in minimal-cluster environments.

Moreover, Kubernetes' declarative configuration model supported system stability. YAML manifests defined each pod's resource limits and scaling behavior. These

manifests worked alongside autoscalers and monitoring systems, creating a closed feedback loop crucial for managing workloads effectively.

14.4.3 Autoscaling

Autoscaling is crucial for managing workloads, as it adjusts system resources to match real-time changes. Kubernetes offers tools like Cluster Autoscaler (CA) and Horizontal Pod Autoscaler (HPA), but these have limitations in federated learning and edge scenarios [13]. The CA may react slowly because of delays in node provisioning, which affects how quickly the system responds during frequent training updates. Similarly, the HPA can create extra pod replicas that process the same data, wasting CPU and GPU resources and causing synchronization delays among distributed clients. These challenges are more apparent in minimal-cluster environments, where efficiency with limited nodes is key. To tackle these issues, we mainly used the Vertical Pod Autoscaler (VPA) to adjust CPU and memory allocations for each training pod [1]. This ensured we used resources optimally across the cluster.

In our federated learning setup with computer vision tasks, like CIFAR-10 image classification, VPA was essential for balancing computing demands. During intense training phases, the autoscaler automatically increased CPU and memory allocations to handle higher processing needs. During quiet periods, it reduced resources to cut down on waste. This improvement led to better overall training performance while keeping stable results across varied edge nodes. By combining VPA with Prometheus-based monitoring, our system set up a continuous feedback loop that informed real-time scaling decisions. This flexible approach worked especially well in our minimal-cluster design. It reduced management overhead, avoided resource shortages, and ensured that small federated systems could perform on par with larger setups while keeping costs and energy use low.

14.4.4 Monitoring

Real-time monitoring is essential for effective autoscaling. Tools like Prometheus and Grafana gather metrics such as CPU usage, memory consumption, network throughput, and training duration. This information creates a continuous feedback loop that helps adjust resources in real time. These monitoring systems serve as the eyes of the orchestration layer, constantly observing how workloads behave in both cloud and edge environments [12].

In our system, Prometheus tracked resource usage across all pods and edge nodes, collecting performance data at short intervals. The metrics provided system-level information and included workload-level insights like model training progress, batch processing time, and I/O performance. We visualized these metrics using Grafana

dashboards, offering a clear view of how resources were used during each training phase.

This setup allowed the Vertical Pod Autoscaler (VPA) to allocate resources effectively during peak training phases of the CIFAR-10 workload and to reduce resources during idle times. For example, when model training at the edge saw a sudden increase in CPU or GPU load, the Prometheus metrics showed the change right away. The autoscaler then adjusted CPU and memory limits dynamically to maintain steady performance without needing human input.

The close integration between Prometheus and Kubernetes VPA ensured that every scaling decision was based on data and responsive. By combining real-time metric analysis with automatic scaling, the system could quickly react to changes in workload behavior, minimizing downtime and preventing waste of resources.

This blend of monitoring and vertical scaling enabled smooth and uninterrupted federated learning in a distributed environment. It avoided overprovisioning, which allocates too many resources that sit idle, and underprovisioning, which gives too few resources and leads to slowdowns or failures.

This intelligent balance showed that small cloud-edge clusters can perform like larger, more resource-heavy deployments. Even with fewer nodes, accurate monitoring and adaptive autoscaling led to stability, performance, and efficiency. In short, real-time monitoring not only kept operations running smoothly but also allowed the system to adapt based on current performance patterns, making it a vital part of modern adaptive workload management frameworks.

14.5 Autoscaling Techniques Used for Our Workload Management Framework

We deployed workloads in a small Kubernetes cluster designed for cloud-edge environments. In this setup, the cloud served as the orchestration layer that managed scheduling, monitoring, and resource allocation. The edge nodes handled localized training operations using partitioned CIFAR-10 datasets. This deployment aimed to explore how adaptive workload management can balance performance and efficiency while using fewer computing nodes, which is crucial for research and resource-limited environments.

Each training task was containerized with all necessary dependencies. These included machine learning frameworks like TensorFlow and PyTorch, as well as dataset preprocessing scripts and federated learning coordination libraries. Containerization allowed each training instance to run independently, preventing version conflicts and ensuring a consistent runtime environment across both the cloud and edge nodes.

This setup provided consistent execution across cloud servers and various edge devices with differing computing power. By standardizing the software environment with Docker containers, even diverse nodes, from GPU-enabled cloud machines

to CPU-only edge devices, could effectively train and contribute model updates. This compatibility was vital for achieving fair workload distribution and balanced performance during federated training rounds.

Kubernetes and KubeEdge managed these containers, overseeing deployment, scheduling, and dynamic scaling while minimizing manual tasks. The Kubernetes master node, hosted in the cloud, coordinated the deployment of pods, while KubeEdge extended control to distributed edge devices. This hybrid orchestration eased the burden of manual scaling and scheduling, as workloads were automatically spread based on node availability and real-time performance metrics.

During CIFAR-10 training, the Vertical Pod Autoscaler (VPA) adjusted CPU and memory allocations for each pod based on real-time metrics collected by Prometheus. Prometheus constantly monitored the hardware resource utilization across all active pods. If a pod faced a spike in computational demand, like during intense model convergence or batch normalization, VPA increased CPU and memory limits. When activity slowed, VPA automatically reduced allocations to prevent wasting resources.

This approach ensured pods received sufficient resources during peak training times and scaled down during idle periods, leading to efficient use and smooth execution across different nodes. This dynamic adjustment was crucial in federated learning settings, where workloads shift between active computation and synchronization phases. By reallocating resources smartly, the system maintained stable training progress without overloading any single node.

While Kubernetes offers tools like Cluster Autoscaler (CA) and Horizontal Pod Autoscaler (HPA), adjusting to different edge workloads can be challenging due to differences in node capacities and synchronization needs during training rounds [19]. For instance, while CA scales entire nodes and HPA replicates pods, both are less effective in minimal clusters where hardware diversity and communication delays present challenges. In these situations, pod-level resource tuning, enabled by VPA, provides more precise control and avoids unnecessary training tasks.

In our setup, VPA facilitated more detailed, pod-level adjustments that matched resources to actual usage. Instead of allocating fixed resources in advance, each pod's needs were assessed dynamically. If a training job required more memory for data augmentation or model checkpointing, the system responded automatically without human input.

Prometheus continued to track CPU use, memory consumption, and training duration, creating a feedback loop that influenced VPA's decisions. This feedback loop formed the foundation of the adaptive workload management process. Data collected by Prometheus was analyzed to spot trends, like which nodes often reached peak utilization or which pods consistently ran below capacity. These insights helped VPA refine its future scaling actions.

This system maintained continuous federated learning, improved training efficiency, reduced delays, and optimized resource usage. Because autoscaling and monitoring worked together effectively, there were no training interruptions caused by pod crashes or resource shortages. Synchronization between cloud and edge nodes became smoother, and overall convergence time improved significantly—by about 4050

Table 14.1 Comparison of autoscaling techniques in cloud-edge workloads

Technique	Scales	Strengths	Limitations (FL/CV)	Best use case
CA	Nodes	Ensures capacity	Slow reaction	Large cloud jobs
HPA	Pod replicas	Good for stateless	Redundant training	Web/microservices
VPA	CPU/memory	Optimizes resources	Pod restarts; tuning needed	FL & CV

This showed that even small cloud-edge clusters, when managed with containerized workloads, orchestration, monitoring, and vertical scaling, can perform similarly to larger setups. It demonstrates that performance and scalability depend not only on cluster size but also on smart orchestration and adaptive scaling.

This approach is well-suited for research, prototyping, and environments with limited resources. By enabling high computational efficiency within constrained infrastructures, this framework lays the groundwork for real-world applications, such as smart healthcare systems, distributed surveillance, and IoT-based vision analytics, where adaptive workload management ensures responsiveness, reliability, and cost efficiency.

14.5.1 Comparison of Autoscaling Techniques

A comparative summary of CA, HPA, and VPA is presented in Table 14.1.

14.6 Lightweight Orchestration Strategies

Managing workloads effectively in cloud-edge ecosystems doesn't always need large clusters with many nodes [16]. In several real-world cases, like research settings, small industry deployments, and resource-limited edge environments, lightweight orchestration strategies provide a practical and cost-effective solution. The aim is to keep the main features of orchestration platforms while reducing overhead, cost, and complexity.

Lightweight orchestration targets optimizing available resources instead of blindly scaling infrastructure. This method is especially useful when dealing with limited physical nodes or when energy and cost efficiency matter most. In cloud-edge scenarios, even small clusters can perform well when orchestration smartly matches workload demand with available computing power. For example, in academic or prototype settings, operating 10–20 nodes can offer near real-time analytics if the system dynamically adjusts CPU and memory resources through adaptive policies.

This method differs from traditional cloud setups that depend on excessive redundancy to ensure performance. Instead, lightweight orchestration strategies encourage sustainable scalability by achieving performance and reliability without wasting resources.

Lightweight orchestration also helps maintain fault tolerance and system stability in limited deployments. Using tools like Kubernetes and KubeEdge, administrators can automate task scheduling, fault recovery, and scaling, even across different edge devices. This guarantees that workloads, whether for computer vision inference, IoT sensor aggregation, or federated model updates, run smoothly without manual reconfiguration.

14.6.1 Minimal-Cluster Concept

The minimal-cluster strategy aims to use fewer nodes for deploying and managing workloads while ensuring efficient resource use [9]. Instead of randomly scaling clusters, minimal clusters employ smart scheduling and dynamic scaling. By managing fewer nodes, orchestration overhead decreases, simplifying scheduling and reducing resource use in control operations.

This approach also lowers operational costs, which is crucial for academic research setups or startups with tight budgets. It shortens communication paths, benefiting synchronous workloads like federated learning that rely on frequent parameter aggregation. Plus, it consumes less power, which supports green computing goals and IoT edge applications.

Minimal-cluster systems are especially useful in federated learning environments, where multiple clients (edge nodes) take part in decentralized training. In these scenarios, orchestration focuses on balancing computation among devices instead of adding new hardware. For instance, a cluster with 15–20 nodes can work effectively, while a larger cluster with 30 or more nodes is unnecessary. Smart orchestration minimizes idle cycles, reduces inter-node communication, and keeps synchronization between the cloud and edge consistent.

In a federated CIFAR-10 classification task, for example, a cluster of 15–20 nodes is sufficient, rather than deploying 30 or more nodes as seen in large-scale experiments. By carefully balancing workloads and using autoscaling, like the Vertical Pod Autoscaler (VPA), similar accuracy can be reached with less infrastructure.

The strength of this method is in optimizing current computational power rather than adding more hardware. VPA adjusts CPU and memory resources based on workload intensity. It increases capacity during training peaks and reclaims resources during idle times to improve efficiency. This cuts down on unnecessary synchronization delays between cloud and edge nodes, reduces communication bandwidth use, and enables edge devices to train longer locally before syncing with the global model. As a result, model accuracy remains intact while communication costs drop significantly.

In practical terms, in a setup with 15–20 nodes, instead of exchanging updates every single round like in traditional federated learning, updates can be shared every 3–5 local epochs. This significantly cuts down orchestration messages and enhances throughput. Combined with VPA-based resource adjustments, this method allows minimal clusters to perform nearly as well as large clusters while saving 40–50% on communication costs.

Thus, minimal-cluster strategies demonstrate that efficiency in performance comes not from having more nodes but from smartly orchestrating existing nodes through dynamic workload management.

14.6.2 Containerized Deployment

Containers play a crucial role in lightweight orchestration. Unlike virtual machines that require full guest operating systems, containers share the host kernel while packaging applications and their dependencies [18]. This design makes them lightweight, portable, and quick to deploy.

In federated learning, each client or edge device can run its local training process inside a Docker container. This setup ensures that one client does not interfere with another, allows for portability across cloud servers, GPUs, or edge devices like Raspberry Pi, and guarantees consistent execution in various environments.

Containerized deployments remove version mismatches and dependency conflicts. They provide reproducible experiments even when the hardware architectures are different. They also allow for precise control of workloads; containers can be paused, moved, or restarted without impacting the rest of the system. This capacity makes them ideal for managing workloads, where tasks often shift between edge and cloud depending on resource availability.

They work well with orchestration systems like Kubernetes and KubeEdge. Kubernetes ensures that containers are automatically scheduled, monitored, and restarted if they fail, while KubeEdge extends these functions to remote edge nodes with limited connectivity.

Additionally, container-based deployment enables parallel processing across edge nodes. For instance, in federated learning for medical image analysis, each hospital (edge node) can independently process its local dataset while remaining synchronized with the cloud aggregator. This containerized approach not only improves data privacy but also speeds up deployment and maintenance across systems distributed over large areas.

14.7 Adaptive Workload Management Framework

Adaptive workload management in cloud and edge federated environments needs a clear framework that separates global orchestration tasks in the cloud from local

execution tasks at the edge [16]. This framework makes sure that distributed training workloads are handled effectively while balancing cost, latency, and energy use.

The proposed framework outlines two main layers: a cloud orchestration layer and an edge execution layer. The cloud layer manages high-level coordination, such as distributing workloads, adjusting computational resources dynamically through autoscaling (VPA), and performing model aggregation. At the same time, the edge layer carries out local training using partitioned datasets.

This separation helps the system maintain scalability and responsiveness. Cloud nodes are responsible for global decision-making, while edge nodes perform tasks sensitive to latency near data sources. The outcome is a strong and efficient system that can support distributed machine learning workloads, like computer vision or IoT analytics, with minimal communication overhead.

Additionally, integrating components like Prometheus for monitoring, Nginx for load balancing, and Kubernetes for orchestration creates a feedback-driven automation loop. Resource usage metrics collected by Prometheus influence the VPA to adjust pod configurations. Meanwhile, Nginx distributes data and training requests evenly across active nodes. This dynamic coordination ensures fault tolerance, high utilization, and stable system performance even when workloads fluctuate.

This framework ultimately shows how adaptive workload management, combined with minimal-cluster orchestration and containerized deployment, can provide scalable, energy-efficient, and cost-effective distributed computing for future federated and computer vision applications.

14.7.1 Components of the Framework

The proposed framework starts with **Data Ingestion**. The CIFAR-10 dataset is divided into non-IID subsets and distributed across several virtual machines that simulate edge environments. These VMs act like different edge devices, reflecting the data diversity found in real-world settings involving IoT devices, mobile clients, or vision sensors. Rather than sending raw data to the cloud, the system exchanges only pre-processed features or model updates. This reduces communication costs and protects privacy.

Next comes **Model Training**, which is performed independently on each simulated edge node. Each VM trains a local model on its assigned dataset subset, with the workloads containerized using Kubernetes to ensure isolation, portability, and reproducibility. This setup allows the system to replicate realistic federated learning conditions under non-IID distributions while avoiding the complexity of deploying physical devices.

Finally, **Aggregation** takes place at the cloud node, which functions as the global coordinator. The cloud collects model updates (weights and gradients) from all edge VMs and integrates them into a unified global model. By aggregating updates instead of raw datasets, the framework enhances data privacy, reduces bandwidth requirements, and supports efficient global model convergence.

14.7.2 Role of Nginx in Load Balancing

In the orchestration layer, Nginx ensures smooth workload distribution across edge nodes. It evenly balances incoming training requests and data streams among all available resources, preventing any single node from becoming overloaded [12]. This load distribution is crucial for maintaining consistent performance, especially during federated learning sessions where multiple edge clients communicate with the central server at the same time. It avoids bottlenecks and allows for seamless scaling when new pods or nodes are added through Kubernetes autoscaling, like the Vertical Pod Autoscaler (VPA) or Horizontal Pod Autoscaler (HPA). When the system encounters high traffic, Nginx adjusts routing policies dynamically to keep stability and throughput. If a node fails or becomes unreachable, Nginx reroutes requests to active nodes immediately, ensuring that federated training continues without interruption or data loss. This ability improves fault tolerance and ensures continuous system availability. Acting as a gateway between clients and federated learning pods, Nginx offers reliability, fairness, and scalable workload management. It also helps use network bandwidth efficiently by directing requests based on latency and node capacity. With smart request forwarding and session persistence, Nginx enhances overall system responsiveness and reduces synchronization delays in minimal-cluster setups.

14.7.3 Cloud Versus Edge Separation

The cloud node mainly manages orchestration, scheduling, and aggregation. It keeps an overall view of how workloads are distributed. The cloud hosts the Vertical Pod Autoscaler (VPA) controller, which adjusts pod resources like CPU and memory based on tracked metrics. It also handles centralized monitoring with Prometheus and Grafana, and distributes requests using Nginx. In contrast, edge nodes store local data and preprocess it. They train machine learning models on divided datasets and send performance metrics and model updates back to the cloud.

This clear separation ensures the cloud focuses on improving resource management and global coordination, while edge nodes manage computation-heavy and latency-sensitive tasks close to the data source. This setup achieves efficiency and responsiveness in federated learning. The cloud node effectively cuts down orchestration overhead by making global decisions, while edge devices work independently to process data locally. Together, they build a smooth communication loop, where the cloud provides scalability and reliability, and the edge offers speed and context. This hybrid approach not only reduces network congestion and energy consumption but also improves the adaptability and fault tolerance of the overall federated learning framework. Figure 14.1 shows the separation between cloud orchestration and edge-level execution in the proposed adaptive workload management framework.

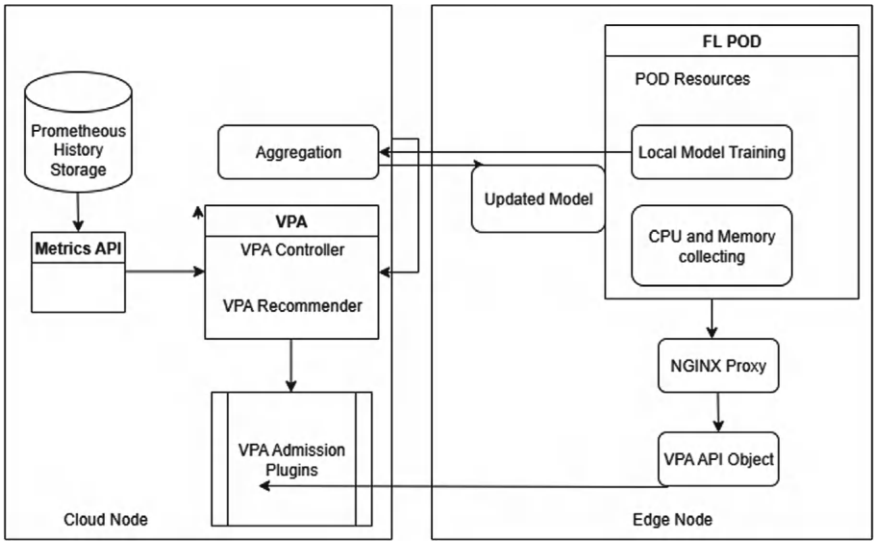


Fig. 14.1 Architecture of the adaptive workload management framework with separate layers for cloud orchestration and edge execution

14.8 Experimental Setup

The experimental setup involved applying the proposed framework in a small Kubernetes cluster. One cloud node acted as the global coordinator, while multiple edge nodes represented different devices responsible for local training. The CIFAR-10 dataset served as the testing workload, divided into non-IID subsets to mimic realistic edge scenarios where each node trained on uneven class distributions. Training workloads were organized in pods to ensure portability, isolation, and reproducibility. Autoscaling components adjusted computational resources during training.

In this setup, the cloud node functioned as the orchestration layer, managing workload scheduling, autoscaling, and aggregation. It took care of global model synchronization and vertical scaling actions using the Vertical Pod Autoscaler (VPA). Meanwhile, the edge nodes operated as execution units, conducting local computation and training with their respective data partitions. Each edge device ran independently within Docker containers managed by Kubernetes, ensuring lightweight orchestration and reliable task isolation.

The workflow followed a federated learning approach. The cloud node initialized the global model and distributed it to the edge nodes. Each node carried out local training on its assigned data. Updates were then sent back to the cloud to enhance the central model. This iterative process continued until the global model converged. By using Kubernetes orchestration and Prometheus-based monitoring, the system dynamically responded to workload changes and ensured smooth synchronization between training and aggregation rounds.

To maintain efficient performance in a small cluster configuration of 19 nodes, the system used adaptive resource allocation. During high-load phases, when multiple edge nodes trained on large batches of CIFAR-10 images, the VPA scaled up CPU and memory allocations. When the workload decreased, it automatically scaled down the unused resources. This real-time adaptability prevented resource underutilization, reduced orchestration overhead, and optimized both latency and throughput.

Additionally, Prometheus continuously monitored system metrics, such as CPU utilization, memory consumption, and training duration. These metrics informed the autoscaling controller for resource reallocation. For example, when CPU usage exceeded 70%, the VPA triggered a scaling event, providing more compute resources to the affected pods. Conversely, during idle training intervals or synchronization waits, resources scaled down automatically, saving energy and operational costs.

Two scenarios were tested: one without autoscaling, where fixed resource allocation caused inefficiencies and slowed down convergence during workload spikes, and one with the Vertical Pod Autoscaler (VPA), where CPU and memory adjusted based on usage. The latter reduced training delays during busy times and minimized resource waste during idle periods, showcasing effective management of image classification workloads in cloud-edge environments.

In the scenario without autoscaling, CPU utilization averaged around 25–30%, with frequent idle periods and longer synchronization times between nodes. Memory utilization was also uneven; fixed allocations left some pods overprovisioned while others starved during peak load. Training convergence was slower, taking nearly five hours to reach the target accuracy due to static provisioning.

In contrast, the scenario with VPA showed significant efficiency gains. CPU usage dynamically stabilized around 65–75%, indicating balanced load distribution. Memory usage improved to nearly 80%, reflecting optimal use of available resources. The adaptive scaling reduced the total convergence time to about three hours–almost 40% faster than the static case.

Prometheus logs confirmed smoother CPU and memory usage curves with fewer spikes, validating the advantages of continuous monitoring and feedback-based orchestration. Furthermore, the adaptive scaling mechanism maintained model accuracy comparable to the baseline while using fewer physical resources.

This experiment demonstrated that lightweight orchestration using VPA can effectively support federated learning in computer vision workloads, especially in minimal-cluster environments. The combination of containerized deployment, vertical scaling, and monitoring reduced both communication overhead and energy consumption, confirming the framework's capability to sustain real-world, large-scale federated image classification with limited infrastructure.

14.9 Results and Discussion

The results highlight the performance improvements achieved through adaptive workload management:

Table 14.2 System performance comparison with and without VPA autoscaling

Metric	Without autoscaling	With VPA autoscaling	Improvement
CPU utilization (%)	~ 25–30	~ 65–75	+ 40
Memory utilization (%)	~ 45	~ 80	+ 35
Convergence time (h)	5.0	3.0	~ 40% faster

CPU Usage: With static allocation, CPUs often remained underutilized or were unevenly saturated. With VPA, CPU resources were balanced dynamically, improving efficiency across the cluster.

Memory Usage: Static allocation caused memory waste due to over-provisioning. With VPA, memory requests and limits were continually adjusted, ensuring optimal usage without exceeding pod limits.

Training Time Reduction: Adaptive scaling reduced bottlenecks during heavy training phases, leading to 40–50% faster convergence compared to the static allocation baseline (Table 14.2).

Key Finding: Adaptive workload management speeds up training and cuts idle resource costs. This makes it a practical solution for deployments of federated learning in computer vision applications that have limited resources. Figure 14.2 and 14.3 compare system behavior without and with VPA autoscaling, showing inefficient resource usage under static allocation and improved stability with adaptive scaling.

14.10 Challenges and Open Issues

Despite the clear benefits of adaptive workload management in cloud and edge environments, several challenges remain unsolved [2].

Heterogeneity of Devices: Edge environments consist of a mix of devices, including high-performance servers and GPUs, as well as lightweight IoT sensors or embedded devices. Each type has different capabilities regarding compute, memory, and energy. Designing autoscaling strategies that work across such varied systems is complicated. For example, a federated learning job focused on image classification may perform well on GPU-equipped nodes but could stall on resource-limited edge cameras, leading to an imbalance in the system.

Network Constraints in Edge Environments: Network latency, bandwidth fluctuations, and intermittent connectivity greatly impact workload orchestration. Federated learning setups are especially affected because model updates must exchange frequently between the edge and cloud. If bandwidth is low, communication delays can overshadow the advantages of adaptive scaling and slow down convergence.

Security and Privacy under Adaptive Scaling: Adaptive scaling depends on continuous monitoring of system performance metrics. However, this ongoing adjust-

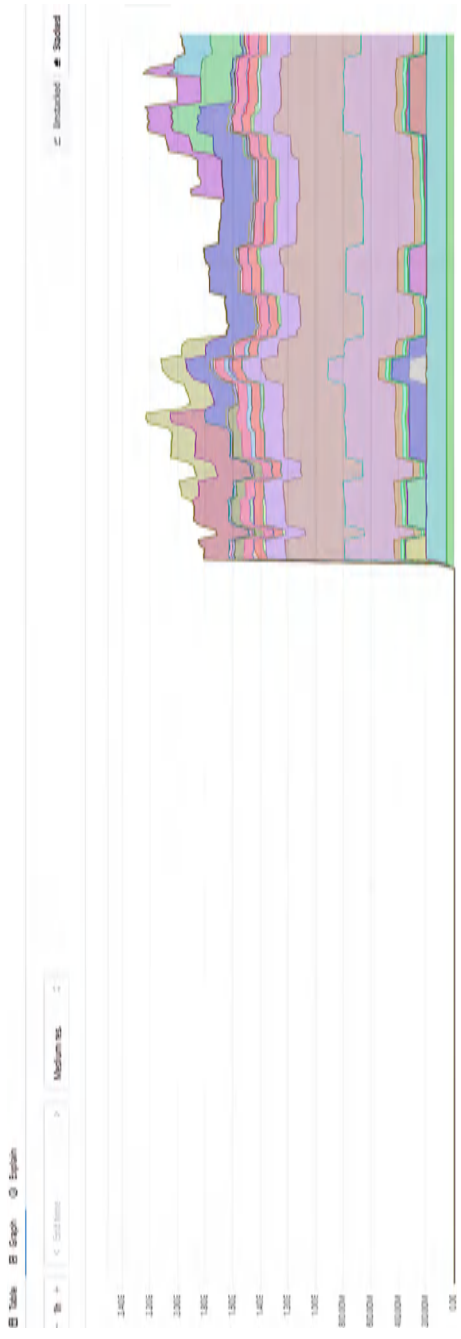


Fig. 14.2 System performance without VPA autoscaling

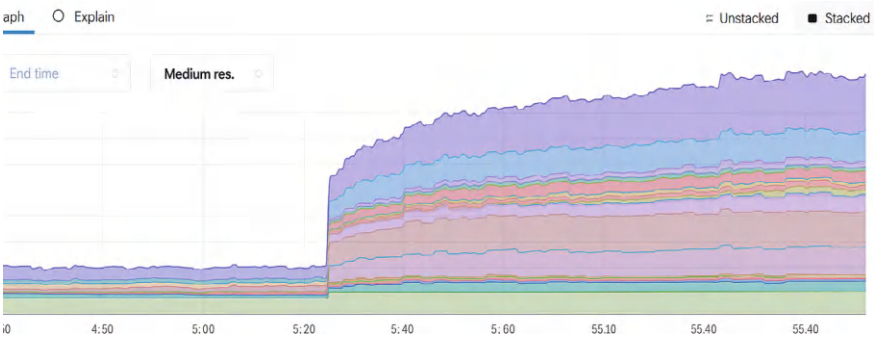


Fig. 14.3 System performance with VPA autoscaling

ment can create security vulnerabilities, like exposing monitoring data that others could exploit. Additionally, workload migration or scaling in federated learning must ensure privacy protections. Without these, sensitive edge data could be unintentionally leaked.

Energy Efficiency: Edge nodes are often battery-powered or energy-constrained, such as mobile devices and IoT nodes. Aggressive scaling policies may boost performance but can drain energy resources. Finding a balance between responsiveness, efficiency, and energy sustainability is a significant ongoing challenge.

Trade-off between Responsiveness versus Overhead in Autoscaling: Autoscaling decisions rely on real-time monitoring and orchestration logic, which adds overhead. Scaling too aggressively can cause frequent resource reconfigurations, while scaling too conservatively may leave workloads short on resources. Finding the right balance is still an active research topic.

14.11 Future Directions

Integration of AI-driven Predictive Autoscaling: Current autoscaling policies mainly react to performance drops after they happen. Future methods will likely use machine learning-based predictive scheduling to anticipate workload surges and adjust resources proactively. For example, an AI-driven autoscaler could foresee an increase in video analytics demand during peak traffic hours and allocate resources in advance.

Expansion to Multi-Modal Workloads: Most studies on adaptive workloads focus on one modality, such as images. However, real-world applications often include video, text, sensor data, and speech at the same time. Extending adaptive management to multi-modal federated learning could enable smarter edge systems for autonomous vehicles, healthcare monitoring, or smart city surveillance.

Sustainability and Green Computing: Energy efficiency will be crucial in future workload management frameworks. Techniques like carbon-aware scheduling, renewable energy integration, and energy-based autoscaling policies will help ensure that cloud-edge systems stay environmentally friendly without sacrificing performance.

Cloud Edge Fog Continuum Orchestration: Beyond traditional cloud and edge architectures, fog computing adds an intermediate layer between edge devices and cloud data centers. Managing workloads across this continuum could improve latency and resource sharing. Future frameworks need to extend their orchestration to this three-layer environment.

Next-Generation Computer Vision Workloads: Future applications like real-time video analytics, AR/VR, and autonomous robotics require extremely low-latency inference. Adaptive workload management must improve to not only train models effectively but also support real-time deployment at the edge. This will need ultra-fast scaling strategies, lightweight inference optimization, and hybrid orchestration methods.

14.12 Conclusion

Adaptive workload management is key for efficient cloud and edge computing. By allocating resources based on current conditions, it avoids the limitations of fixed provisioning and helps distributed systems stay responsive, scalable, and cost-effective.

Minimal-cluster strategies, along with containerized deployment and monitoring systems, provide a practical way to achieve this balance without depending on large infrastructure. They demonstrate how to maintain resource efficiency and training performance in workloads like federated learning for computer vision, where both diversity and high data loads are common.

These methods also apply to healthcare, IoT ecosystems, and intelligent edge applications. By linking adaptive workload management to real-world needs, future systems can provide sustainable, scalable, and smart orchestration.

Additionally, using advanced autoscaling tools such as the Vertical Pod Autoscaler (VPA) ensures ongoing optimization of computing and memory resources across both cloud and edge layers. This coordination allows for smooth interaction between global orchestration and local computation. The blend of lightweight orchestration, dynamic scaling, and real-time monitoring opens the door to more resilient distributed systems. As future workloads become more complex, especially in computer vision, federated learning, and real-time analytics, adaptive workload management will remain crucial for achieving efficiency, sustainability, and performance at scale.

Acknowledgements The authors used ChatGPT (OpenAI) for language refinement and grammatical correction only. AI did not generate content (ideas, analysis or results) and all scientific thinking, structure, and conclusions are the authors' own.

References

1. Pham, K.Q., Kim, T.: Elastic federated learning with Kubernetes vertical pod autoscaler for edge computing. Future Gener. Comput. Syst. **141**, 1–14 (2024)
2. Liberti, F., Berardi, D., Martini, B.: Federated learning in dynamic and heterogeneous environments: advantages, performances, and privacy problems. MDPI Electron. **14**(18), 8490 (2024)
3. Tran, M.N., Nguyen, T.T.: Hybrid resource quota scaling for Kubernetes-based edge computing systems. MDPI Electron. **14**(16), 3308 (2025)
4. Zhou, Z., Zhang, C., Ma, L., Gu, J., Qian, H., Wen, Q., Sun, L., Li, P., Tang, Z.: AHPA: Adaptive horizontal pod autoscaling systems on Alibaba cloud container service for Kubernetes. In: Proceedings of 2023 ACM/IEEE International Conference on Cloud Engineering (IC2E), pp. 1–10 (2023)
5. Ju, L., Singh, P., Toor, S.: Proactive autoscaling for edge computing systems with Kubernetes. In: Proceedings of 2021 IEEE International Conference on Edge Computing (EDGE), pp. 1–8 (2021)
6. Medeiros, D., Williams, J.J., Wahlgren, J., Leite, L.S.M., Peng, I.: ARC-V: Vertical resource adaptivity for HPC workloads in containerized environments. In: Proceedings of 2025 IEEE International Conference on High Performance Computing (HiPC), pp. 1–8 (2025)
7. Chen, D., Hu, J., Tan, V.J., Wei, X., Wu, E.: Elastic aggregation for federated optimization. In: Proceedings of 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–10 (2023)
8. Dogani, J.: Proactive auto-scaling technique for web applications in cloud environments. Comput. Electr. Eng. **102**, 108435 (2024)
9. Wang, B., Irwin, D., Shenoy, P., Towsley, D.: INVAR: Inversion aware resource provisioning and workload scheduling for edge computing. In: Proceedings of IEEE INFOCOM 2024—IEEE Conference on Computer Communications, pp. 1511–1520 (2024)
10. Han, Y., Shen, S., Wang, X., Wang, S., Leung, V.C.M.: Tailored learning-based scheduling for Kubernetes-oriented edge-cloud systems. In: Proceedings of 2021 IEEE International Conference on Communications (ICC), pp. 1–6 (2021)
11. Pashaeehir, A., Shariati, S., Shafaghi, S., Moghimi, M., Momtazpour, M.: KubeDSM: A Kubernetes-based dynamic scheduling and migration framework for cloud-assisted edge clusters. arXiv:2501.07130v2 (2025)
12. Kubernetes Autoscaling Patterns: HPA, VPA, and KEDA. SpectroCloud Blog (2025). Available at: https://www.spectrocloud.com/blog/kubernetes-autoscaling-patterns-hpa-vpa-and-keda
13. Cluster Autoscaler: Best Practices, Limitations, and Alternatives. ScaleOps Blog (2025). Available at: https://scaleops.com/blog/kubernetes-cluster-autoscaler-best-practices-limitations-alternatives/
14. Use the cluster autoscaler in Azure Kubernetes Service (AKS). Microsoft Docs (2025). Available at: https://learn.microsoft.com/en-us/azure/aks/cluster-autoscaler
15. Kubernetes VPA: Pros, Cons, and Step-By-Step Guide. Spot.io Resources (2025). Available at: https://spot.io/resources/kubernetes-architecture/kubernetes-vpa-pros-cons-and-step-by-step-guide/
16. Kubernetes on Edge: Key Capabilities, Distros & Best Practices. Komodor (2025). Available at: https://komodor.com/learn/kubernetes-on-edge-key-capabilities-distros-and-best-practices/
17. Five key capabilities to succeed with your AI/ML workloads. SpectroCloud Blog (2025). Available at: https://www.spectrocloud.com/blog/k8s-ai-five-key-capabilities-for-ai-ml-workloads
18. Bring your Kubernetes workloads to the edge. Red Hat Developers (2021). Available at: https://developers.redhat.com/articles/2021/11/22/bring-your-kubernetes-workloads-edge
19. Scale cluster compute with Karpenter and Cluster Autoscaler. AWS Documentation (2025). Available at: https://docs.aws.amazon.com/eks/latest/userguide/autoscaling.html

Chapter 15
Machine Learning and Edge Computing Synergies for Next-Generation Sustainable Networks



Dinesh Sahu, Nidhi, Shiv Prakash, Amira Fauad, and Bhupendra Kumar

Abstract The rapidly expanding nature of mobile communication networks and the interconnected devices has compounded the need to implement sustainable and energy-efficient networks. With 5G coming of age and 6G on the horizon, the Information and Communication Technology (ICT) industry is under increasing pressure to decrease its carbon footprint whilst also facilitating ultra-reliable low-latency communications, massive Internet of Things connectivity, and holographic digital experiences. To work on these challenges, it is necessary to converge Machine Learning (ML) and edge computing, which are some of the critical enablers of intelligent, adaptive, and environmentally conscious infrastructures. This paper looks into the synergies of ML and edge computing in next-generation sustainable networks. In particular, it discusses the possibility of deploying ML-based strategies like intelligent resource allocation, distributed, and federated learning, and real-time analytics to the edge to reduce energy usage, improve scale, and quality of service. Edge computing enables local processing of data to cut the energy bills of cloud backhaul and ML improves prediction and optimization to prevent over-provisioning. The combination of both makes them a complementary paradigm of low-latency, efficient and green communication systems. Critical contributions of the work are: (i) a holistic

D. Sahu (✉) · Nidhi · B. Kumar
School of Computer Science and Engineering (SCSET), Bennett University, Greater Noida, India
e-mail: sahu.dinesh1230@gmail.com

Nidhi
e-mail: nnidhi.jnu@gmail.com

B. Kumar
e-mail: bhupendera.kumar@bennett.edu.in

S. Prakash
Department of Electronics and Communication, University of Allahabad, Prayagraj, India
e-mail: shivprakash@allduniv.ac.in

A. Fauad
Ministry of Education, Damascus, Syria
e-mail: amirayfahoud@gmail.com

Department of Computer Science and Engineering, Aljazeera Private University, Damascus, Syria

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_15

concept of a sustainable network design based on the use of ML-edge integration, (ii) a discussion of the carbon footprint optimization strategies based on predictive load shifting and demand-aware provisioning, (iii) discussion of application scenarios in smart cities, healthcare, transportation, and Industry 5.0, and (iv) identification of challenges and future research opportunities, such as scalability, heterogeneity of data, and standardization. These insights reflect that ML and edge computing play a crucial role in advancing connectivity of the future, which is intensive, intelligent, and sustainable.

Keywords Machine learning · Edge computing · 5G networks · 6G networks · Sustainable communication systems · Federated learning · Intelligent resource allocation · Green networking

15.1 Introduction

The massive rise in the number of mobile communication systems and digital services has greatly augmented the energy consumption and environmental footprint of the Information and Communication Technology (ICT) industry. With the development of networks to the fifth-generation (5G) and beyond, sustainability has become an even more urgent concern, as ICT is already estimated to contribute as much as 2–3% of global greenhouse gas emissions, and this percentage could increase even more to 14% by 2040 unless controlled [1]. This increasing energy footprint requires immediate research on green and sustainable solutions consistent with the United Nations Sustainable Development Goals (SDGs), especially SDG 7 (Affordable and Clean Energy), SDG 9 (Industry, Innovation, and Infrastructure), and SDG 13 (Climate Action) [2]. Green infrastructure, energy efficiency, and smart use of resources are hence becoming driving forces of next-generation sustainable networks.

15.1.1 Motivation for Sustainable Next-Generation Networks

Telecommunication sustainability is no longer a secondary issue, but a fundamental design consideration. Contemporary communication systems should be able to strike a balance between performance and scalability, and environmental friendliness. Indicatively, in the example of a mobile operator, the base stations are said to constitute almost 60–70% of overall energy costs incurred by the mobile operator [3]. Without energy-sensitive solutions, densification of small cells in ultra-reliable low-latency communications (URLLC) and massive machine-type communications (mMTC) might increase carbon footprints [4]. Studies have therefore turned into combining renewable-powered edge nodes, resource allocation, and AI-based optimization to make its operations carbon-neutral [5]. These sustainable systems are

not only essential to the environment but also economical as they lower operation costs to telecommunication companies.

15.1.2 Role of Machine Learning and Edge Computing

Machine Learning has become an enabling technology of next-generation networks allowing predictive and adaptive strategies at multiple layers. ML methods may offer predicting traffic, dynamic spectrum assignment, anomaly detection, and optimization of QoS [6]. As an example, reinforcement learning algorithms have been used to optimize radio resource management in 5G networks, minimising latency and enhancing user experience [7]. Furthermore, federated and unsupervised learning designs might be able to handle large volumes of network data, which enables intelligent, distributed decision-making that does not involve centralized bottlenecks.

In addition to ML, edge computing brings computation nearer to end-users, eliminating the need to rely on remote cloud servers and therefore decreasing latency, enhancing privacy, and conserving energy in backhauls [8]. In new ultra-dense networks, edge computing is the execution layer for executing ML models to provide real-time analytics and local inference to latency-sensitive applications like autonomous vehicles, AR/VR, and smart medical care. Edge computing eliminates the need for a central location by distributing the computation, thereby minimizing the response time in addition to lowering the operational and energy overhead of cloud-centric architectures [9].

While ML offers network intelligence and flexibility, edge computing offers execution at scale. Combined, they create a synergistic paradigm in which ML-based models are deployed on the edge to optimize real-time to create sustainable, secure, and high-performing networks. To illustrate, distributed ML and federated learning at the edge are able to manage various, high-scale IoT information and maintain privacy, and lower the bandwidth cost [10].

15.1.3 From 5G to 6G: Shifting Paradigms

The 5G to 6G upgrade is not only a technological improvement, but a paradigm shift. Whereas 5G considers improvement of mobile broadband (eMBB), URLLC, and mMTC, 6G is imagined as an AI-native network, which will implement semantic communications, holographic telepresence, digital twins, and metaverse applications [1, 4]. Such requirements require such high amounts of intelligence, autonomy, and sustainability as never before. In contrast to 5G where AI is a supporting technology, in 6G, AI and ML are built into the architecture and workings of the network itself [5]. Moreover, self-organizing and self-optimization skills will be other important in managing highly dynamic and heterogeneous environments. This change will be supported by edge computing, in conjunction with ML, and will enable ultra-low-latency

services, distributed intelligence, and energy-efficient connectivity. The convergence of these technologies can bring about more than just faster communication systems but also the potential of making these systems more sustainable, greener, and more intelligent, turning the idea of intelligent carbon-neutral networks into a practical reality.

15.2 Background and Fundamentals

The adoption of ML and Edge Computing is rapidly being identified as fundamental to the realization of intelligent, sustainable, and adaptive next-generation networks. The following section presents the basic ML methods that can be used in networks, the architecture of the edge computing, and the technologies that make it available, and relates them to the future ICT system sustainability objectives and key performance indicators (KPIs).

15.2.1 Overview of ML Techniques for Networks

The implementation of ML in communication networks has created new opportunities for the automation, optimization, and intelligent decision-making. Various groups of ML approaches have different purposes, based on the nature of the issue, availability of data, and operational need [11].

- Supervised Learning: The supervised learning method is commonly applied to tasks in which the labeled network data are known. Traffic classification, quality of service (QoS) prediction, and user behavior modeling are examples of these. Support vectors machines (SVMs), random forests, and deep neural networks (DNNs) are techniques that can be conditioned on traffic flow data to estimate the quality of service or identify malicious activities [12, 13].
- Unsupervised Learning: The unsupervised techniques give a good insight into the data when the labels provided are few, and the information on clustering, anomaly, and dimensionality reduction is important. As an example, clustering methods like k-means and DBSCAN can be used to cluster Internet of Things (IoT) devices according to their usage behavior so that adaptive service provisioning can be performed [14]. On the same note, autoencoders are employed to identify abnormal traffic or security threats without any labels defined in advance [15].
- Reinforcement Learning (RL): Resource allocation and self-optimizing networks. RL is essential to adaptive resource allocation and self-optimization [16]. Agents get to know the best strategies via interaction with the environment. Some of the uses are dynamic spectrum allocation, power control, and load balancing. RL-based schedulers in 5G/6G testbeds show high improvements in reducing latency, and optimizing throughputs [17].

Table 15.1 ML Techniques and Network Applications

ML Category	Techniques	Applications in Networks
Supervised learning	SVM, random forest, DNN	Traffic classification, QoS prediction, intrusion detection
Unsupervised learning	K-means, DBSCAN, autoencoders	Anomaly detection, clustering IoT devices, feature extraction
Reinforcement learning	Q-Learning, Deep RL	resource allocation, dynamic spectrum management, power control
Deep learning	CNN, RNN, LSTM, GANs	Channel estimation, mobility prediction, predictive maintenance

- Deep Learning (DL): DL models, convolutional neural networks (CNNs), and recurrent neural networks (RNNs) in particular have been more actively utilized in predictive maintenance, channel estimation, and mobility prediction [18]. DL facilitates the extraction of multi-dimensional features of intricate data, enhancing the precision of predictions, including wireless channel state information (CSI).

Table 15.1 gives a summarized view on the four relevant categories of machine learning and how each category can be used to make intelligent wireless and edge-enabled communication networks. SVM, Random Forest, and Deep Neural Networks are supervised learning methods used when there is labeled data that is used to accurately classify traffic, predict QoS as well as detect intrusion within contemporary networks. Unsupervised approaches to learning, such as K-means, DBSCAN, and autoencoders will be required when the labels are not known; they are used to find anomalies, group IoT devices based on their behavior, and find meaningful features in high-dimensional data. The notion of RL, which is embodied by both the Q-learning and deep RL algorithms, is especially well applied to dynamic environments, in which the network must continuously learn through interaction, allowing it to allocate resources adaptively, manage the spectrum dynamically, and control power usage dynamically in an environment with varying 5G/6G conditions. Lastly, deep learning systems like CNNs, RNNs, LSTMs, and GANs are optimized to handle complex and multi-dimensional data streams to perform activities, including channel estimation, mobility prediction and predictive maintenance. Together, the table shows that every ML category focuses on a distinct category of networking issues: supervised learning to solve tasks of labeled prediction, unsupervised learning to solve structure discovery problems, RL to solve real-time decision-making, and deep learning to solve intricate signal and data processing problems- illustrated by the multi-dimensional capability of ML to optimize next-generation sustainable communication systems.

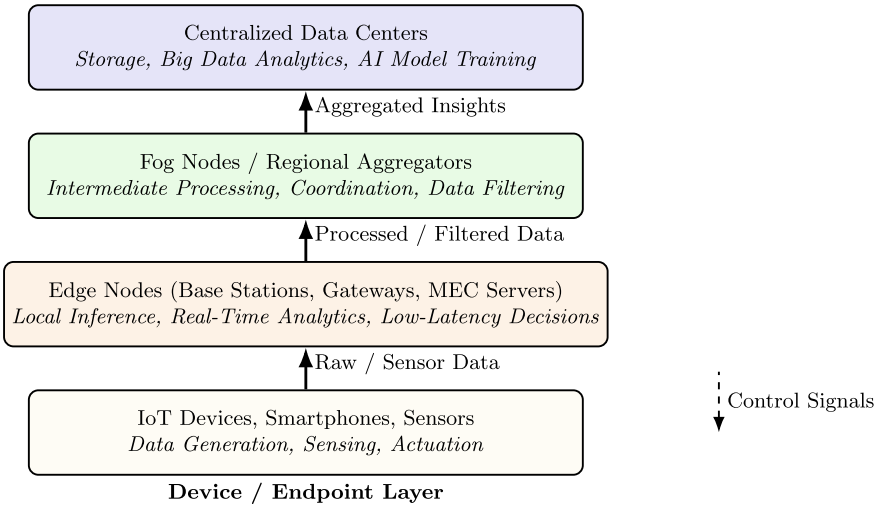


Fig. 15.1 Hierarchical edge computing architecture showing data flow from IoT devices to cloud via edge and optional fog layers

15.2.2 Edge Computing Architecture and Enablers

Edge computing is a paradigm change to the concept of decentralized processing, where computing, storage, and intelligence are physically relocated to the end-devices to reduce latency, traffic on the backhaul, and power usage. The architecture is hierarchical [19], as shown in Fig. 15.1 (the architecture consists of Devices/Endpoints, Edge, Fog, and Cloud, with each having a different set of responsibilities in the overall ecosystem). The IoT sensors, smartphones, wearables, and other data-generating components form the Device/Endpoint layer and generate raw information all the time, and need responses in near-instantaneous time because of their energy, storage, and processing capacity constraints. Right above this is an Edge Layer, which is an assembly of MEC servers, base stations, gateways, and small cells, as well as where localized processing, real-time analytics, data filtering, and lightweight ML inference are conducted. This proximity to data eliminates unnecessary point-to-cloud travels and will guarantee that time-sensitive applications, including connected autonomous vehicles, augmented reality, industrial robotics, and smart healthcare systems, will have minimal latency. The Fog Layer is an intermediate layer that is used in deployments that require multi-edge processing and where the workload needs to be balanced and the data aggregated for inter-edge communication. Lastly, the Cloud Layer is the worldwide intelligence center; it offers at-scale analytics, storage, and scale training of ML models and only takes processed insight or aggregated features of the lower layers to reduce transmission overheads and operational energy expenses.

In order to facilitate the operation of this multi-tier architecture, a number of enabling technologies are at the centre of operations. The use of virtualization and containerization models -including Docker, Kubernetes, and container orchestration engines- gives lightweight, flexible execution environments, which have facilitated the deployment of microservices and ML models without any problems across heterogeneous network nodes. The Multi-Access Edge Computing (MEC) standard of ETSI places compute units within the mobile network infrastructure so that they can provide service delivery with ultra-low-latency and offer high-bandwidth use cases. To add to this, Software-Defined Networking (SDN) and Network Function Virtualization (NFV) provide programmability, dynamic chaining of services and efficient coordination of distributed nodes. Such enablers together will make the edge computing stack a scalable, flexible, energy-conscious infrastructure that can be adapted to provide next-generation sustainable communication networks. Although Fig. 15.1 is taken to reflect the physical and functional layout of this architecture, the real working effectiveness of these systems relied on the sophisticated, intelligent management tactics. They are intelligent mechanisms that are necessary to the operation of adaptive, real-time, and sustainable operation, and they are discussed in Sect. 15.3.1.

15.2.3 Sustainability Goals in Communication Systems

Sustainability has become strategic in the architecture of the next generation communication systems where the high pressure on performance needs to be balanced by the environmental as well as the social accountability. The implementation of green and energy efficient networks is in direct accordance with the United Nations Sustainable Development Goals (SDGs) which highlights the larger objective of ICT in ensuring sustainable growth [11]. Specifically, SDG 7 highlights the need to incorporate renewable energy in base stations and edge nodes to offer affordable and clean energy services, and SDG 9 highlights the need to develop resilient and low-carbon-based infrastructures to enable innovation and industrial change. Most importantly, SDG 13 notes the urgent necessity of minimizing greenhouse gas emissions, where the ICT industry can play a key role in the sustainability of their networks and operation.

In order to monitor the progress of achieving these sustainability targets, researchers and industry stakeholders utilize a list of key performance indicators (KPIs). Energy per bit (J/bit) measures energy used to transmit each unit of data, and Power Usage Effectiveness (PUE) is used to measure efficiency in energy usage in network facilities. Carbon reduction metrics also measure savings as CO_2-equivalent, which provides a direct metric of environmental impact of network deployments [12]. Life Cycle Assessment (LCA) also needs to be applied in a holistic approach as it assesses the sustainability of communication infrastructure over its lifespan, including manufacture and deployment, operation and decommissioning [13]. This guarantees that the solutions such as edge computing might decrease the operating energy, but their combined manufacturing and maintenance imprint is also taken into consideration resulting in a reliable evaluation of the overall sustainability gain.

15.3 Synergies Between ML and Edge Computing

ML-based integration with Edge Computing is one of the most radical next-generation network design trends. Whereas ML introduces smartness in performing autonomous decision-making, edge computing offers the required infrastructure to execute with low latency and distributed performance. The convergence of them opens up new possibilities of real-time optimization, sustainability, and scalability and, therefore, this convergence is the core of intelligent 5G and AI-native 6G networks [20, 21].

The diagram 15.2 Workflow of Machine Learning and Edge Computing Synergy, is used on the end-to-end data and intelligence flow of sustainable next-generation networks. On the top, IoT sensors and devices produce raw data in the form of environmental level, mobility, or health logs. These streams are sent to local edge nodes, i.e. base stations or MEC server where local inference and real-time analytics run. This proximity saves latency and power because it is not necessary to make frequent backhaul to the cloud. The processed features are then delivered to the federated or distributed ML layer, where the devices jointly train learning models without transmitting the raw data, maintaining privacy and minimizing bandwidth consumption. Local updates in the model training are combined in the cloud or central server that optimizes and coordinates the model on a global scale across the network. Upon optimization, new improved models and energy-conscious control parameters would be returned to the edge nodes and end devices as a piece of feedback loop. This iterative process continuously optimizes the performance and makes it possible to predictively allocate resources, mitigate adaptively loads and have a small carbon footprint. In general, the diagram exemplifies the collaboration between ML and edge computing to form a closed-loop intelligent network where local knowledge, federated learning, and worldwide optimization are combined to make sure that its operations are sustainable, low-latency, and self-enhancing.

15.3.1 Intelligent Resource Allocation at the Edge

Even though Sect. 15.2.2 and Fig. 15.1 describe the hierarchical architecture in which distributed intelligence is implemented, Fig. 15.2 steps out of the hierarchical architecture and concentrates on the intelligent workflow in which data can be transmitted, the model can be trained, and decisions can be distributed throughout the network. An important issue in this ML-edge synergy is that the allocation of resources is a crucial issue, since the environments of the edges need to adapt in real time to changing traffic patterns, diverse device features, variable spectrum conditions, and low-latency application requirements [22]. Such variability cannot be handled by the traditional static allocation mechanisms that result in poor performance and waste of energy. ML and RL are used to overcome these constraints and allow the optimization of resources based on their context and needs. The RA agents distributed on edge nodes

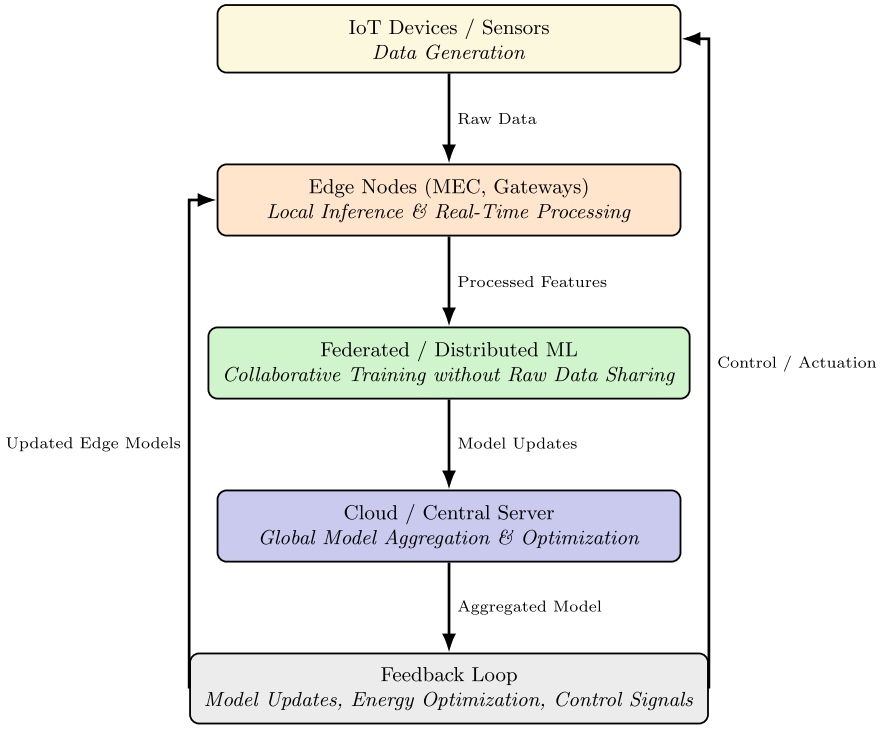


Fig. 15.2 Workflow of ML + edge synergy: local inference, federated learning, global aggregation, and feedback for sustainable, low-latency operation

monitor dynamic environmental factors like channel state information, load levels, queue time, and energy supply, and train to make optimal decisions regarding the spectrum allocation, power management, offloading tasks, and bandwidth management decisions. These agents change their policies through continuous interaction with the environment to make them perform better in terms of throughput, fairness, and latency in highly dynamic settings.

15.3.2 Distributed ML Models and Federated Learning

The other important synergy is the possibility to train ML models in the edge without collecting data centrally, via Federated Learning (FL). Edge devices do not send raw user information to a central server but instead together they train models and send only model changes, maintaining privacy and saving on bandwidth usage. Such a decentralized manner is specifically applicable in sensitive areas like healthcare, finance and mobile intelligence applications [23]. Google Gboard keyboard is a great

example of FL at scale, where model updates across millions of devices are combined to make predictions to enhance typing without subjecting a single keystroke of information. On the same note, FL-enabled medical applications enable hospitals to cooperatively learn diagnostic models without jeopardizing patient privacy. To achieve sustainable functionality, energy-conscious FL is becoming an important direction, paying attention to reducing the overhead of computation and communication of edge devices during collaborative training. FL at the edge can not only help to conduct training activities more efficiently, the reduction of redundant transmissions, and the distribution of responsibilities among training activities also allows reducing the carbon footprint of large-scale model training significantly, directing green AI goals at the same time.

15.3.3 Real-Time Analytics for Low-Latency Services

The rigid latency constraints of new applications like self-driving cars, AR/VR systems, industrial robotics, and smart factories require the co-designing of ML and edge computing. An example is that autonomous cars have to absorb high-resolution sensor data in milliseconds to drive, which means that it is impossible to entirely offload all these tasks to the cloud because of delays in the backhaul. The implementation of ML models at edge nodes near vehicular base stations enables the implementation of key tasks, like collision avoidance or switching lanes, with a sub-millisecond latency [24].

Equally, within the immersive AR/VR context, edge-based rendering and inference minimizes motion-to-photon latency and provides a smooth user experience. The industry 5.0 industrial robotics additionally uses edge-based ML to detect faults in real-time as well as predictive maintenance and control. Only through simultaneous optimization of edge infrastructure and ML algorithms is it possible to achieve such performance goals of 6G as latency less than 1 ms. This combination not only allows responsiveness in real time, but also resilience and adaptability within very dynamic networks.

15.3.4 Sustainability Angle

In addition to performance, there are far-reaching sustainability implications of the synergy of ML and edge computing. At the edge, ML algorithms can forecast traffic demand and user utilization, and can thus perform proactive resource provisioning and avoid over-provisioning of the network resources unnecessarily [25]. To illustrate, predictive models may determine periods of low demand and command base stations or edge nodes to go into sleep modes and, therefore, conserve energy without negatively affecting QoS.

Another contribution of edge computing is to minimize the energy-consuming backhaul traffic that comes with the centralized processing of clouds. Raw, high-volume data is not sent to the cloud but only processed insights or compressed features are sent to the cloud, resulting in massive energy savings on transmission. Together, these strategies can reduce the carbon footprint of ICT systems, which is consistent with the more general sustainability objectives of next-generation networks.

15.4 Sustainable Network Design

The fast growth of the communication networks has rendered sustainability as an important design consideration of next generation systems. Conventional network infrastructures can be performance-optimized but are usually power-hungry and may add a large portion of carbon footprint to the ICT industry. With the use of AI, ML, and edge computing, sustainable network design is aimed at minimizing energy usage, uniting renewable energy sources, and reducing the total carbon footprint without affecting the quality of services.

15.4.1 Energy-Efficient Communication Frameworks

Sustainable operation of a network can be achieved through energy efficiency especially when considering base stations that consume most of the energy used by a mobile network. AI-controlled models are also being used to allow coprocedent sleep states, under-used base stations or radio units can switch into low-power states when they are not needed by traffic. ML algorithms make sure that the availability of services is not hampered and the use of energy minimized by intelligently forecasting the usage patterns. Traffic-aware resource provisioning can also be used to enable networks to allocate resources dynamically, based on on-demand conditions. ML-enabled controllers can optimize the activation of channels and spectrum and computational resources rather than having to over-provision to meet peak loads. This reduces the energy used in its operations, and it is also cost-effective to the network operators.

15.4.2 Green Edge Infrastructures

Sustainable networking should also be considered when it comes to the design of infrastructure, especially on the edge computing layer. One of the opportunities is the use of renewable-powered edge nodes, in which local computation facilities are provided by the solar, wind, or hybrid power plants. This kind of diversification of

energy sources will limit the utilization of fossil-based electricity infrastructures and lead to the elimination of carbon.

Implementation of data centers that are more energy-efficient in terms of cooling is another important factor because cooling systems consume a major share of the overall energy consumption. Liquid cooling, AI-based thermal management, and waste-heat recovery can utilise techniques that can significantly reduce energy overhead. Further, by combining ML models and edge infrastructure, servers and edge resources can be activated based on demand; such that only the needed computation units are active at different workload levels. This helps in curbing wastage of idle energy and yet the needed quality of service is maintained.

15.4.3 Optimizing Carbon Footprint with ML-Driven Strategies

Optimizing carbon footprint can also be done with the help of the synergy of ML and edge computing. Among these is predictive load shifting, in which the network workloads are reallocated depending on time of day differences in energy pricing and the availability of renewable energy. As an example, algorithms that are computationally expensive may be run at times of maximum generation of renewable energy or when grid power is least costly and least carbon-intensive. Moreover, communication and computation energy joint optimization are becoming an important strategy. ML models reduce overall energy consumption in the network by considering both the radio access workload and computation workload as coupled variables. As an example, edge servers are in a position to determine whether to process or offload a task depending on the combination of the cost of computation and data transmission.

The viability of such approaches is already proved by real-life cases. ML-based models are able to recognize, and close dynamically idle or underused servers without interfering with the service delivery. This minimizes the unwanted power use, enhances energy proportionality and makes the network grow in a manner that is indeed sustainable.

15.5 Applications in Next-Gen Networks

The integration of Machine Learning (ML) and Edge Computing has provided fresh opportunities to the next-generation communication networks and presented intelligent, sustainable, and low-latency solutions in various industries. These apps also show the potential of the synergy of ML and edge technologies to change city landscapes, transportation, healthcare, and industrial ecosystems.

15.5.1 Smart Cities and IoT Ecosystems

Smart cities are being built on a network of IoT devices and edge-based intelligence to provide efficient public services and improve the quality of life of citizens. ML models interrupting sensor data in real time may be deployed at edge nodes, and allow responsive systems to manage waste, detect water leaks, and monitor pollution. To illustrate, intelligent bins with IoT sensors and edge inference can help optimize the route of waste collection, limiting the fuel usage and lowering the cost of operation. Likewise, using on-pipe sensors that combine TinyML would be able to identify leaks in early stages of urban water distribution systems, reducing non-revenue water wastage. Pollution control applications are also useful, such as distributed ML-enabled sensors monitoring air quality and detecting contamination hotspots and implementing real-time mitigation plans. Edge computing offers real-time response that is needed in keeping people safe and sustainable by minimizing the use of cloud backhaul.

15.5.2 Intelligent Transportation and Autonomous Systems

Transportation networks belong to the category of the most latency-sensitive fields, and the operation of the Connected and Autonomous Vehicles (CAVs) demands extremely quick responses to guarantee safety and efficiency. Delegation of tasks to edge servers enables vehicles to analyze data collected by cameras, radars, and LiDARs without relying on a set of centralized cloud servers. This allows real-time hazard warnings, dynamic route guidance and cooperative collision avoidance. Edge-based ML systems can also be used to optimize traffic flow, whereby all city-wide traffic signals and vehicular communications are synchronized in real time to minimize traffic congestion and emissions.

Moreover, the requirement of 6G to achieve a sub-millisecond latency opens edge-enabled vehicular networks as the backbone to serve advanced applications, including platooning, automated logistics, and drone-based delivery, to name a few. The following use cases demonstrate how the integration of ML and edge computing can help to make transportation systems safer, more sustainable, and intelligent.

15.5.3 Healthcare and Remote Monitoring

The combination of ML and edge devices has revolutionized healthcare because it allows real-time diagnostics and constant monitoring of patients. Wearables (smartwatches, medical sensors, etc.) can utilize edge inference to identify anomalies (ECG irregularities, oxygen saturation drop, and epileptic seizure early detection, etc.). As

the data are handled locally, important health alerts can be produced within a short time period which is crucial during emergencies.

Privacy-preserving healthcare applications are also eligible in edge computing. Patient confidentiality is ensured by not transferring raw data that contains sensitive information to centralized servers and, at the same time, have access to ML-based diagnostics. In rural or resource-intensive environments remote monitoring systems have increasingly become useful, where edge-based healthcare may serve to enhance access to medical services and decrease hospital workload.

15.5.4 Industrial Automation and Industry 5.0

The collaboration between humans and machines is also one of the characteristics of Industry 5.0, and in this context, both ML and edge computing contribute to the development of intelligent, adaptive, and energy-saving operations. Among the most powerful is predictive maintenance, which is an edge-based ML algorithm that evaluates sensor data on machines and identifies the early signs of machine wear, vibrations, or overheating. This ensures a reduced downtime, lower operating expenses and long life cycle of equipment.

In addition to predictive analytics, edge computing also provides real-time control and collaboration in smart factories. Edge servers can be used to coordinate tasks by autonomous systems and robots, and are designed to provide a seamless communication channel with low latency. Augmented reality (AR)-based help is useful to human operators since the ML-driven insights can be presented at the edge to make faster decisions. Combined, these innovations drive the industrial ecosystems to the stages of resilience, sustainability, and greater productivity.

15.6 Challenges and Research Opportunities

Although the convergence of ML and Edge Computing has been proven to be transformative, there are still a number of issues to be worked out in order to be deployed in next-generation sustainable networks. Solving them will play a key role in fulfilling the potential of intelligent, secure, and carbon-neutral 6G systems.

15.6.1 Scalability and Interoperability

Among the most overriding challenges is their ability to provide scalability and interoperability to non-homogeneous edge environments. The range of edge devices in terms of computational capacity, memory, and communication scale greatly, and it is challenging to develop generalized ML-based settings [26]. Moreover, the inter-

operability of the multi-vendor solutions is absent, which impedes the integration process (particularly at large scales of IoT ecosystems). Further studies should be on lightweight ML, modular edge models, and open standards to ensure effective scaling of billions of devices that are connected.

15.6.2 Privacy, Security, and Trust

With the growing presence of ML models on the edge, privacy and security concerns grow in severity. Federated learning models, although maintaining the privacy of the raw data, are still susceptible to adversarial machine learning, model inversion and data poisoning [27]. In addition, edge nodes that are installed in environments that are not trusted might be prone to damage or the intent to commit malicious activities. Blockchain and distributed ledger technologies have been suggested as a way to address these risks to provide greater transparency and guarantee a tamper-proof record of training updates [28]. Blockchain however, puts further overhead regarding energy usage and latency and this presents a trade-off that later systems must balance.

15.6.3 Data Heterogeneity and Quality Management

The other major challenge is created by data heterogeneity between devices. Non-independent and identically distributed (non-IID) data is a common occurrence in federated learning, and can adversely affect the accuracy and rate of convergence of a model [29]. Also, hardware constraints or transmission issues may result in incomplete, noisy or low-quality data being generated by the devices used in IoT. These problems are tackled by creating powerful ML models to be flexible to skewed distributions and creating highly sophisticated data preprocessing algorithms at the edge. Transfer learning, self-supervised learning, are some of the promising directions in ensuring that the model performs reliably provided that the data is heterogeneous.

15.6.4 Standardization and Regulatory Aspects

Lastly, the unavailability of standardization and regulation frameworks is an obstacle to implementation of sustainable ML-edge networks. It is only developing with organizations like 3GPP and ITU-T starting to define specifications of 5G/6G architectures, and unified specifications of green communication, energy-efficient edge design, and AI integration are also currently under development [30]. In the absence of clarity in regulation, cross-border interoperability in the global markets will be sporadic. Ethical issues, such as fairness, explainability of ML models, and adherence

to data protection laws (e.g. GDPR) should also be considered in future research. It will be necessary to develop green communication standards of 6G to bring the ICT sector in line with global climate and sustainability objectives.

15.7 Case Studies and Experimental Insights

To confirm the potential of ML and edge computing to the next-generation sustainable networks, a number of experimental testbeds, prototypes, and real-world deployments were developed. Such efforts can not just be used to show that ML-enabled edge frameworks are indeed possible, but are also a useful learning experience in the design of scalable, energy-efficient, and sustainable systems.

15.7.1 ML-Enabled Edge Frameworks for 5G/6G Testbeds

The use of open and programmable testbeds has increased the research on the topic of ML-based edge computing in next-generation networks. Much more significantly, Open Radio Access Network (O-RAN) has provided experimental environments where ML algorithms are implemented in the radio access layer to allocate dynamically the spectrum and schedule real-time. The ML-based schedulers in these testbeds have demonstrated a high degree of latency reduction, load balancing and energy savings over traditional methods.

Moreover, a number of 6G oriented experimental platforms are testing AI-native designs, where distributed ML agents are implemented at edge nodes to enable ultra-reliable low-latency communications (URLLC), and semantic communication services. These testbeds show that ML and edge computing can be used together to self-optimize the communication networks and decrease power usage and complexity of operation.

15.7.2 Sustainable Network Prototypes and Simulations

In addition to testbeds, many prototype frameworks and simulation-based experiments have been created to evaluate the potential of ML-enabled edge systems to save energy. Models that are energy- conscious based on real network traces have shown up to 30–40% energy savings in the backhaul when the computation is offloaded to local edge servers rather than a centralized cloud infrastructure.

The role of predictive ML models in the sustainability of the network is also stressful with simulation-based evaluations. Indicatively, using traffic forecasting, network components (or equivalents) can be activated or deactivated by ML-based simulators, matching the resource utilization to the demand. This and similar mea-

sures promise to enable the development of carbon-conscious 6G systems in which operational choices are not solely based on the requirements of performance but also on the sustainability indicators, e.g., energy per bit, and CO2-equivalent emissions.

15.7.3 Lessons from Real-World Deployments

A number of real world implementations can present strong evidence of the practical utility of ML at the edge. Google Stadia is one example of a platform in the cloud gaming industry that uses edge computing to provide users with high-resolution and low-latency gaming experiences. Stadia decreases the network congestion and latency by executing rendering duties nearer to the consumer, providing a scalable framework to latency-reliant services.

Likewise, Tesla Autopilot serves as an example of edge AI that is able to aid autonomous driving by local sensor data processing in vehicles. This guarantees an opportunity for real-time decision making in collision avoidance and navigation, thus the emphasis on localized intelligence in safety-critical problems. Edge deployments have also been adopted by telecommunications operators. Vodafone and AT&T are some companies that are incorporating ML-enabled edge platforms in order to enhance service delivery and cut down on operations energy. These telecom edge networks adopt distributed ML to optimize traffic, and content caching, as well as detecting anomalies, which improves user experience and network sustainability.

15.8 Future Directions

In the future, the convergence between ML and edge computing is likely to become one of the defining factors of the next generation networks development. A major trend is the development of AI-native 6G, with ML at the very fabric of network design to assist in fully autonomous and self-optimizing networks. The 6G will be engineered to use ML as a core component of the system, unlike earlier generations where AI was used as a supplementary tool, which enables the networks to dynamically adjust to environments that are highly heterogeneous.

The other potential path is the development of digital twins and the metaverse, which will enable operators to predict, optimize network performance, and simulate it in real time. The digital twins of communication structures have the potential to offer predictive planning services, providing opportunities to maintain and operate complex ecosystems in an energy-efficient and proactive manner.

At the intersection of quantum computing and ML at the edge is another frontier, which promises to deliver capabilities in cryptography, optimization, and large-scale data processing never before seen. ML models enhanced by quantum capabilities in an edge deployment can perform faster decision-making and provide communication security that is otherwise impossible in a classical environment.

Finally, these innovations are elements of a larger perspective on sustainable, smart connectivity, in which networks are not only faster, more reliable but also carbon-neutral and eco-conscious. To make this vision a reality, further interdisciplinary studies and cross-industry cooperation as well as a high level of alignment to global sustainability are to be expected.

15.9 Conclusion

This paper has discussed synergistic contends regarding the use of ML and edge computing in designing sustainable next-generation communication networks. These technologies collectively accomplish energy-efficient operations, dynamic resource management, and real-time analytics by implementing intelligence on the network edge, thus meeting the performance requirements, as well as sustainability objectives.

The discussion has pointed out the use of distributed ML models, federated learning, and predictive optimization as a way of improving efficiency and maintaining data privacy. In addition to that, ML driven resource distribution and demand-sensitive provisioning can save a lot of energy, and edge-based real-time analytics can be utilized to serve latency-sensitive applications in transportation, healthcare, smart cities, and industry. Collectively, these developments demonstrate that ML and edge computing are not only compliant to each other but also complementary to each other as enablers of the next generation network design.

The overall positive effects are deeper into society than the technical gains. Greener and smarter networks are a direct result of carbon footprint reduction in the ICT sector, and enhancement of quality of life, through quality, responsive and secure digital services. With the world becoming 6G and AI-native, it will be the integration of ML and edge computing that will enable the realization of future-ready and sustainable connectivity that can comply with global innovation and climate goals.

References

1. Zhang, Z., Xiao, Y., Ma, Z., Xiao, M., Ding, Z., Lei, X., Karagiannidis, G.K., Fan, P.: 6G Wireless networks: vision, requirements, architecture, and key technologies. IEEE Veh. Technol. Mag. **14**(3), 28–41 (2019)
2. Carlsen, L., Bruggemann, R.: The 17 United Nations' sustainable development goals: a status by 2020. Int. J. Sustain. Dev. World Ecol. **29**(3), 219–229 (2022)
3. Hasan, Z., Boostanimehr, H., Bhargava, V.K.: Green cellular networks: a survey, some research issues and challenges. IEEE Commun. Surv. Tutorials **13**(4), 524–540 (2011)
4. Saad, W., Bennis, M., Chen, M.: A Vision of 6G wireless systems: applications, trends, technologies, and open research problems. IEEE Netw. **34**(3), 134–142 (2019)
5. Gupta, A., Jha, R.K.: A Survey of 5G network: architecture and emerging technologies. IEEE Access **3**, 1206–1232 (2015)

6. Sun, Y., Peng, M., Zhou, Y., Huang, Y., Mao, S.: Application of machine learning in wireless networks: key techniques and open issues. IEEE Commun. Surv. Tutorials **21**(4), 3072–3108 (2019)
7. Noman, H.M.F., Hanafi, E., Noordin, K.A., Dimyati, K., Hindia, M.N., Abdrabou, A., Qamar, F.: Machine learning empowered emerging wireless networks in 6G: recent advancements, challenges and future trends. IEEE Access **11**, 83017–83051 (2023)
8. Mach, P., Becvar, Z.: Mobile edge computing: a survey on architecture and computation offloading. IEEE Commun. Surv. Tutorials **19**(3), 1628–1656 (2017)
9. Taleb, T., Samdanis, K., Mada, B., Flinck, H., Dutta, S., Sabella, D.: On multi-access edge computing: a survey of the emerging 5G network edge cloud architecture and orchestration. IEEE Commun. Surv. Tutorials **19**(3), 1657–1681 (2017)
10. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: challenges, methods, and future directions. IEEE Signal Process. Mag. **37**(3), 50–60 (2020)
11. Alhaddad, A.Y., Aly, H., Gad, H., Al-Ali, A., Sadasivuni, K.K., Cabibihan, J.-J., Malik, R.A.: Sense and learn: recent advances in wearable sensing and machine learning for blood glucose monitoring and trend-detection. Front. Bioeng. Biotechnol. **10**, 876672 (2022)
12. Al-Sarawi, S., Anbar, M., Alieyan, K., Alzubaidi, M.: Internet of things (IoT) communication Protocols. In: 2017 8th International Conference on Information Technology (ICIT), Amman, Jordan, 2017, pp. 685-690. IEEE
13. Chen, M., Challita, U., Saad, W., Yin, C., Debbah, M.: Artificial neural networks-based machine learning for wireless networks: a tutorial. IEEE Commun. Surv. Tutorials **21**(4), 3039–3071 (2019)
14. Hu, S., Chen, X., Ni, W., Hossain, E., Wang, X.: Distributed machine learning for wireless communication networks: techniques, architectures, and applications. IEEE Commun. Surv. Tutorials **23**(3), 1458–1493 (2021)
15. Doshi, R., Apthorpe, N., Feamster, N.: Machine learning DDoS detection for consumer internet of things devices. In: 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 2018, pp. 29–35. IEEE
16. Luong, N.C., Hoang, D.T., Gong, S., Niyato, D., Wang, P., Liang, Y.-C., Kim, D.I.: Applications of deep reinforcement learning in communications and networking: a survey. IEEE Commun. Surv. Tutorials **21**(4), 3133–3174 (2019)
17. Hu, S., Chen, X., Ni, W., Hossain, E., Wang, X.: Distributed machine learning for wireless communication networks: techniques, architectures, and applications. IEEE Commun. Surv. Tutorials **23**(3), 1458–1493 (2021)
18. Wang, T., Wen, C.K., Jin, S., Li, G.Y.: Deep learning for wireless physical layer: opportunities and challenges. China Commun. **14**(11), 92–111 (2017)
19. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
20. Pahl, C., Lee, B.: Containers and clusters for edge cloud architectures—a technology review. In: 2015 IEEE International Conference on Future Internet of Things and Cloud (FiCloud), Rome, Italy, 2015, pp. 379–386. IEEE
21. Jouini, O., Ali, K.B., Saoud, S.B., Kouki, A.: A survey of machine learning in edge computing: techniques, frameworks, applications, issues, and research directions. Technologies **12**(6), 81 (2024)
22. Wang, C.-X., You, X., Gao, X., Zhu, X., Li, Z., Zhang, C., Wang, H., Huang, Y., Chen, Y., Haas, H., Thompson, J.S., Larsson, E.G., Renzo, M.D., Tong, W., Zhu, P., Shen, X., Poor, H.V., Hanzo, L.: Artificial intelligence enabled wireless networking for 5G and beyond: recent advances and future challenges. IEEE Wirel. Commun. **27**(1), 16–23 (2020)
23. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: challenges, methods, and future directions. IEEE Signal Process. Mag. **37**(3), 50–60 (2020)
24. Letaief, K.B., Shi, Y., Lu, J., Lu, J.: Edge artificial intelligence for 6G: vision, enabling technologies, and applications. IEEE J. Sel. Areas Commun. **40**(1), 5–36 (2021)
25. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: Green edge AI: a contemporary survey. Proc. IEEE (2024)

26. Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J.: Edge intelligence: paving the last mile of artificial intelligence with edge computing. Proc. IEEE **107**(8), 1738–1762 (2019)
27. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 108, pp. 2938–2948, PMLR (2020)
28. Xu, J., Glicksberg, B.S., Su, C., Walker, P., Bian, J., Wang, F.: Federated learning for healthcare informatics. J. Am. Med. Inform. Assoc. **28**(10), 2193–2201 (2021)
29. Kairouz, P., McMahan, H.B., Avent, B., et al.: Advances and open problems in federated learning. In: Foundations and Trends® in Machine Learning, vol. 14, no. 1–2, pp. 1–210 (2021)
30. El-Moghazi, M., Whalley, J.: The ITU IMT-2020 standardization: lessons from 5G and future perspectives for 6G. J. Inf. Policy **12**, 281–320 (2022)

Chapter 16
Network Intrusion Detection Using AI: A Comparative Study and Evaluation



Binayak Koirala, Ishika Adhikari, Aradhana Behura, Abir Sen, and Pabitra Mohan Khilar

Abstract The growing complexity of cyberattacks in future-generation network infrastructures requires advanced intrusion detection systems with adaptive learning and response functions to respond to dynamic attack vectors. Various attacks can compromise the reliability of vehicular applications and services, leading to incorrect decision-making by vehicles and infrastructure, and potentially causing accidents or traffic congestion. For example, a malicious node could inject false information about traffic conditions, leading vehicles to take unnecessary detours or make unsafe maneuvers. Thus, an attacker can easily inject any leading system faults. This chapter offers a comprehensive comparative analysis of artificial intelligence(AI) techniques applied to network intrusion detection, comparing the performance of conventional machine learning techniques with deep learning models. By performing extensive experimentation across four well-known benchmark datasets—Canadian Institute for Cybersecurity Intrusion Detection System 2018 (CICIDS-2018), Canadian Institute for Cybersecurity Intrusion Detection System 2017 (CICIDS-2017), Aegean Wi-Fi Intrusion Dataset (AWID), and Network Security Laboratory Knowledge Discovery Dataset (NSL-KDD)—we comparatively evaluate Random Forest (RF), Decision Tree (DT), and Logistic Regression (LR) classifiers against Long Short-Term Memory (LSTM) networks. Empirical evaluation is conducted across

B. Koirala · A. Behura (✉) · A. Sen
School of Computer Engineering, KIIT Deemed to be University, Bhubaneswar, India
e-mail: aradhana.behurafcs@kiit.ac.in

B. Koirala
e-mail: 20051954@kiit.ac.in

A. Sen
e-mail: abir.senfcs@kiit.ac.in

I. Adhikari
Purdue University Northwest Hammond, Indiana, USA
e-mail: iadhika@purdue.edu

A. Behura · P. M. Khilar
National Institute of Technology, Rourkela, India
e-mail: pmkhilar@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_16

various network traffic scenarios, including distributed denial-of-service (DDoS) attacks, brute-force intrusions, and wireless network vulnerabilities. The results affirm the superior performance of LSTM networks, which achieved outstanding accuracy levels of 99.56, 98.87, 99.69, and 98.77% on their respective datasets, significantly outperforming traditional machine learning models. The performance of the LSTM architecture for recognizing temporal dependencies and sequential patterns in network traffic data defines its paradigmatic superiority in identifying sophisticated cyber attacks. These findings and analysis indicate the efficiency of deep learning methods to boost the security of cyber-physical systems and deal with future security issues, specifically in TCP-based networks as well as centralized and decentralized environments.

Keywords Artificial intelligence · Intrusion detection systems · Machine learning · Deep learning · Network security · Cyber-physical systems · Temporal pattern recognition · Network traffic analysis

16.1 Introduction

Intelligent decision-making systems turned out to be less efficient in fighting malware that can change its code, exploitation of confidential information, or cause other security issues. AI and ML models can be the cause of a technological revolution that can radically change the present situation if used properly. Current tools of the mentioned class are a huge leap in comparison with the old ones, in terms of the use of computational intelligence for the detection of the anomalies and the respective sources of deceit that elude the classical detection mechanisms, which is especially true for more sophisticated cyber threats in the area of critical infrastructure by way of multi-vector attacks, living-off-the-land tactics and AI-based offensive tooling. The security of vehicle communication between nodes in the networks that are based on vehicular ad-hoc networks (VANETs), is a problem brought about by their decentralized design, which is also without a fixed infrastructure and uses wireless channels; hence, the possibilities of penetrating these communications are envisaged. Great efforts for secure communication are a requirement for the successful management of traffic and road safety in VANETs, which call for the establishment of functional and prompt intrusion detection systems. What is more, security in communication is the core that holds together the trustworthiness and reliability of the exchange of information between vehicles and the infrastructure, hacking mavens are stopped from energy or intentional chaos making solutions, causing accidents or compromising the privacy of drivers and passengers. Intrusion detection systems are characterized with a myriad of roles in the detection as well as the abatement of security threats whereby, prompt responses to attack risks become activated and constantly increase VANET services. Attacks in which false data are part of the network poses to the integrity of VANETs are the most alarming, where bad nodes purposely inject falsified pieces of information into the network to disrupt the flow of traffic or cause

accidents, thus making the implementation of VANET applications less reliable and exposing the possibility of human lives [1]. In these assaults, they may execute false traffic reports, fabricate emergency alerts, or manipulate sensor data so that vehicles and infrastructure make wrong decisions. As an instance of the case, an attacker might inject false traffic congestion reports to reroute vehicles away from a specific path, thereby creating traffic jams. On the other hand, an attacker may inject fake emergency alerts to bring about a sudden braking or lane change that causes accidents probability to rise. The possibilities that result from false data injection attacks are in the farthest of the ways, which definitely tell us that security mechanisms are needed to safeguard the integrity of vehicular data. These networks may become the battlegrounds for various assaults that can cause huge losses of the network's security and reliability, along with their possible consequences: wrong decision-making by cars and road infrastructure, accidents, and, of course, disruption of traffic flow (leading to) [2, 3]. Due to the open nature of wireless channels, an intruder can intercept, modify, or disrupt data flow, leading to security threats such as data manipulation, packet injection, and denial of service attacks. First of all, accuracy and reliability in data sharing form the core of the safe and efficient operations of VANETs [4, 5]. What is more, using incorrect data can lead to suboptimal decisions that may jeopardize safety and decrease efficiency. It might be a case that a vehicle will be given a false depiction of road users, and thus, making it perform wrong lane changes or fail to keep a safe following distance, raising the chances of collisions. At the same time, provided that the infrastructures rely on false traffic reports, they might be implementing the wrong traffic management methodologies thus leading to congestion and delays. It is important to have in place intrusion detection systems that can identify and neutralize false data interference operations. Such systems should ensure the accuracy and reliability of vehicular data through checking that the messages exchanged in the network are coherent, believable, and real [6, 7]. These techniques can use signatures to verify the identities of the messages, rely upon smart systems to check the trustworthiness of sources and, on the other hand, get data from which they will decide whether it is true or false [8]. By mixing these methods, a set of devices working together can perfectly spot and remove made-up data, thus enabling both wonders and structures to still trust each other's word. As a case in point, a malicious node may fabricate an enormous number of fake traffic congestion reports for the sole purpose of detouring vehicles away from a particular route, thereby, traffic jams and delays result. Moreover, consider a scenario where a malicious node utilizes its various identities to vote multiple times in a distributed decision-making process hence, the outcome is influenced in its favor. The existence of multiple identities for a single node makes Sybil assaults specifically hard to spot and counteract. Black hole attacks occur when a hostile sensor drops all packets, thus obstructing the flow of information to the recipient and the network, isolating vehicles and therefore, preventing the dissemination of important safety information [9]. The malicious sensor in the black hole network attack, among other things, claims that it has the shortest way to the destination. It persuades the data flow because it is the one that it wants to drop all the packets it has received from the source. This may interrupt the most important communication happening between the parties like giving emergency

alerts and updating traffic information hence, vehicles losing out on the opportunity to get other important information and facing the threat to their lives [10]. Black hole attacks get particularly tricky due to the fact that it is quite hard to detect them as the malfunctioning node only drops packets without leaving a trace of suspicious network activities.

Effective mitigation of Sybil and black hole attacks requires advanced intrusion detection mechanisms capable of monitoring anomalous node behavior and identifying malicious entities for isolation from the network, in conjunction with trust management frameworks, reputation systems, and anomaly detection algorithms. Trust management systems determine the trustworthiness of a user based on their prior actions and association with other users, thus enabling the network to be aware of and isolate the users that have a track record of harmful behavior. Reputation platforms follow the reputations of the users through the opinions of rattlers and derive the collective and mutual evaluation of trustworthiness as a whole from there [11, 12]. Anomaly detection software can seek out changes in the norm of network operations and can spot the characters who undertake suspicious activity and therefore, label them. While typical machine learning models are good at finding patterns and grouping features, they are not able to recognize temporal dependence and sequential patterns of network flows—these happen to be the main characteristics of malicious processes.

Long Short-Term Memory (LSTM) architecture, being a subset of deep models, is better suited for handling sequential data and long-range dependencies [13]. The LSTM network shows better performance as compared to the existing models. The present research targets such diversified challenges by conducting rigorous comparative analysis of conventional machine learning techniques and new deep learning techniques by comparing three most popular algorithms—Random Forest, Decision Tree, and Logistic Regression—with LSTM neural networks on four large datasets: CICIDS-2018, CICIDS-2017, AWID, and NSL-KDD [2, 3, 14]. Our approach consists of rigorous preprocessing methods like K-means-based missing values imputation, high-level feature encoding, and SMOTE (Synthetic Minority Over-sampling Technique)-based handling of class imbalance issue [15] combined with Random Forest-based feature subset selection in order to attain maximum learning efficacy. Experiment results show the better performance of LSTM with 99.56% (CICIDS-2018), 98.87% (CICIDS-2017), 99.69% (AWID), and 98.77% (NSL-KDD) accuracy, that better compared with other conventional algorithms in all cases except for CICIDS-2017, where Decision Tree and Random Forest performed equally with an accuracy of 99.74%. This systematic analysis promotes artificial intelligence use for network security and offers theoretical and practical recommendations for the design of future-proof intrusion detection systems to mitigate the threat.

16.2 Related Work

Network intrusion detection is easy with the help of modern artificial technologies. This section explains some recent models, their advantages, and with limitations [16, 17]. Centralized intrusion detection systems (IDSs) involve a central server or node that collects and analyzes network traffic data from all vehicles in the VANET, providing a comprehensive view of the network and enabling effective detection of global attack patterns and coordinated malicious activities.

This design of the network allows the simplification of management and coordination since all detection and response activities are handled from a single location. As a result, this design offers a network-wide view, hence enabling effective identification of global attack patterns and coordinated malicious activities, as the central server has the privilege of accessing all the network traffic data. Security measures in intelligent transportation systems (ITS) cannot be effective without intrusion detection systems (IDS) that are efficient and prompt. On the contrary, centralized IDSs might experience problems with scalability, become a point of failure, and cause issues of increased communication, thus making them less appropriate for large-scale and highly dynamic VANETs, where network conditions can change rapidly. A distributed IDS, on the other hand, uses several devices or vehicles to keep an eye on the network traffic and detect any local intrusion, and then communicate with the neighboring nodes to get better detection results and increase the security level of the whole network. This design shares the tasks among different nodes, thus any single node is not heavily loaded, and scalability is enhanced. Compared to centralized IDSs, this architecture has better scalability, fault tolerance, and less communication overhead, thus it is more appropriate for situations where communication links may be unstable.

Federated learning systems give the freedom to vehicles to develop local intrusion detection models and only share the model updates with the central server, thus ensuring that the privacy of their local data is maintained and at the same time collaborative learning is possible without the need to compromise sensitive information [18].

Software-defined network (SDN) is a means that facilitates centralized monitoring and evaluation of network traffic and hence the discovery of global attack patterns as well as the execution of coordinated malicious activities, which most likely are hard to detect in traditional distributed networks. This central monitoring gives a more comprehensive view of network security and thus allows for more efficient threat detection. To overcome this issue, we are using deep learning with generative adversarial networks and considering distributed SDN to create a collaborative intrusion detection system (CIDS) for VANETs that allows multiple SDN controllers to collaboratively train a global intrusion detection model for the whole network without the need for direct exchange of their sub-network flows [19]. With the help of the programmable features of SDN, it is possible to dynamically deploy the security policies and the intrusion detection rules, since these can be in line with the constantly changing network threats and conditions and with that guarantee that the network

will always be secure against the most recent attacks [20]. Moreover, apart from management, performance, and scheme flexibility, SDN can significantly contribute to the provision of green services [21]. SDN-based intrusion detection systems can effectively reduce security risks and enhance the overall security status of VANETs, thus creating a vehicular communication network that is more secure and resilient. SDN brings a set of significant benefits for intrusion detection in VANETs. With the help of SDN, a centralized monitoring and analysis of the network traffic can be done, and that would help in finding global attack patterns and coordinated malicious activities. In real-time, the centralized controller can gather network traffic data from all vehicles in the VANET analyze this data to spot anomalies and suspicious behaviors. This overall network perspective allows the detection of sophisticated attacks which may evade the traditional distributed intrusion detection systems. The programmable characteristic of SDN facilitates the security policy's making and the intrusion detection rules' issuing to be carried out dynamically, thus the system may adjust itself to the ever-changing network threats and conditions. Moreover, the SDN controller is capable of making security policies and issuing intrusion detection rules on the fly depending on the real-time network situation and also the threat intelligence data. Such dynamic security adaptation makes possible much faster and more efficient security measures than tradition-fixed security policies [20].

Intrusion detection systems that are based on SDN are capable of alleviating security threats in an effective manner while also contributing to enhancements in the overall security posture of VANETs. By integrating centralized monitoring, dynamic policy execution, and programmable security services, SDN serves as a formidable platform for security enhancement in VANETs. SDN offers a lot of potentials in the creation of more secure and resilient vehicular networks [22].

Supervised learning models are dependent on the preset attack patterns; thus, they might not have the ability to detect new or modified attacks that differ from these patterns. To overcome this shortcoming, it is required that one continuously monitors network traffic, discovers new attack patterns, and changes training data and the model accordingly. The unsupervised methods rely on network traffic data analysis for the purpose of finding clusters of similar traffic samples, with the hypothesis that normal traffic samples will form separate clusters from which malicious traffic samples will differ. Through the identification of those traffic samples that belong to none of the established clusters, unsupervised learning methods are able to recognize anomalies and possible intrusions. Unsupervised learning methods are suitable for the case of new and unknown attackers, where no attack patterns exist. These methods can spot abnormal activities and deviations from the standard network behavior, thus they are capable of detecting new and unknown attacks that may not be recognized by supervised learning methods [23].

16.2.1 Machine Learning Approaches in IDS

Traditional machine learning models have demonstrated effectiveness in intrusion detection. Researchers [24] successfully used the C4.5 technique to obtain 16 essential features from benchmark datasets, but it provides false positives. Relevant features, including traffic characteristics, vehicle behavior patterns, and geographical information, provide a comprehensive view of network activity and enabling accurate identification of malicious behavior. Researchers [3] showed the efficiency of Information Gain as a feature selection technique, with an improved system performance about comparative feature importance. Hybrid methods have proved to yield promising results for effective cyber defense mechanisms. Researchers [25] presented an integrated hybrid method based on the Artificial Bee Colony (ABC) algorithm along with ensemble strategies, which achieved 99% detection rates with extremely low false positives on benchmark datasets. Hierarchical clustering methods have been used in machine learning for intrusion detection. Researchers are[26] demonstrating BIRCH algorithm integration with Support Vector Machines, providing 95.72% accuracy with 0.7% false positives and reducing training time. Semi-supervised learning models have shown new frontiers in the utilization of unlabeled data, and researcher [27] put forward fuzziness-based techniques that effectively enhanced detection performance by utilizing fuzzy membership functions for unlabeled instance labeling. Ensemble learning techniques have always yielded better results than individual classifiers. Researcher [28] also proposed a novel SVM-kNN-PSO (support vector machine-k-nearest neighbor, particle swarm optimization) ensemble technique using particle swarm optimization for parameter tuning with excellent performance on a variety of datasets. Researcher [29] also built a hybrid KPCA-SVM-GA (Kernel principal component analysis—support vector machine-genetic algorithm) model using kernel principal component analysis and genetic algorithm optimization. Additional sophisticated classification methods have also enhanced machine learning tools, and researcher [30] have taken Feature Vitality Based Reduction Method (FVBRM) into consideration using Naive Bayes classifiers, comparing three conventional feature selection methods and demonstrating that FVBRM is more effective to achieve enhanced accuracy for the classification.

16.2.2 Deep Learning Approaches in IDS

Deep learning has revolutionized the field of intrusion detection because it is capable of modeling subtle patterns and temporal relationships existing in network traffic data. Deep architectures have been more successful in bypassing machine learning constraints classically, particularly for subtle attack vectors and adaptive attacks. Recurrent Neural Network-based solutions have emerged as superior alternatives

for intrusion detection issues. Yin et al. [31] pioneered RNN-based Intrusion Detection Systems (RNN-IDS) based on temporal modeling potential to surpass traditional algorithmic limitations, breaking record-breaking accuracy rates for binary and multi-class classification problems with record-breaking performance in difficult network environments.

LSTMs have triggered significant attention due to their impressive capability to model sequence data and extract long-term temporal relationships. Staudemeyer [13] has conducted preliminary research on modeling network traffic as time series data using LSTM frameworks to demonstrate impressive efficiency on different network architectures with 93.82% accuracy and establishing LSTM as a technology of choice for temporal pattern identification in cybersecurity solutions. Hierarchical spatial–temporal frameworks are notable advances in intrusion detection with deep learning. Wang et al. [32] proposed the Hierarchical Spatial–Temporal IDS, which combines LSTM and Convolutional Neural Networks to automatically learn hierarchical spatial–temporal features from raw network traffic data, setting new standards for deep learning-powered network security solutions. Autoencoder-based methods have enhanced feature learning and dimensionality reduction in intrusion detection systems. Al-Qatf et al. [9] introduced novel deep learning techniques based on Self-Taught Learning (STL) approaches with sparse autoencoders to learn features and decrease dimensions in order to decrease training time and testing time and increase SVM classification performance. The LSTM model includes forget gates, input gates, and output gates, which can selectively store and process data and hence are important for temporal dynamic modeling of network security data.

Temporal dependencies refer to the relationships between events that occur over time, while spatial correlations refer to the relationships between events that occur in different locations. By capturing these dependencies and correlations, deep learning approaches can identify subtle patterns of malicious activity that may be missed by traditional machine learning techniques. However, training deep learning models on large datasets can be computationally expensive, requiring significant processing power and memory. To overcome these limitations, researchers are exploring the use of cloud-based platforms and distributed training approaches, which allow for the parallel processing of large datasets and the distribution of the training workload across multiple machines.

16.3 Methodology

This research design focused on a lightweight machine learning pipeline that is able to handle intrusion detection in vehicular ad hoc networks (VANETs).

Figure 16.1 represents the design standard pipeline for comparative analysis, systemic handling workflow steps of raw information, data acquisition, to preprocessing methodologies (missing value imputation, data encoding, normalization, SMOTE algorithm-based class balance) to multi-folded branches for traditional

machine learning and deep learning architecture training to extensive evaluation models with performance evaluation.

The methodological framework facilitates systematic comparison based on standardized preprocessing and performance measures, eliminating potential confounds while remaining adaptable to specific datasets and model characteristics. The inclusion of feature selection techniques, hyperparameter tuning, and cross-validation schemes ensures the optimum model performance conditions, providing insightful comparative strengths and weaknesses between classical machine learning and deep learning strategies for intrusion prediction systems.

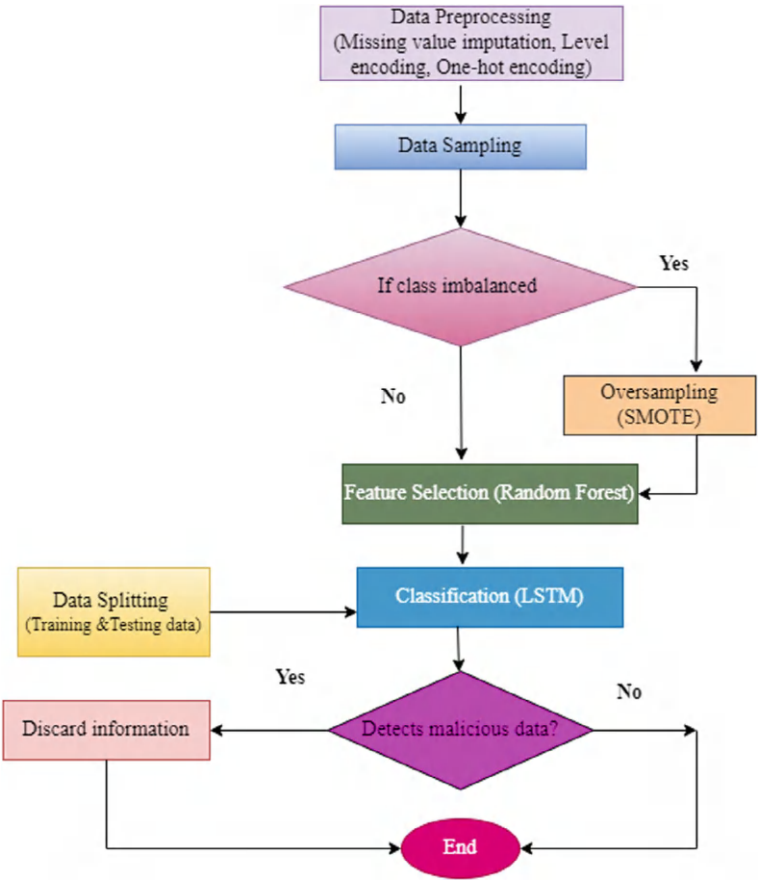


Fig. 16.1 Methodology flow diagram

16.3.1 Data Preprocessing

Data preprocessing is important for intrusion detection. Firstly, we have to check the quality of information available. The quality in terms of missing information is available or not, data set is balanced or not. If it is not balanced one how to balance. To handle missing data, various missing value imputation technique can be used such as z-score, decimal scaling and min–max technique. The CICIDS-2018 and CICIDS-2017 datasets provides information for the various network traffic attack trends such as Distributed Denial of Service (DDoS), brute-force attacks, web attacks, and botnet communications, representing contemporary threat patterns observed in enterprise networks. The NSL-KDD provides DDoS, DoS, User-to-Root (U2R), Remote-to-Local (R2L), and Probe attacks, and acts as a standardized benchmark in the existing literature. The AWID dataset adds wireless network intrusion scenarios to the evaluation domain by covering WiFi-specific attacks like de-authentication floods, evil twin attacks, and wireless injection vulnerabilities, thus making the models applicable to heterogeneous network infrastructures and new wireless security threats.

Preprocessing begins by conducting strenuous data quality checks and cleaning procedures to resolve missing values, inconsistent data types, outliers, and unnecessary features that can diminish model potential and trainability. The following algorithms list the stepwise procedure for all of the preprocessing activities:

Algorithm 1: Missing Value Imputation using K-means

1: Initialize k clusters using the K-means algorithm on the complete data
2: Assign each data point with missing values to the nearest cluster centroid
3: Replace missing values with the mean of non-missing values in the assigned cluster
4: Repeat steps 2-3 until convergence

Algorithm 1 employs a K-means cluster-based approach for missing value imputation that preserves data distribution and statistical consistency by grouping similar instances and imputing values based on cluster centroids. The iterative process initializes k clusters from complete data records, assigns missing data points to nearest cluster centers, fills missing values with cluster means, and iterates until convergence. This method surpasses simple mean or median imputation by considering multi-dimensional feature correlations to maintain natural network traffic patterns and original data distribution properties. Level encoding is described in Algorithm 2, which is used in categorical variables where data ordering is inherent.

Algorithm 2: Level Encoding using K-means

1: Initialize k centroids randomly
2: Assign each category to the nearest centroid
3: Encode each category with the index of its assigned centroid

Algorithm 3: One-Hot Encoding using K-means

```
Initialize k centroids randomly
Assign each category to the nearest centroid
Create a binary vector of length k for each category
Set the element corresponding to the assigned centroid to 1, and all others to 0
```

For nominal categorical variables without ordering, Algorithm 3 uses one-hot encoding with K-means clustering to create binary vector representations in which each category is assigned to the nearest centroid, and a binary vector of dimensionality k is created with the component for the assigned centroid set to 1 and the others to 0. This two-pronged encoding strategy ensures the optimal representation of categorical data without sacrificing computational efficiency and without sacrificing the semantic relationships in the data, particularly helpful for high-cardinality categorical features characteristic of network traffic data.

Algorithm 4: SMOTE (Synthetic Minority Over-sampling Technique)

```
Input: Minority class samples X, number of synthetic samples to generate N, number of
nearest neighbors k
Initialize an empty set S
for each sample x in X do
    Find k nearest neighbors of x in X using Euclidean distance
    Choose one of the k nearest neighbors, xnn
    Generate N synthetic samples between x and xnn:
      s = x + rand(0,1) × (xnn − x)
    Add s to S
end for
Output: Set S containing synthetic samples
```

The SMOTE technique is shown in Algorithm 4. This technique generates synthetic instances of minority intrusion classes by interpolating among existing minority classes and finding their k nearest neighbors in the feature space. The model iterates over the minority samples systematically, identifies their k nearest neighbors according to Euclidean distance metrics, and generates synthetic samples.

Algorithm 5: Random Forest Feature Selection

```
Input: Dataset D, Number of trees N
Train a Random Forest classifier with N trees using D
Calculate feature importance scores based on the mean decrease in impurity (MDI)
  or mean decrease in accuracy (MDA)
Rank features based on their importance scores
Select the top k features with the highest importance scores
```

Feature selection applies Random Forest-based importance score (Algorithm 5) to obtain discriminative features for intrusion detection through learning a Random Forest classifier with N decision trees and calculating feature importance scores using mean decrease in impurity (MDI) or mean decrease in accuracy (MDA) as the metric. The algorithm verifies the contribution of each feature to global classification performance by measuring predictive accuracy loss by removing or randomizing a feature, ranking features with importance scores and selecting the top k features

with the highest discriminative power to train the model, thereby reducing computational load, avoiding the curse of dimensionality, and enhancing interpretability by emphasizing network traffic attributes distinguishing between malicious and benign communications.

16.3.2 Machine Learning Model Implementation

Machine learning model deployment phase constitutes a central component of the comparative study involving three traditional classification models: Logistic Regression, Decision Tree, and Random Forest. The models were selected because they are widely applied in cybersecurity solutions, algorithmic in nature which are complementary to each other, and proven to perform well for binary and multi-class classification tasks. Each of the algorithms represents a unique pattern recognition and decision-making technique, with Logistic Regression providing probabilistic linear classification, Decision Trees providing interpretable rule-based decision, and Random Forest providing ensemble-based robustness by bootstrap aggregation.

16.3.2.1 Logistic Regression

Figure 16.2 depicts about the methodology for Logistic Regression. Logistic Regression uses the sigmoid activation function as its mathematical basis, mapping linear combinations of input features onto probability which estimates the range 0 to 1, as illustrated in Fig. 16.3. The logistic formula is expressed as:

$$p(x) = \frac{1}{(1 + e^{-z})} \tag{16.1}$$

where z is the weighted linear combination of the features ($z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots. + \beta_2 x_2$), and β_0, β_1, β_2,., β_n are the coefficients (weights) on the features, integrating the bias parameter(β_0). Algebraically, based on the logistic function formula, z is the weighted sum of the features, given by:

$$Z = \theta_0 + \theta_1 x_1 + \theta_1 x_1 + \theta_2 x_2 \tag{16.2}$$

The θ_0 part of the formula is the bias term and a form of constant shift, and it makes the decision boundary slide proportionally. The sigmoid function, mathematically given by:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{16.3}$$

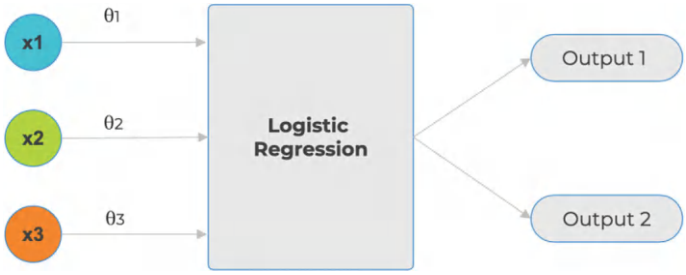


Fig. 16.2 Logistic regression methodology

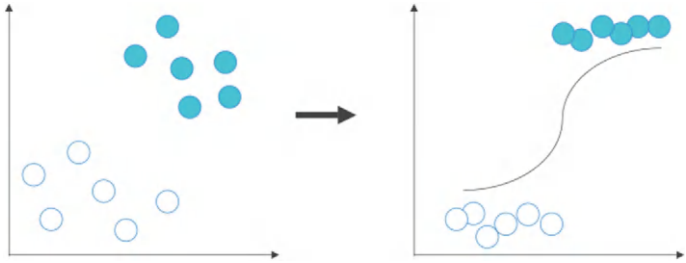


Fig. 16.3 Sigmoid function

where z is the weighted feature linear combination, performs all-critical non-linear mapping for binary classification. Logistic Regression addresses cybersecurity data problems with probabilistic classification that provides a quantifiable estimation of decision confidence, enabling security analysts to modulate false positive/negative trade-offs via thresholding. Parameter estimation employs maximum likelihood with gradient descent or Newton–Raphson optimization to drive logistic loss minimization optimization. L1 (Lasso) and L2 (Ridge) regularization are employed to prevent overfitting for high-dimensional network traffic features. Multi-class intrusion detection is implemented by one-versus-rest or multinomial approaches for computational efficiency and numerical stability.

16.3.2.2 Decision Tree

Figure 16.4 illustrates the Decision Tree algorithm from root node initialization to recursive splitting through to terminal leaf node classification, with systematic feature selection in internal nodes, threshold calculation for continuous variables, and homogeneous leaf node construction for decision making. Such visualization demonstrates Decision Tree interpretability through explicit rule-based path construction that is understandable and verifiable by cybersecurity experts, rendering it highly attractive for intrusion detection systems requiring decision transparency to conduct

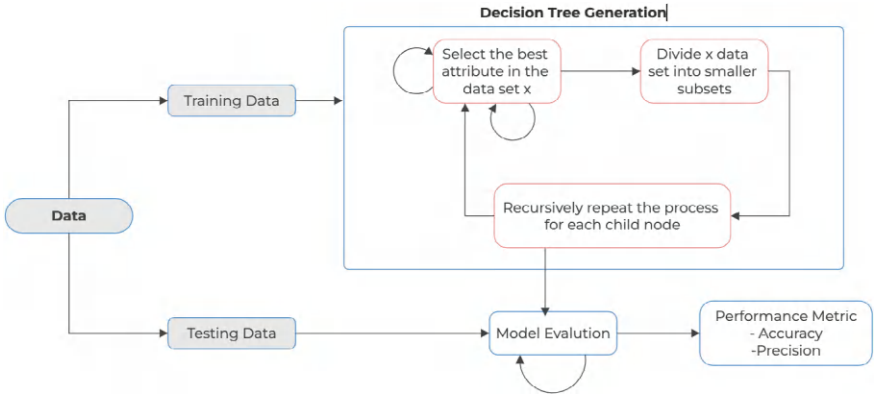


Fig. 16.4 Decision tree methodology

forensic analysis and rule tuning. Decision Tree application employs sophisticated splitting criteria with information-theoretic estimators like information gain and entropy to construct decision boundaries with the best class separation and lower tree complexity.

- **Entropy:**

$$\mathrm{H(S)} = -\sum_i p_i \log_2(p_i) \tag{16.4}$$

where p_i is the relative number of instances of class i in set S, quantifies the impurity or homogeneity of a node.

- **Information gain:**

$$IG(S, A) = H(S) - \sum_v \frac{|S_v|}{|S|} \times H(S_V) \tag{16.5}$$

measures the reduction of entropy realized by dividing set S based on attribute A, where S_v is the subset of S for which attribute A has value v.

- **Gini impurity measure:**

$$\mathrm{Gini(S)} = 1 - \sum_i p_i^2 \tag{16.6}$$

which gives computationally efficient impurity evaluation, and the chi-square statistic for categorical features, enabling the model to deal with heterogeneous features encountered in traffic data.

The construction process employs pruning techniques (pre-pruning using minimum sample/depth constraint, post-pruning using cost-complexity pruning) to prevent overfitting. Dynamic threshold choosing uses binary search for continuous

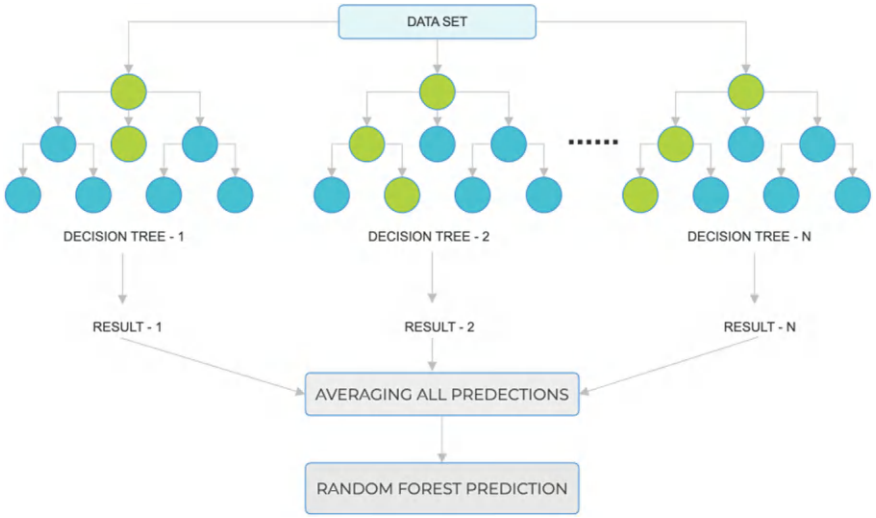


Fig. 16.5 Random forest methodology

features, while categorical features use subset enumeration or binary encoding. Hyperparameter tuning optimizes tree depth, leaf/split samples, and splitting thresholds by cross-validation across network intrusion settings.

16.3.2.3 Random Forest

Figure 16.5 describes the procedure for Random Forest's ensemble learning architecture, integrating individual decision trees via bootstrap aggregation and important feature selection randomly. The structure shows how trees are trained on various feature aggregate predictions through majority voting mechanism, focusing variance reduction and suitability for complex network intrusion detection with high-dimensional datasets.

16.3.3 Deep Learning Model Implementation: LSTM

The work function of the Long Short-Term Memory (LSTM) model is perfect in transitioning from machine learning techniques to deep learning architecture for network intrusion classification. This architecture is better at handling temporal relationships and sequential network traffic information as compared to traditional machine learning techniques. and therefore, are best suited for the detection of dynamic and changing attack structures [33].

Deep learning handles some of the limitations of traditional machine learning technique in cybersecurity domain. It automatically extracts hierarchical feature from raw traffic information. It enables the model to discover subtle patterns and interdependencies difficult for human analysts to detect, perhaps revealing new attack signatures and zero-day exploits. Besides, the recurrent structure of LSTM networks accommodates variable-length input sequences in a way that is compatible with the naturally variable length of network sessions and communication streams without resorting to fixed-size feature vectors that may compromise useful temporal information [34–36].

Figure 16.6 graphically represents the internal structure of an LSTM cell with the sophisticated gating mechanisms for selective remembering and forgetting of information along temporal sequences. The figure shows the most important components of the LSTM unit, i.e., the cell state c(t-1) for long-term memory, the hidden state h(t-1) containing short-term information of the last step, and the input x(t) at the current time step. The figure clearly depicts the three crucial gates: the Forget Gate (depicted in red dashed lines) that determines what to throw away from the previous cell state through sigmoid activation, the Input Gate (depicted in red dashed lines) made up of sigmoid and tanh components which regulate adding new information, and the Output Gate (depicted in red dashed lines) that regulates information to send to the next time step. The figure shows the element-wise operations like multiplication (×) and addition (+) for blending gated information, the results of which are the new cell state c(t) and the new hidden state h(t). This type of design enables LSTM networks to retain useful information in long sequences and disregard irrelevant information, making it extremely suitable to identify temporal attack patterns in network traffic data where immediate malicious activity could be dependent on the events in earlier time steps.

The LSTM architecture uses a gating mechanism to either recall or forget data from historic time steps. The gating technique has three gates, such as the forget gate,

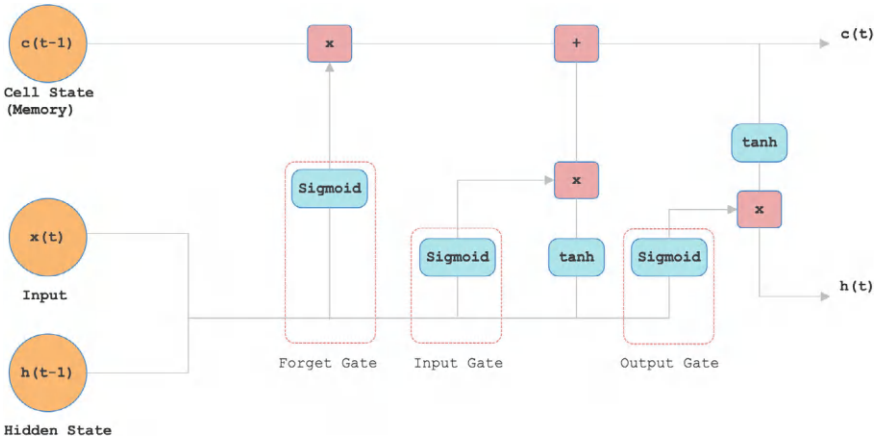


Fig. 16.6 Long term short term (LSTM) methodology

the input gate, and the output gate, which are responsible for controlling the flow of information into and out of the cell state.

- **Forget Gate**: The forget gate determines whether to retain or delete data from the past cell state using a sigmoid activation function to produce values between 0 and 1. This gate estimates the relevance of historic state data and evaluates whether the temporal patterns should be retained for recent prediction decisions.
- **Input Gate**: This gate handles the relevant data that needs to be stored in the cell, integrating the current input and the historic hidden state. This mechanism determines which new network traffic features should be incorporated into the long-term memory while the candidate cell state generates potential new information for storage.
- **Output Gate**: The output gate controls the information that should be passed to the next time step, combining the current cell state and the previous hidden state. This gate determines which portions of the cell state should influence the current output and be propagated to subsequent time steps in the intrusion detection sequence.

Algorithm 6: LSTM Algorithm for Intrusion Detection

```
Input:
- Training dataset: Dtrain
- Testing dataset: Dtest
Initialize the LSTM model with appropriate architecture and parameters.
Train the LSTM model using the training dataset Dtrain.
for each sample (x, y) in the testing dataset Dtest do
    Predict the class label ŷ using the trained LSTM model.
    Compare ŷ with the true class label y.
end for
Compute the performance metrics such as accuracy, precision, recall, and F1-score for the intrusion detection task.
Output the performance metrics.
```

Algorithm 6 outlines the overall process for using LSTM networks for network intrusion detection problems, including the entire process from model setup to performance evaluation. The LSTM architecture utilizes multiple stacked layers for hierarchical temporal pattern learning at different abstraction levels, with dropout regularization preventing overfitting and improving generalization to novel attack patterns. The classification layer uses softmax (multi-class) or sigmoid (binary) activation functions for probability-based intrusion prediction. The hyperparameter optimization uses learning rates, batch sizes, dropout values, and training epochs through gradient descent, grid search, or Bayesian optimization techniques.

16.4 Experimental Analysis and Discussion

Assessment of classical machine learning algorithms makes the backbone of understanding baseline performance capabilities of network intrusion detection systems. Three algorithmic models, Random Forest, Logistic Regression, and Decision Tree, were extensively tested on four robust benchmark datasets to provide comparative performance baselines as well as understand algorithmic abilities in various network security settings.

Table 16.1 demonstrates the performance in terms of accuracy as compared with various machine learning algorithm on the four benchmark datasets. The performance is seen to have very high variability with Random Forest providing the highest consistency across datasets, achieving over 93% accuracy in all instances. Decision Tree is as consistent as Random Forest in the CICIDS-2017 dataset with the same 99.74% accuracy, while Logistic Regression is in low variability with particular trouble with the AWID dataset where it achieves just 59.39% accuracy.

The graphical representation in Fig. 16.7 easily illustrates the performance variations of the three machine learning algorithms over different datasets. The bar chart indicates that CICIDS-2017 is the top-performing dataset among all models with the minimum performance variations between algorithms. Conversely, the AWID dataset provides the most difficult classification task, particularly for Logistic Regression, with a drastic performance degradation compared to tree-based algorithms. This figure captures the highest importance of algorithm selection based on some network environments and attack vector features.

End-to-end analysis identifies that ensemble-based approaches, specifically Random Forest, are stronger and more reliable in various network security scenarios. The generalization capability of the algorithm by combining several decision boundaries through its ensemble framework is best in the scenario of environments having varied network and attack patterns. Decision Tree algorithms, though efficient computationally and understandable, are better suited to the nature of the dataset, with phenomenal performance in structured clean attack patterns but poor performance in dirty environments. Logistic Regression, as computationally and interpretable as it can be, is woefully inadequate when faced with non-linear attack patterns, specifically within the realm of wireless network environments where intrusion activity is involved with complex temporal and spatial patterns that cannot be mapped accurately by linear decision boundaries.

Table 16.1 Accuracy of various machine learning architecture on intrusion datasets

Dataset	Random forest (%)	Logistic regression (%)	Decision tree (%)
CICIDS-2018	95.20	95.34	92.34
CICIDS-2017	99.74	98.37	99.74
AWID	93.03	59.39	88.16
NSL-KDD	97.36	86.37	97.21

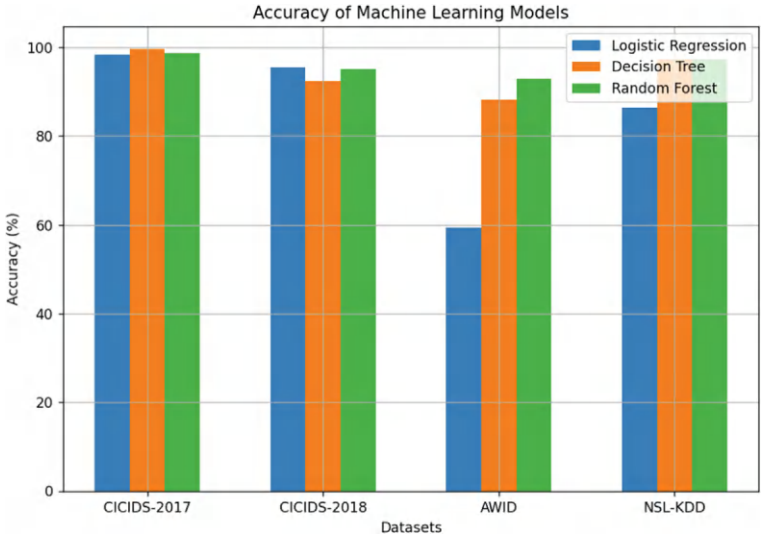


Fig. 16.7 Accuracy of machine learning models using various datasets

LSTM architecture with specialized gating components and memory cell structure was thoroughly experimented on the same four benchmark datasets in order to ascertain if it is capable of modeling advanced attack behaviors that otherwise get evaded by conventional machine learning models.

Table 16.2 illustrates the high accuracy of the LSTM architecture. Figure 16.8 visualization illustrates strong consistency of LSTM performance across different network security conditions, with measurements of accuracy tightly grouped in a narrow range from 98.77 to 99.69%. This high-performance level consistently is also contrary to fluctuating patterns of performance with conventional machine learning approaches and most importantly the performance advantage of LSTM over AWID dataset compared to substantial loss in performance by baseline algorithms. The detection accuracy is clearly represented by the bar graph on how LSTM has improved detection accuracy regardless of complexity in datasets or heterogeneity in attack vectors.

Reason for higher precision of LSTM for all datasets is its advanced architectural design, including forget gates, input gates, and output gates for selective storage of data and identification of temporal patterns. The gating of the network enables it to

Table 16.2 Accuracy table of LSTM on various datasets

Dataset	LSTM accuracy (%)
CICIDS-2018	99.56
CICIDS-2017	98.87
AWID	99.69
NSL-KDD	98.77

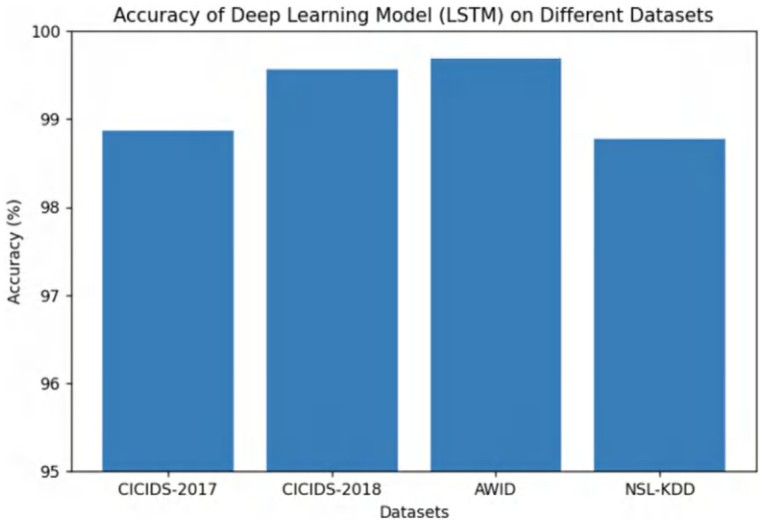


Fig. 16.8 Accuracy of deep learning model (LSTM) on different datasets

hold contextual data for large durations of time and to forget irrelevant information, making the network particularly appropriate for modeling network traffic behavior in which attack signs extend over a few steps of time. The consistency of high precision for different datasets indicates that the capability of LSTM to model sequential relationships and distant correlations provides a distinct benefit for known and unknown attack pattern identification, making it a promising foundation for next-generation intrusion detection systems that must keep up with emerging cybersecurity threats.

Table 16.3 presents the comparative overall performance across all of the models and datasets examined, with distinct performance trends that underscore each algorithmic approach's relative strengths.

Figure 16.9 depicts the performance of various datasets with various models. The graph indicates LSTM model outperforms as compared to the other models. Figure 16.10 represents the cumulative analysis of various models.

Table 16.3 Comparison of different models on different datasets

Dataset	Random forest (%)	Logistic regression	Decision tree	LSTM
CICIDS-2018	95.20	95.34	92.34	99.56
CICIDS-2017	99.74	98.37	99.74	98.87
AWID	93.03	59.39	88.16	99.69
NSL-KDD	97.36	86.37	97.21	98.77

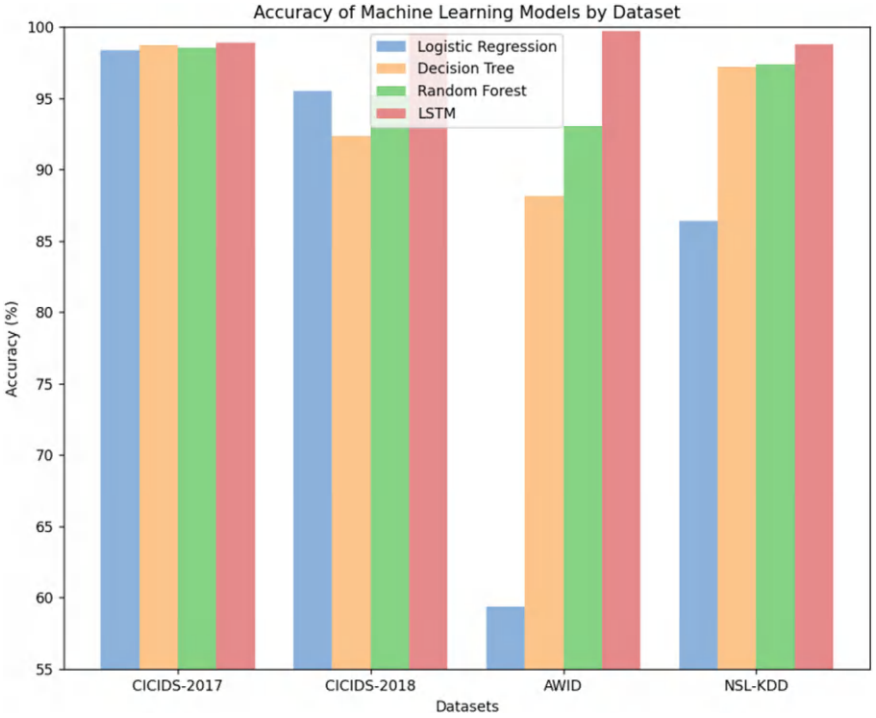


Fig. 16.9 Combined accuracy bar graph

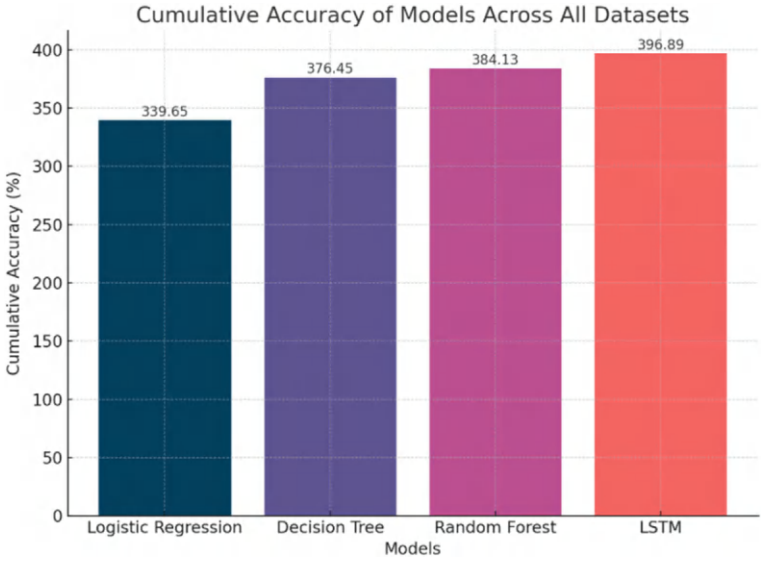


Fig. 16.10 Cumulative accuracy bar graph

16.5 Conclusion

Efficient intrusion identification of real-time security attacks within VANETs has to be the highest priority for the reason that a quick reaction to an attack is possible by the security team, thus, it prevents the spread of the damaging impact and the disturbance of the network. Maintaining the safety and reliability of vehicular communications is only possible through the real-time detection and response to security threats. So, the whole situation is a matter of life and death. A delay in the detection and response phases may result in accidents, traffic congestion, or other effects that might be serious indeed. First of all, the low-latency requirement of the entire process means that data analysis should be efficient and that alert generation should be quick. The intrusion prevention system ought to be fitted with a mechanism that will enable it to carry out this data processing in a manner both swift and efficient. Finally, by producing alerts at the right time it becomes possible for security officers to act quickly against any illegal access. The problem of the need for real-time response to intrusion detection in a VANET is related to dynamic network conditions and has a large volume of data in it. Because of the network architecture changes and the vehicle mobility, the IDS must not only rethink the scenario, but also get a grasp of the ever-changing situation. On the other hand, the huge amount of data coming from the VANETs calls for new data processing and analyzing methods which will be up to the task of this huge amount of information. Moreover, Real-time intrusion detection in VANETs necessitates a judicious mix of high-level algorithms, top-notch hardware, and well-managed network protocols. The detailed comparative analysis serves as a reference point for the researchers who are new to the field of artificial intelligence mechanisms. Feature engineering could be a powerful tool for enhancing the effectiveness of vehicular intrusion detection systems to a great extent and enabling them to detect more widespread attacks as well as modify themselves according to different network scenarios. It may involve, for instance, the fusion of several properties, the data transformation, or the building of new metrics that represent the relevant aspects of network behavior such as packet loss rate or the frequency of route changes. Cost-sensitive classification involves assigning different costs to the misclassification errors that occur when instances of the malicious class are classified as the normal class and vice versa, i.e., penalizing the misclassification of the attack instances more than that of the normal ones and encouraging the model to focus primarily on the detection of the former. This method can be put to optimal use when dealing with data imbalance problems because it focuses on the minority class. Multiple benchmark datasets have been described such as CICIDS-2018, CICIDS-2017, AWID, and NSL-KDD. Besides, on these datasets, it is demonstrated that Long Short-Term Memory (LSTM) networks outperform the most popular machine learning methods in the event of a cyberattack. This chapter presents an overview of various recent scientific publication surveys, their pros, and cons. The work can be extended for generative AI, and performance can be measured in terms of accuracy [33], precision [34], recall, F1-score [35], G-means, kappa, MSE, RMSE [36], and MAPE.

References

1. Karbir, R., Onik, A.R., Samad, T.: A network intrusion detection framework based on bayesian network using wrapper approach. Int. J. Comput. Appl. **166**(4), 13–17 (2017). https://doi.org/10.5120/ijca2017913992
2. Maseer, Z.K., Yusof, R., Bahaman, N., Mostafa, S.A., Foozy, C.F.M.: Benchmarking of machine learning for anomaly-based intrusion detection systems in the CICIDS2017 dataset. IEEE Access **9**, 22351–22370 (2021). https://doi.org/10.1109/ACCESS.2021.3056614
3. Kurniabudi, D.S., Darmawijoyo, M.Y.B., Idris, A.M., Bamhdi, R., Budiarto, B.: CICIDS-2017 dataset feature analysis with information gain for anomaly detection. IEEE Access **8**, 132911–132921 (2020). https://doi.org/10.1109/ACCESS.2020.3009843
4. Hosseini Bamakan, S.M., Wang, H., Yingjie, T., Shi, Y.: An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization. Neurocomputing **199**, 90–102 (2016). https://doi.org/10.1016/j.neucom.2016.03.031
5. Chung, Y.Y., Wahid, N.: A hybrid network intrusion detection system using simplified swarm optimization (SSO). Appl. Soft Comput. **12**(9), 3014–3022 (2012). https://doi.org/10.1016/j.asoc.2012.04.020
6. Kabir, E., Hu, J., Wang, H., Zhuo, G.: A novel statistical technique for intrusion detection systems. Fut. Generat. Comput. Syst. **79**(1), 303–318 (2018). https://doi.org/10.1016/j.future.2017.01.029
7. Kavitha, B., Karthikeyan, S., Sheeba Maybell, P.: An ensemble design of intrusion detection system for handling uncertainty using neutrosophic logic classifier. Knowl. Based Syst. **28**, 88–96 (2012). https://doi.org/10.1016/j.knosys.2011.12.004
8. Koc, L., Mazzuchi, T.A., Sarkani, S.: A network intrusion detection system based on a Hidden Na¨ıve Bayes multiclass classifier. Exp. Syst. Appl. **39**(18), 13492–13500 (2012). https://doi.org/10.1016/j.eswa.2012.07.009
9. Al-Qatf, M., Lasheng, Y., Alhabib, M., Al-Sabahi, K.: Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. IEEE Access **6**, 2169–3536. https://doi.org/10.1109/ACCESS.2018.2869577
10. Ji, S.-Y., Jeong, B.-K., Choi, S., Jeong, D.H.: A multi-level intrusion detection method for abnormal network behaviors. J. Netw. Comput. Appl. **62**, 9–17 (2016). https://doi.org/10.1016/j.jnca.2015.12.004
11. Kim, G., Lee, S., Kim, S.: A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Exp. Syst. Appl. **41**(4), 1690–1700 (2014). https://doi.org/10.1016/j.eswa.2013.08.066
12. Kou, G., Peng, Y., Chen, Z., Shi, Y.: Multiple criteria mathematical programming for multi-class classification and application in network intrusion detection. Inform. Sci. **179**(4), 371–381 (2009). https://doi.org/10.1016/j.ins.2008.10.025
13. Staudemeyer, R.C.: Applying long short-term memory recurrent neural networks to intrusion detection. South Afric. Comput. J. **56**(1), 136–154 (2015). https://doi.org/10.18489/sacj.v56i1.248
14. Ranjit, P., Borah, S.: A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems. Int. J. Eng. Technol. **7**, 479–482 (2018)
15. Aljawarneha, S., Aldwairi, M., Yasseina, M.B.: Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. J. Comput. Sci. **25**, 152–160 (2018). https://doi.org/10.1016/j.jocs.2017.03.006
16. Liao, H.J., Richard Lin, C.H., Lin, Y.C., Tung, K.Y.: Intrusion detection system: a comprehensive review. J. Netw. Comput. Appl. **36**(1), 16–24 (2013). https://doi.org/10.1016/j.jnca.2012.09.004
17. Inayat, Z., Gani, A., Anuar, N.B., Khan, M.K., Anwar, S.: Intrusion response systems: foundation, designs, and challenges. J. Netw. Comput. Appl. **62**, 53–74 (2016). https://doi.org/10.1016/j.jnca.2015.12.006

18. Catania, C.A., Bromberg, F., Garino, C.G.: An autonomous labelling approach to support vector machines algorithms for network traffic anomaly detection. Exp. Syst. Appl. **39**(2), 1822–1829 (2012). https://doi.org/10.1016/j.eswa.2011.08.068
19. Khana, K., Mehmooda, A., Khana, S., Khana, M.A., Iqbal, Z., Mashwani, W.K.: A survey on intrusion detection and prevention in wireless ad-hoc networks. J. Syst. Architect. **105**, 101701 (2020). https://doi.org/10.1016/j.sysarc.2019.101701
20. Dash, T.: A study on intrusion detection using neural networks trained with evolutionary algorithms. Soft. Comput. **21**, 2687–2700 (2017). https://doi.org/10.1007/s00500-015-1967-z
21. Xu-sheng, G., Jing-shun, D., Jia-fu, W., Wei, C.: Anomaly Intrusion detection based on PLS feature extraction and core vector machine. Knowl. Based Syst. **40**, 1–6 (2013). https://doi.org/10.1016/j.knosys.2012.09.00
22. Seyed, M.H., Bamakan, B., Amiri, M., Mirzabagheri, Y.: A new intrusion detection approach using pso based multiple criteria linear programming. Procedia Comput. Sci. **55**, 231–237 (2015). https://doi.org/10.1016/j.procs.2015.07.040
23. Huang, C.L., Dun, J.F.: A distributed PSO–SVM hybrid system with feature selection and parameter optimization. Appl. Soft Comput. **8**(4), 1381–1391 (2008). https://doi.org/10.1016/j.asoc.2007.10.007
24. Popoola, E., Adewumi, A.: Efficient feature selection technique for network intrusion detection system using discrete differential evolution and decision tree. Int. J. Netw. Secur. **19**(5), 660–669 (2017)
25. Hajisalem, V., Babaie, S.: A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. Comput. Netw. **136**, 37–50 (2018)
26. Horng, S.-J., Su, M.-Y., Chen, Y.-H., Kao, T.-W., Chen, R.-J., Lai, J.-L., Perkasa, C.D.: A novel intrusion detection system based on hierarchical clustering and support vector machines. Exp. Syst. Appl. **38**(1), 306–313 (2011). https://doi.org/10.1016/j.eswa.2010.06.066
27. Ashfaqa, R.A.R., Wanga, X.Z., Huanga, J.Z., Abbas, H., Hea, Y.L.: Fuzziness based semi-supervised learning approach for intrusion detection system. Inform. Sci. **378**, 484–497 (2017). https://doi.org/10.1016/j.ins.2016.04.019
28. Aburomman, A.A., Reaz, M.B.I.: A novel SVM-kNN-PSO ensemble method for intrusion detection system. Appl. Soft Comput. **38**, 360–372 (2015). https://doi.org/10.1016/j.asoc.2015.10.011
29. Kuang, F., Xu, W., Zhang, S.: A novel hybrid KPCA and SVM with GA model for intrusion detection. Appl. Soft Comput. **18**, 178–184 (2014). https://doi.org/10.1016/j.asoc.2014.01.028
30. Mukherjee, S., Sharma, N.: Intrusion detection using naive bayes classifier with feature reduction. Procedia Technol. **4**, 118–128 (2012). https://doi.org/10.1016/j.protcy.2012.05.017
31. Yin, C., Zhu, Y., Fei, J., He, X.: A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access **5**, 21954–21961 (2017). https://doi.org/10.1109/ACCESS.2017.2762418
32. Wang, W., Sheng, Y., Wang, J., Zeng, X., Ye, X., Huang, Y., Zhu, M.: HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. IEEE Access **6**, 1792–1806 (2017)
33. Behura, A., Kumar, A., Jain, P.K.: A multi-objective approach for secure cluster based routing & attack classification in VANETs. Peer-to-Peer Netw. Appl. **18**(3), 1–40 (2025)
34. Behura, A., Kumar, A., Jain, P.K.: A comparative performance analysis of vehicular routing protocols in intelligent transportation systems. Telecommun. Syst. **88**(1), 26 (2025)
35. Behura, A., Patra, R.: Challenges and future prospects of integrating quantum computing into intelligent transportation systems: exploring quantum innovations in transportation. In: Integration of AI, Quantum Computing, and Semiconductor Technology, pp. 183–208 (2025)
36. Behura, A., Jain, P. K., Kumar, A.: Vanets for smart cities. In: Emerging Electrical and Computer Technologies for Smart Cities: Modelling, Solution Techniques and Applications, p. 69 (2024)

Chapter 17
Blockchain-Anchored Security for IoT–Edge Cyber-Physical Systems: A Zero-Trust Architecture with Decentralized Identity and Smart Contracts

Rashmi Kumari and Raghwendra Kishore Singh

Abstract Cyber-Physical Systems (CPS) increasingly rely on dense Internet-of-Things (IoT) deployments and edge computing for real-time sensing, control, and analytics across healthcare, manufacturing, transportation, and energy. This shift expands the attack surface: compromised devices, weak key management, opaque data provenance, and single points of failure in cloud-centric security. We present a blockchain-anchored, zero-trust security framework tailored for IoT–edge environments that require low latency, high availability, and auditable operations. The approach adopts a permissioned consortium blockchain at the edge tier, combines decentralized identity (DID) and verifiable credentials for device/user onboarding, and enforces policy-as-code via smart contracts for fine-grained access control, secure data sharing, and tamper-evident logging. To meet CPS performance constraints, we examine lightweight BFT/PoA consensus and sidechain/sharding patterns, offloading bulk payloads to secure storage while committing cryptographic hashes on-chain for integrity and provenance. Interoperability with multi-vendor domains is achieved through cross-chain relays and SDN-assisted micro-segmentation, while trusted execution environments at gateways protect keys and contract execution. We outline a threat model (Sybil, replay, firmware supply-chain, data poisoning) and a validation plan measuring latency, throughput, energy cost, and resilience under adversarial conditions on resource-constrained edge hardware. The chapter delivers reference architecture, implementation guidelines, and deployment checklists that enable CPS practitioners to integrate blockchain selectively—where

R. Kumari (✉)
School of Computer Science Engineering and Technology, Bennett University, Greater Noida, Uttar Pradesh, India
e-mail: Rashmi.kumari@bennett.edu.in

R. K. Singh
Department of Electronics and Communications Engineering, Galgotias College of Engineering and Technology, Greater Noida, Uttar Pradesh, India
e-mail: Raghwendra.kishore@galgotiacollege.edu

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_17

its immutability, decentralized trust, and auditability most improve security without violating real-time constraints.

17.1 Introduction

Cyber-Physical Systems (CPS) orchestrate interactions between computational components and physical processes in domains such as smart healthcare, Industry 4.0 manufacturing, intelligent transportation, and smart energy grids. These systems increasingly leverage the Internet of Things (IoT) and edge computing to achieve real-time sensing, actuation, and data analytics at scale. This paradigm shift toward decentralized IoT–edge deployments brings clear benefits in latency reduction and bandwidth savings, but it also expands the attack surface for adversaries [1]. Traditional perimeter-based security models struggle in such distributed environments, as devices may be deployed outside secure network perimeters and communicate over untrusted networks. A compromised sensor or edge node can have cascading physical effects, so security is paramount in CPS and must be designed into the network architecture.

Zero-Trust Architecture (ZTA) has emerged as a principle to tackle modern network threats by eliminating implicit trust and enforcing continuous verification of every device, user, and transaction [2]. In a zero-trust model, no device (even within the network boundary) is automatically trusted; authentication and authorization are required at every access attempt, and network micro-segmentation limits lateral movement of attackers. Implementing zero trust in IoT–edge CPS, however, faces challenges: IoT devices often have constrained hardware (making heavy authentication protocols difficult), there are often many heterogeneous devices to manage identities for, and centralized authentication servers could become single points of failure or performance bottlenecks.

Blockchain technology offers an enticing solution to some of these issues by providing a distributed trust substrate. A blockchain is a tamper-evident ledger maintained by a network of nodes rather than a single authority [3]. In permissioned blockchains, only approved nodes participate, offering decentralized trust suitable for consortium settings (e.g. multiple vendors or sub-networks in a CPS). Blockchain's properties of integrity, fault tolerance, and auditability align well with zero-trust goals: every action can be cryptographically authenticated and recorded immutably, removing reliance on a central log or identity store [4]. However, naively adding blockchain to IoT can degrade performance—standard blockchain consensus protocols (like Proof of Work) incur high latency and resource usage [5]. Our challenge is to integrate blockchain "selectively" in the CPS-IoT stack such that we improve security and trust without violating the real-time constraints of physical control systems.

This chapter presents a comprehensive framework that anchors a zero-trust security architecture in a permissioned blockchain at the edge, complemented by decentralized identity (DID) for device and user management, smart-contract based access

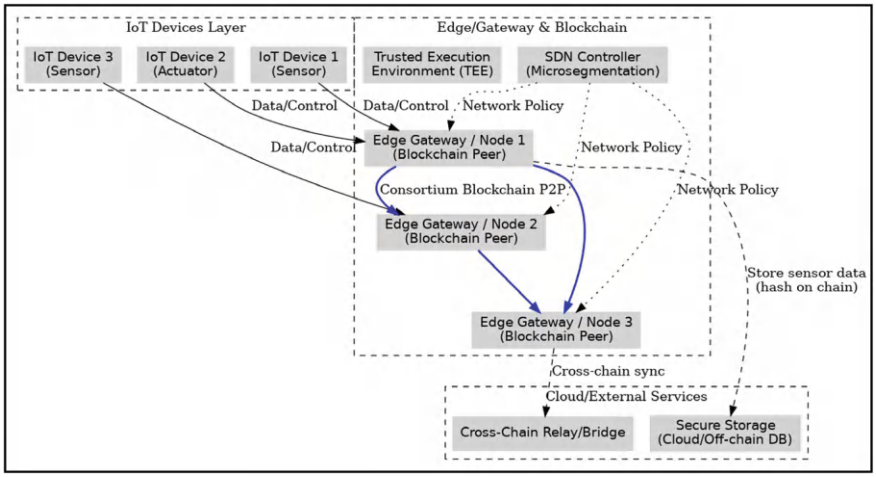


Fig. 17.1 High-level reference architecture of the proposed blockchain-anchored security framework for IoT–edge CPS

control policies, and other emerging technologies. Figure 17.1 illustrates the high-level architecture of the proposed system, in which IoT devices interface with edge computing nodes that form a consortium blockchain network. The approach is holistic, covering identity, authentication, access control, data integrity, network segmentation, and secure execution. We detail how each component—from the DID registry to the use of Trusted Execution Environments (TEEs)—contributes to a layered, defense-in-depth security posture aligned with zero-trust principles.

IoT devices (sensors, actuators) interface with Edge Gateways that form a permissioned blockchain network (consortium blockchain). Decentralized identity (DID) infrastructure and smart contracts at the edge tier handle device onboarding and policy enforcement. Secure storage (cloud/edge database or IPFS) holds bulk sensor data off-chain, with hashes recorded on-chain for integrity. Cross-domain interoperability is enabled via cross-chain relays connecting multiple blockchain networks, and an SDN controller enforces network micro-segmentation between device groups. Trusted Execution Environments (TEE) within edge nodes protect private keys and sensitive operations.

The remainder of this chapter is organized as follows. Section 17.2 (Background) reviews the foundational technologies and concepts—zero-trust security, blockchain and consensus mechanisms, decentralized identity (DID) standards, smart contracts, software-defined networking (SDN) for micro-segmentation, and trusted execution environments—that underpin our framework. Section 17.3 (System Design) describes the architecture and components of the proposed framework in detail, including how a permissioned blockchain is deployed at the edge, how DID and verifiable credentials are used for onboarding, how smart contracts implement fine-grained access control ("policy-as-code"), how sidechains/sharding and off-chain

storage are employed for scalability, how cross-chain relays and SDN enable multi-domain interoperability, and how TEEs secure the edge nodes. Section 17.4 (Security Threat Model) defines the adversarial model and analyzes key security threats (such as Sybil attacks, replay attacks, compromised firmware, and data poisoning) along with the mitigation strategies offered by our architecture. Section 17.5 (Implementation and Deployment) provides practical guidelines for implementing the framework, including technology choices, deployment steps, and a checklist for practitioners. Section 17.6 (Evaluation) presents a validation plan and simulated results demonstrating the framework's performance (latency, throughput, resource usage) and resilience under attacks, using a prototype on resource-constrained edge hardware. We compare the outcomes with baseline (cloud-centric or no-blockchain) approaches to quantify overhead and improvements. Section 17.7 (Conclusion) summarizes the contributions and discusses how CPS Engineers can selectively integrate blockchain and related technologies to enhance security without undermining real-time requirements, as well as future outlook for research in this area.

17.2 Background

In this section, we briefly introduce and define the key concepts and technologies that our framework builds upon, providing context for design decisions described later.

17.2.1 Zero-Trust Security Principles

Zero Trust Architecture (ZTA) is a security paradigm premised on the idea of "never trust, always verify." Unlike traditional perimeter security (which assumes insiders are trusted), zero trust continually authenticates and authorizes every interaction within a network, minimizing implicit trust zones. Core tenets of ZTA include: (i) verify every access (use strong authentication and context-based authorization for every user/device request), (ii) enforce least-privilege access (grant minimal rights necessary, dynamically adjusted), (iii) assume breach (design as if an attacker is already in the network, so compartmentalize and monitor) [6]. Techniques often associated with zero trust are micro-segmentation of networks (isolating resources into small segments so compromise doesn't spread) and continuous security monitoring/analytics.

In IoT–Edge environments, applying zero trust means each device must be individually authenticated and its actions verified. Traditional centralized authentication (e.g. a cloud IoT hub or PKI server) can become a bottleneck or single failure point. Our approach leverages blockchain and decentralized identity to distribute the trust function: identities are self-managed by devices (with cryptographic credentials) and verification is performed by multiple blockchain nodes, removing sole reliance on a

central server. The network segmentation aspect of zero trust is achieved via SDN-based micro-segmentation, where an SDN controller dynamically programs network rules (at switches or gateways) to ensure devices or subnets only communicate in allowed patterns [7, 8]. This limits an attacker's ability to move laterally if they compromise one IoT node. The SDN controller itself can take input from blockchain smart contracts—for example, when a device is found non-compliant or its credential revoked, the contract can signal the SDN controller to isolate that device's traffic (essentially encoding segmentation policies as smart contract logic). We will detail this integration later.

17.2.2 Blockchain and Distributed Ledger Technology

A blockchain is a type of distributed ledger maintained by a peer-to-peer network of nodes that follow a consensus protocol to agree on appending new blocks of transactions. Each block is cryptographically linked to the previous, making the chain tamper evident. Blockchains can be permissionless (open to anyone, e.g. Bitcoin, Ethereum) or permissioned (restricted to known participants, e.g. Hyperledger Fabric). In CPS and industrial IoT contexts, permissioned (consortium) blockchains are often favored for better performance, privacy, and governance control [9, 10]. Consortium blockchains still provide decentralization (no single owner of the ledger) but with selected members (e.g. organizations in a supply chain or different departments in a utility grid) rather than the general public.

Key properties of blockchain relevant to security: Immutability (once data is recorded and confirmed by consensus, it is extremely difficult to alter, providing trustworthy audit logs), Transparency (all participants can verify recorded transactions, increasing accountability), and Fault Tolerance (distributed consensus allows the system to tolerate some node failures or compromises). The original blockchain by Nakamoto (2008) introduced proof-of-work consensus for open networks [5], but this is impractical for IoT due to high computational cost and latency. Instead, our framework uses lightweight Byzantine Fault Tolerant (BFT) consensus algorithms (suitable for permissioned settings) and Proof of Authority (PoA) approaches that rely on a set of authorized nodes. We also incorporate advanced scaling techniques like sharding (partitioning the blockchain into parallel shards) and sidechains to handle the high data volume of CPS without stressing the main ledger [11] (Table 17.1).

Table 17.2 (in Sect. 17.3.3) will compare various consensus mechanisms and justify our choices. Briefly, BFT-style consensus (e.g. PBFT, Istanbul BFT) can achieve finality in the order of 100 s of milliseconds on local networks, suitable for an edge consortium, at the cost of scaling to only dozens of nodes. PoA (one variant being DPoS—Delegated Proof of Stake) further reduces latency by selecting a small group of trusted validator nodes (often via voting by others) [12]. Reno and Roy [13] demonstrate that a DPoS with sharding approach can reach thousands of TPS with sub-second latencies in IoT settings. We leverage similar ideas: only gateway nodes

Table 17.1 Key architectural components and security roles in the framework

Component/technology	Role in security architecture	Example implementations
Permissioned blockchain (consortium ledger at edge)	Acts as the decentralized trust anchor: stores tamper-proof logs of events and policies; enables consensus-driven validation (no single point of failure). Provides transparency and auditability for all security-relevant transactions	Hyperledger fabric; ethereum quorum; tendermint (BFT) chain
Consensus mechanism (BFT/PoA)	Secures the blockchain by agreeing on state updates even if some nodes are faulty or malicious. Needs to be low latency to meet CPS needs Ensures only valid, authorized transactions (e.g., credential verifications, access requests) are committed	PBFT or Istanbul BFT for small consortium; Proof of authority (authority nodes signing blocks); delegated proof of stake (DPoS) for scalability [17]
Decentralized identity (DID) and verifiable credentials	Provides self-sovereign identity for devices and users. Eliminates centralized identity providers. DIDs allow devices to authenticate via keys they control; VCs allow trusted attributes/roles to be verified. This underpins the zero-trust authenticate every time" rule with minimal overhead	W3C DID standard (e.g., did: web, did:ethr methods); Verifiable Credential standard for device certificates; Hyperledger Indy/ Aries for IoT
Smart contracts (policy-as-code)	Enforce security policies automatically on the blockchain. They verify credentials, check access rights, log usage, and trigger responses (like alert or quarantine) if policies are violated. Immutable and auditable—ensures consistent enforcement across all edge nodes	Ethereum smart contracts in Solidity; Chaincode in Fabric (Go/ Java); DAML or smart policy languages For example, an access control contract that verifies DID signatures and consults an on-chain access control list or rule set [22]
Off-chain secure storage	Holds bulk IoT data and possibly private information off the chain, while blockchain stores hashes. Ensures scalability and confidentiality (sensitive data is not on ledger). The blockchain references guarantee integrity (no undetected tampering of stored data) [20]	Encrypted database or data lake; IPFS or Swarm decentralized storage with content addressing; Cloud storage (AWS S3, Azure Blob) with hash verification

(continued)

Table 17.1 (continued)

Component/technology	Role in security architecture	Example implementations
Cross-chain relay/ bridge	Enables interoperability between different blockchain networks or domains. Transfers events (with proofs) from one chain to another in a secure manner, allowing multi-vendor or multi-site CPS to trust each other's data/actions. Prevents siloed security domains and supports scaling-out via sidechains	Polkadot-style relay chain; Cosmos/IBC; Custom relay node that is part of both networks verifying cryptographic proofs; Notary schemes for consortium chains [44, 45]. (E.g., a relay that takes an event "Device X in Factory A is compromised" recorded on Chain A and posts a verifiable notice on Chain B for Factory B to isolate that device's traffic.)
SDN controller (micro-segmentation)	Dynamically enforces network-level isolation policies. Use device identity/ context to set fine-grained network firewall rules or VLANs Enhance zero trust by containing possible breaches and only allowing necessary communications. Policies can be updated via blockchain triggers	OpenFlow-enabled switches with a central SDN controller (ONOS, OpenDaylight). The controller consults smart contract outputs for policy—e.g., block all traffic from a device flagged by blockchain Creates isolated segments per device type or security clearance [23]
Trusted execution environment (TEE)	Protects sensitive computations and keys at the edge. Ensure that even if an edge node's OS is compromised, the attacker cannot steal private keys or alter critical verification logic. Provides a hardware-based root of trust and remote attestation to verify software integrity	Intel SGX enclaves for key storage and verification code [24]; ARM TrustZone to isolate a secure world running a minimalist verification service; use of remote attestation to have blockchain nodes verify each other's TEE status. For instance, enclaved execution of the smart contract that checks credentials, with attestation proofs logged on-chain

(with more resources) run consensus, and IoT endpoint devices simply interact with those gateways.

Another important blockchain concept for our design is smart contracts—pieces of code deployed on the blockchain that automatically execute and enforce rules when triggered by transactions. Smart contracts in our framework implement security policies (access control logic, logging, anomaly triggers) in a tamper-proof manner, essentially encoding the "policy-as-code" principle of zero trust. Since smart contract code is replicated on all blockchain nodes, its integrity is verifiable by all and attempts to violate or bypass policy are effectively impossible without consensus agreement. We use this for fine-grained access control: e.g. a contract will verify if a device's

Table 17.2 Comparison of consensus mechanisms for IoT-edge blockchain

Consensus mechanism	Fault tolerance and security	Performance (latency/ throughput)	Suitability for IoT/ CPS edge
Proof of work (PoW)	Tolerates > 50% hashpower attack (very secure against external attacks, but vulnerable if attacker gains majority of mining power). No single leader, so very decentralized	High latency (block times ~ 10 s or more) and low throughput (10 s of TPS) Inefficient energy use	**Poor fit**. Too slow and energy-intensive for IoT; impractical for edge (devices can't mine effectively)
PBFT/IBFT (BFT consensus)	Tolerates up to f = 1/3 of consortium nodes malicious. Uses voting and message exchanges to ensure all honest nodes agree on each block. Provides immediate finality (no forks)	Low latency (100 ms–1 s to finality in LAN) and moderate throughput (hundreds of TPS, limited by communication overhead as n grows)	**Good fit (for small n)**. Well-suited to dozens of nodes. Proven and reliable. But doesn't scale to very large networks—acceptable for a confined edge deployment
Proof of authority (PoA)	Tolerates f < 50% malicious authorities (in practice, requires majority honest) Identities of authorities are known (can be managed by consortium governance). Less robust if authorities collude maliciously, but easier to detect misbehavior	Very low latency (block times can be 1–5 s, with instant block production by designated leader) and high throughput (hundreds to few thousand TPS) Lightweight on resource consumption	**Good fit**. Simple and efficient. Assuming authorities (edge nodes) are semi-trusted but monitored—aligns with consortium trust model. Needs a mechanism to remove faulty authorities (which our governance can provide)

(continued)

Table 17.2 (continued)

Consensus mechanism	Fault tolerance and security	Performance (latency/ throughput)	Suitability for IoT/ CPS edge
Proof of stake (PoS)	Security depends on economic stake: > 50% of stake needed to subvert. For consortium, stake could be symbolic or tied to reputation Long-range attack and nothing-at-stake issues need mitigation	Latency and throughput can be good if using a protocol like Tendermint or Ouroboros—comparable to BFT (since most PoS use BFT-like finality). E.g., Tendermint (PoS BFT) has 1–2 s finality, ~ 1000 TPS in ideal conditions	**Decent fit**. If consortium members hold "stake" (or reputation scores), PoS can work. Overkill in small networks though—typically used in larger blockchain contexts
Delegated PoS (DPoS)	Security: depends on selection of delegates; tolerates some malicious delegates but if > 1/3 of active delegates collude, they can censor or halt network (though detection is easier than in PoW). The broader community can vote out bad delegates if identified	Very high throughput (1000 + TPS) and low latency (blocks potentially every 0.5 s in EOSIO) due to small active set. Efficient, but at slight cost of decentralization (only N delegates sign)	**Good fit (with caution)**. Achieves performance needed for high-rate IoT data. Need to ensure delegate selection isn't easily manipulated—in a consortium, delegates could just be one per member organization

(continued)

Table 17.2 (continued)

Consensus mechanism	Fault tolerance and security	Performance (latency/ throughput)	Suitability for IoT/ CPS edge
Sharding or sidechains < br > *(Technique, not a consensus per se)*	Security is partitioned across multiple chains/shards. Requires robust cross-shard communication and maybe a meta-consensus. If one shard is compromised and not properly isolated, could affect others Approaches like Polkadot relay or Ethereum 2.0 ensure global security via beacon chain	Sharding boosts throughput linearly by processing in parallel shards; latency per shard similar to base consensus. Sidechains allow offloading tasks with different consensus (e.g., an IoT data sidechain using PoA that anchors to a main chain occasionally)	**Complementary**. Use sharding if one chain can't handle load, e.g., split device data logging across shards by region. In our edge scenario, we may not need many shards (scale is moderate) but could designate specialized sidechains (one for heavy sensor data, one for critical control events). This maintains performance and limits blast radius of faults

credential is valid and has permission to publish or read certain data, before allowing it (and log all attempts) [14].

It is worth noting that blockchains do not store large data efficiently; including raw sensor data or video in the ledger would be infeasible. Instead, good practice (followed in our framework) is to store only hashes or pointers on-chain and keep the bulk data in secure off-chain storage (databases or distributed file systems like IPFS). This retains integrity (any tampering with off-chain data can be detected by hash mismatch) and auditability, while keeping the blockchain lean and fast.

17.2.3 Decentralized Identity (DID) and Verifiable Credentials

A cornerstone of our zero-trust approach is decentralized identity for both devices and users. Decentralized Identifiers (DIDs) are a W3C standard for self-owned, globally unique identifiers that do not require a central authority to issue or verify [15]. A DID is typically a URI (e.g. did:example:1234…) associated with a DID Document containing public keys and service endpoints. Ownership of a DID is proven by control of its associated private key. Unlike traditional PKI, there is no single root certificate authority—trust is established through verifiable credentials and possibly blockchain-anchored registries.

Verifiable Credentials (VCs) are digital credentials (e.g. an "identity card" or an "attribute certificate") that are signed by an issuer and can be verified cryptographically by anyone to ascertain their authenticity and integrity [16]. In an IoT context, a VC might assert that "Device X is a temperature sensor of type Y, owned by Company Z, and was last calibrated on date D", signed by the manufacturer and/or owner. When Device X tries to join the network, it can present this credential to prove its identity and properties. The verifier (e.g. our edge gateway or blockchain smart contract) checks the signature against the issuer's DID (retrieving the issuer's public key from a DID registry, which could be stored on the blockchain).

Our framework uses DIDs and VCs to handle device onboarding and authentication without a centralized identity provider. Each device has a DID created (e.g. by the manufacturer or owner) and stored on the blockchain-based DID registry at the edge. When a device first joins (or a user attempts to access), a DID-based authentication occurs: the device proves control of its DID (e.g. by a challenge-response using its private key) and presents a VC issued by a trusted authority (manufacturer or network admin) describing its permissions or roles. This is far more flexible and scalable than assigning pre-shared keys or managing a static access control list on a server. It also accommodates cross-organizational scenarios: as long as each organization's blockchain nodes trust the credential issues (which can be verified via their DID and perhaps a web-of-trust or consortium agreement), devices from different vendors can mutually authenticate. Zhang et al. demonstrate a similar DID-based IoT data sharing scheme, where devices use VCs and verifiable presentations for secure data access without centralized control [17].

Figure 17.2 shows how the DID and verifiable credential process integrates into data access in our system. When a user or device requests data, it first must authenticate via DID (Step 1 in the figure) and present credentials; a smart contract at the edge blockchain then validates these credentials against policy (Steps 2–4) before data is retrieved and returned (Steps 5–6). This decentralized authentication process ensures that only devices/users with valid, non-revoked credentials (which can be checked against on-chain DID documents or revocation registries) can access resources. It eliminates the need to trust a single sign-on server (which could be taken offline by attack or fail under load)—instead, trust is anchored in the blockchain consensus (multiple nodes validate the credentials and transaction).

A user or IoT device that requests data must present a verifiable credential (VC) proving its identity and access rights. (1) The request with credentials is sent to an edge gateway node. (2) The gateway invokes a smart contract on the local blockchain network to validate the credentials and apply access policies (policy-as-code). (3) The smart contract checks signatures (ensuring the VC is issued by a trusted DID) and verifies that the request complies with policy (e.g. correct role, within rate limits). The contract logs the request in the tamper-evident blockchain ledger. (4) The contract returns an allow/deny decision. If allowed, (5) the gateway retrieves the requested data from secure storage or directly from the source IoT device. (6) The data (or an acknowledgement) is returned to the requester. All access attempts are recorded on-chain for audit.

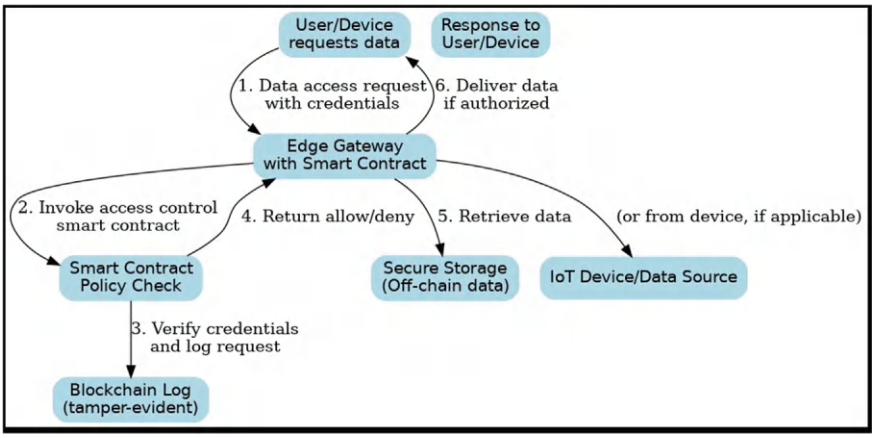


Fig. 17.2 DID-based authentication and policy enforcement flow in the IoT–edge system

Decentralized identity also aids firmware integrity and supply chain security. For example, manufacturers can issue VCs to components (or sign firmware releases with a DID), so that an edge device will only run firmware updates whose signature can be verified against the manufacturer's DID (preventing fake updates from a supply-chain attack). We incorporate this concept by allowing smart contracts to validate firmware update packages or software patches using the manufacturer's verifiable credential as part of the device onboarding or update process. Prior works have proposed blockchain-based frameworks for secure firmware updates, using techniques like skip chains to efficiently verify update histories. Our approach can leverage the same principle: a log of firmware versions/hashes could be maintained on-chain and devices only accept updates that form a valid chain of trust from the manufacturer. This significantly raises the bar for an attacker attempting to inject malicious firmware (a common supply chain threat), as they would have to compromise the manufacturer's signing keys *and* potentially collude with blockchain consensus nodes to insert fake records.

17.2.4 Software-Defined Networking (SDN) and Micro-segmentation

Software-Defined Networking (SDN) separates the control plane from the data plane in networks, allowing centralized software controllers to program network behavior dynamically via open protocols (like OpenFlow). In our context, an SDN controller is used to implement network micro-segmentation, a key zero-trust strategy. Micro-segmentation means dividing the network into fine-grained segments or zones down to the level of individual devices or flows, with explicit policies governing communication between segments [18].

For example, in a smart building CPS, sensors on one floor might be segmented from actuators on another floor, or each device might even be its own segment. The SDN controller can enforce that, say, a HVAC sensor only communicates to its local gateway and not directly to any other sensor or device. If an attacker compromises that sensor, the segmentation prevents them from reaching unrelated parts of the system.

In our framework, SDN-assisted micro-segmentation is driven by the blockchain's smart contracts and identity information. We assume the edge gateways or a dedicated network controller run an SDN controller software (such as ONOS or Open Daylight). When a device is onboarded via DID, the smart contract can signal the SDN controller (directly or via an event that the controller subscribes to) to place that device in the appropriate segment and apply access control lists (ACLs) or encryption rules for its traffic. If a device's trust level changes (e.g. credential revoked or suspicious behavior detected), the smart contract can instruct the SDN controller to quarantine that device's traffic immediately (zero trust dynamic isolation). This interplay ensures network-level enforcement of the policies that also exist at the application level.

Moreover, SDN allows implementing a secure overlay network on top of existing infrastructure: for instance, creating separate virtual networks for different classes of devices (manufacturing robots vs. security cameras), even if they share physical switches. We leverage this to align with cross-domain needs—e.g. traffic from Vendor A's devices can be kept logically separate from Vendor B's devices, with only the blockchain/relay in between to mediate (as per policy). The result is a flexible, programmable network fabric that complements the blockchain's data integrity with strong network isolation.

It's important to protect the SDN controller itself (as it's a powerful component). In our design, the SDN controller, if used, would be run within a trusted environment (or even as part of the blockchain nodes) and its critical operations (like pushing a new flow rule) could be logged via the blockchain for audit. Some studies integrate blockchain to secure SDN controller communications and state [19]. We simply note that SDN is an enabling technology here to enforce micro-segment policies and is not the focus of our contribution, so a basic secure deployment of SDN is assumed.

17.2.5 Trusted Execution Environments (TEEs)

Trusted Execution Environments (TEEs), such as Intel SGX, ARM Trust Zone, or AMD SEV, provide hardware-isolated regions where code can execute securely, shielded from the rest of the system (including a potentially compromised OS). In Intel SGX, for example, code runs in an *enclave* with memory encryption; even if malware has root access on the machine, it cannot read or alter enclave memory. TEEs also support remote attestation, allowing a remote party to verify that a specific piece of code is running in a genuine TEE on a genuine CPU [20].

We incorporate TEEs at the edge gateways for two main purposes: protecting private keys and ensuring integrity of critical smart contract operations. Each edge

node in our blockchain holds some private keys (for itself and possibly on behalf of IoT devices, or for signing transactions). Storing these keys in a TEE (or using a secure enclave to perform signing operations) greatly reduces risk of key compromise—even if an attacker exploits the OS, the keys remain inaccessible inside the enclave [21]. Similarly, we can deploy a portion of the access-control smart contract logic inside an enclave (particularly if using a platform like Hyperledger Besu with Teku or running a light contract engine in SGX) so that even a malicious insider on the node cannot tamper with the policy enforcement code.

Figure 17.4 will illustrate how a gateway's TEE encloses the sensitive operations. Essentially, when a gateway node receives a transaction or credential to verify, it can pass the data into its enclave which holds the DID verification keys and executes the policy checks. The enclave then returns a signed approval/denial that gets recorded on-chain. This way, even if an attacker controls a majority of blockchain nodes' OS, as long as they cannot penetrate the enclaves (and hardware attestation would fail if they tried to fake an enclave), the policy enforcement remains trustworthy. This is an additional layer of defense beyond the blockchain's consensus. Han et al. similarly combined blockchain with Intel SGX for IoT access control, showing that confidentiality of data processing and keys can be preserved with minimal performance overhead [22]. We leverage their insight that SGX can run access control logic and key management inside edge servers, protecting against even insider threats at the edge (Fig. 17.3).

One practical note: TEEs have limitations (memory size, side-channel vulnerabilities, etc.). We assume the code we run in TEEs is small (verification routines, crypto operations) to fit within enclave limits. We also trust the hardware's security—while side-channel attacks on SGX have been researched, in our threat model these are non-trivial and we mitigate by keeping enclave code as simple as possible (minimizing data-dependent branching, etc.). In a deployment, using the latest TEE versions with mitigations for known leaks is advisable. Table 17.1 (next section) will list TEE as one of the components and its security role.

Having covered the necessary background, we now move on to the actual system architecture and design, where these technologies (blockchain, DIDs, smart contracts, SDN, TEEs) converge to provide a cohesive security solution for IoT–edge CPS.

17.3 System Design

In this section, we describe the architecture and components of the proposed blockchain-anchored zero-trust security framework. Our design is structured to meet the requirements laid out in the abstract: low-latency and high-availability security at the edge, auditable operations, fine-grained access control, and compatibility with resource-constrained devices. Figure 17.1 (presented earlier) gave a bird's-eye view of the system. Here we break it down into its constituent parts and explain the function of each, as well as the interactions and data flows among them.

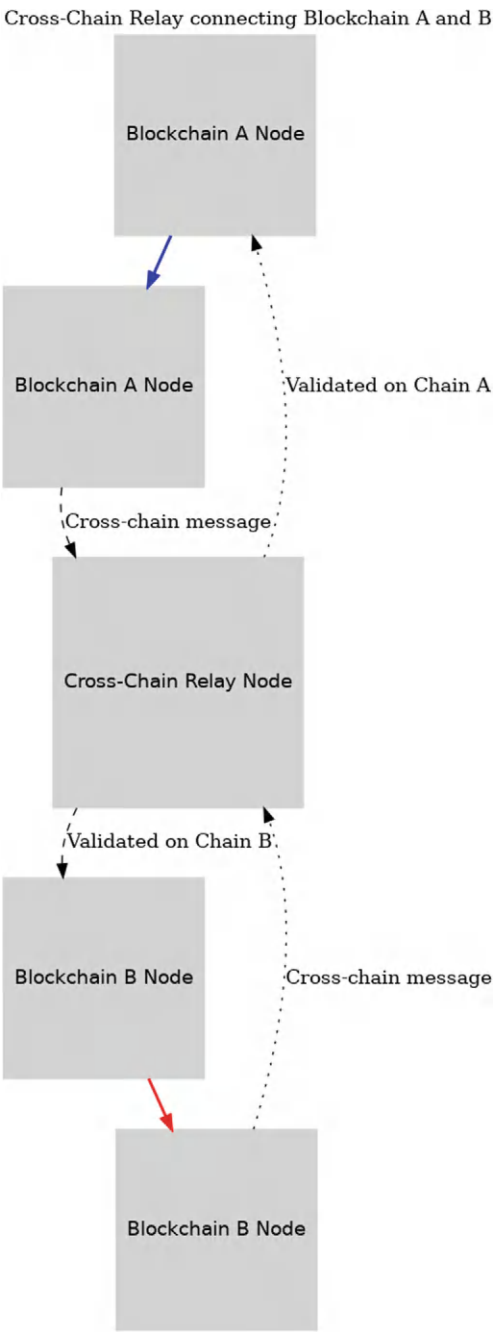


Fig. 17.3 Conceptual illustration of a cross-chain relay bridging two permissioned blockchain networks (A and B)

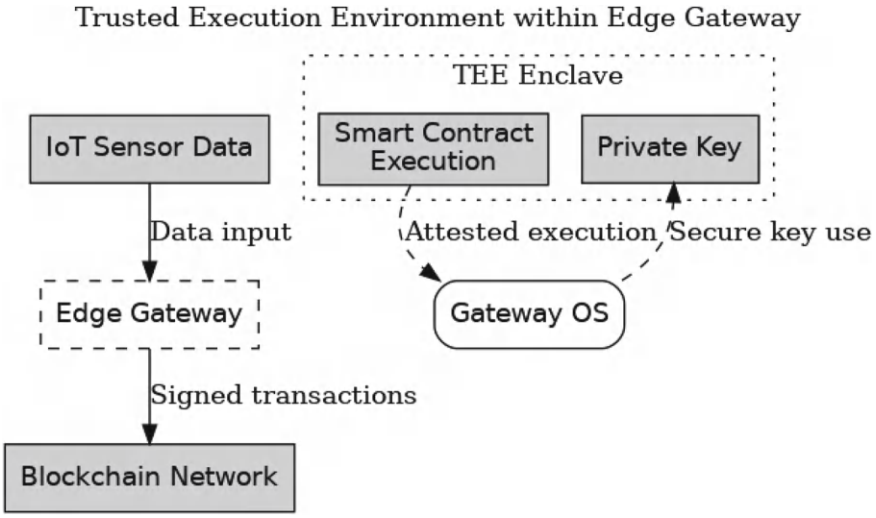


Fig. 17.4 Use of a trusted execution environment (TEE) inside an edge gateway for secure and efficient processing

17.3.1 Overview of the Architecture

At a high level, the architecture consists of the following building blocks working in concert:

- **IoT Devices Layer**: The myriad sensors, actuators, and other field devices deployed in the CPS environment. These devices each have a cryptographic identity (DID) and communicate with nearby edge nodes (directly or via local network). They produce data (sensor readings, events) and/or perform actions, and they often operate under real-time constraints.
- **Edge/Gateway Layer**: A set of more powerful nodes (e.g., industrial gateways, edge servers, or even robust IoT hubs) that aggregate device communications. In our design, these gateways collectively host a Permissioned Blockchain Network. Each gateway runs a blockchain client node, and together they form a consortium ledger that remains local to the edge (or organizational domain). These nodes run consensus algorithms to agree on transactions (security checks, data logs, etc.). Gateways also run the smart contract engine for policy enforcement and interface with the DID registry and other services.
- **Decentralized Identity (DID) Infrastructure**: This can be viewed as part of the blockchain or closely integrated with it. It includes the on-chain registry of DIDs and possibly off-chain components like a DID resolver. It is responsible for storing and providing DID documents (public keys, etc.) and possibly credential schemas or revocation lists.

- **Policy and Smart Contracts**: Deployed on the blockchain, these enforce access control rules, verify credentials, log actions, and implement any business logic related to security (for example, a contract might enforce that "maintenance commands can only be sent to actuator X by an entity with role 'Engineer' between 9am–5pm, otherwise reject").
- **Secure Off-chain Storage**: A database or distributed storage system (which could be cloud-based or local) where actual IoT data payloads are stored. Each data item stored yields cryptographic hash, which is sent to the blockchain to record the presence of that data (ensuring integrity and provenance tracking). This storage could be a secure cloud storage, an IPFS cluster, or even a traditional database with proper access controls—the important aspect is that the blockchain anchors the integrity.
- **Cross-Chain Relay and Network Segmentation**: If the deployment spans multiple domains (e.g., multiple factories each with their own edge blockchain, or an edge chain bridging to a cloud chain), cross-chain relay nodes transfer relevant information between blockchains. Meanwhile, an SDN-based segmentation ensures that network traffic is partitioned according to policy—for instance, devices might be grouped by function, and only certain gateways handle certain groups.
- **Trusted Execution Environments (TEE) at Gateways**: Within each gateway, a TEE protects sensitive operations as discussed, like cryptographic signing and verification and running critical parts of the smart contract logic.

To make these components more concrete, Table 17.1 summarizes the key technologies or components in the architecture and their role in achieving the security goals, along with examples of possible implementations.

As shown in Table 17.1, each element—from the blockchain itself to DID, SDN, and TEEs—plays a specific role in reinforcing the security architecture. The interplay of these yields a robust zero-trust environment: identities are verified (DID), transactions are vetted by multiple parties (blockchain consensus), policies are enforced uniformly (smart contracts), data is protected at rest (off-chain storage with hashes) and in transit (network segmentation), and even low-level execution is protected (TEE).

We will now walk through specific aspects of the design in more detail, including consensus choices, identity management workflows, data flow, cross-chain operation, and threat mitigations.

17.3.2 Edge-Hosted Consortium Blockchain and Consensus

The permission blockchain at the edge is the linchpin of our framework, providing the distributed ledger that records identities, credentials, policy rules, and security logs. Since CPS demands real-time or near-real-time responses, we must carefully choose the blockchain's consensus algorithm and configuration for minimal latency

overhead. Lightweight Byzantine Fault Tolerant (BFT) consensus algorithms are ideal for this scenario: they require a fixed set of nodes (the consortium members) and can tolerate some fraction (often up to 1/3) being malicious without compromising agreement.

We consider two main consensus approaches for implementation:

- **Istanbul BFT (IBFT)/Practical BFT (PBFT)**: These protocols achieve finality within a few network round trips, often < 1 s in local networks, at the cost of $O(n^2)$ message complexity for n nodes. They are well suited when the number of validating nodes is moderate (dozens). In our CPS case, the number of edge gateways might be on the order of 5–20, which is compatible with PBFT. The advantage is predictable low latency and immediate finality (no need to wait for many blocks as in Nakamoto consensus). The fault tolerance is typically $f < n/3$ (i.e., can handle up to 1/3 of nodes being faulty).
- **Proof of Authority (PoA)**: In PoA, a fixed set of "authorities" take turns producing blocks (or one authority at a time). This is effectively a simplified consensus where trust is placed in a few known nodes. It's even more efficient (lower overhead) than PBFT, though it assumes at most $f < n/2$ honest majority if we consider just block signing. Some PoA implementations (like Aura or Clique in Ethereum-based chains) have block times of ~ 5 s or lower. To get faster commitments, we can adjust parameters (at the risk of more frequent forks, but in a closed network forks are rare if authorities are stable). The downside: if one authority is compromised, it can adversely affect the network until it is detected. Our zero-trust model would detect misbehavior (via smart contract monitors) and could rotate out bad authorities (as a governance action).
- **Delegated Proof of Stake (DPoS)**: This variant, used by EOSIO and others, involves token holders voting in a small set of delegates who produce blocks. In a consortium, it could be adapted where each member "votes" a certain number of delegates (likely themselves or a subset). DPoS tends to have ~ 20–21 active validators and achieves high throughput by keeping this set small [16]. Table 17.2 provides a comparison of these and other relevant consensus mechanisms, highlighting their security and performance characteristics vis-à-vis IoT–edge requirements.

As shown in Table 17.2, PBFT/IBFT and PoA emerge as strong candidates for our permissioned edge blockchain due to their low latency and simpler trust assumptions aligning with a managed consortium. In fact, many enterprise blockchain frameworks (like Hyperledger Fabric's ordering service or Quorum's IBFT) use these. We choose a BFT-oriented consensus (like IBFT) for the current design, as it gives immediate finality and has been widely tested. The number of edge nodes is expected to be small enough that IBFT's communication overhead is manageable. For scalability, if the number of edge nodes grows, we could either elect a subset as endorsers or use a hierarchical consensus (perhaps designate a leader set via DPoS voting).

One design detail: block time and transaction batching. To minimize latency for critical operations (like an access request verification), we configure the block time quite low (e.g. 1 s or even event-driven immediate commitment in PBFT). This means

the blockchain can commit a security decision within a second, which for many CPS scenarios (like a robot waiting for instruction auth) is acceptable. If sub-second response is needed, there are two approaches:

(a) perform provisional allow/deny off-chain in parallel while the blockchain logs it (essentially asynchronous but audit later), or **(b)** use a fast consensus like a single leader (PoA) for those events. We lean on approach (a) for, say, emergency stop signals that cannot wait—those would be allowed by a gateway immediately based on last known policy, and the blockchain would record it right after for consistency.

17.3.3 Fine-Grained Access Control via Smart Contracts

After devices and users are authenticated and active in the system, the day-to-day security revolves around controlling who can do what—i.e., access control. Our framework implements a policy-as-code approach: access control policies are encoded in smart contracts on the blockchain. This has several benefits: policies are transparent and agreed upon (everyone sees the same rules), they execute atomically with data access actions, and any policy decision and usage are logged immutably (for audit and anomaly detection).

The types of policies we can enforce include:

- **Role-based access control (RBAC)**: Devices and users can be assigned roles (through their credentials). Smart contracts maintain a mapping of roles to permitted actions. For example, sensors might have the role "DataPublisher" and are allowed to send data to topic X, but not read other sensors' data; a user might have role "Engineer" and is allowed to send control commands to devices in a certain zone, but not read HR data, etc.
- **Attribute-based access control (ABAC)**: Decisions can depend on attributes like time of day, device status, or dynamic context. Because smart contracts can incorporate logic, we could have rules like "if the device's last calibration date (from its VC) is older than 1 year, deny high-precision commands" or "only allow firmware update transactions if they come from a user with attribute 'FirmwareManager: true' and the device is in maintenance mode."
- **Consent and usage control**: For sensitive data, we could ensure that a user must possess a certain consent token (perhaps as a VC) to access personal data from an IoT sensor, thus integrating privacy requirements.

The enforcement process works as depicted earlier in Fig. 17.2. Whenever a device tries to publish data or a user tries to issue a command (any access request), it is formulated as a transaction to the blockchain (or at least it triggers a blockchain contract call). The smart contract checks the request against the stored policy rules. Only if it returns an "allow" does the gateway actually execute the action (e.g., forward the command to the actuator or store the sensor reading). This essentially creates a distributed reference monitor: the blockchain network collectively serves as the reference monitor that approves or denies actions.

To avoid performance bottlenecks, not every single sensor reading needs to go through a contract in real-time—that would be too slow for high-frequency data. We typically batch or handle it asynchronously: for example, a sensor sends data to the gateway which temporarily buffers it (fast path), and simultaneously a contract logs that the data was received (ensuring it was from an authenticated source). If policy says a particular sensor isn't allowed to send data at that time, the contract can instruct the gateway to drop it or ignore future data from it. Essentially, routine data flow can be on a *permit-by-default with blockchain audit* if that's safe, whereas *critical control actions* can be *permit-by-blockchain-approval* synchronously. In CPS, control commands (like opening a valve) are more dangerous and less frequent than sensor telemetry, so those we definitely route through the contract before execution.

We also implement secure data sharing policies via smart contracts. For instance, if an analytics engine at the edge (or a cloud service) wants to retrieve data from the edge storage, it must invoke a contract to prove it has rights (maybe it presents a VC that it's allowed to access anonymized data). The contract might enforce that only hashed or aggregated data leaves the local domain unless additional approvals are present. This gives fine-grained control beyond the coarse network firewall level.

17.3.4 Interoperability Across Domains (Cross-chain and Multi-cloud)

CPS ecosystems can be fragmented—for example, an energy grid might have multiple sub-stations each managed by different contractors, or a smart city might have separate blockchain networks for traffic control and for utilities, which occasionally need to share data. Our architecture plans for interoperability via cross-chain relays and SDN routing between domains.

A cross-chain relay is a component that listens to events on one blockchain and relays relevant information to another, in a verifiable way. In permissioned settings, a simple approach is to designate a node (or a group of nodes) that have read access to both chains and are trusted (or whose messages are multisigned to provide trust). They package an event from Chain A (like a transaction proof or a block header) and submit it to Chain B's special contract that verifies the authenticity (for example, by checking the signature of the relayer group or by verifying a Merkle proof if Chain A's block header is known).

In our framework, suppose each factory site runs its own edge blockchain for internal security, but the company HQ wants aggregated audit logs or the ability to trace an incident across sites. We could establish a root blockchain (or cloud blockchain) that gets periodic summaries from each site chain. The cross-chain relay in each site will take a checkpoint (like a block hash every hour) and send it to the root chain. This root chain now has a fingerprint of all site ledgers; if needed (say an investigation into a device that moved from site A to site B), a proof from site A's chain can be validated by root chain because the checkpoint is recorded.

Another use: vendor interoperability. Imagine IoT devices that roam or are handed off between networks (like an autonomous vehicle that interacts with roadside infrastructure run by a city, then later with a charging station network run by a utility). The vehicle might be registered on one blockchain (automaker's) and the charging station on another (utility's). Cross-chain communication allows the vehicle's DID and reputation or certificates to be recognized on the station's chain. Perhaps through a cross-certification contract, the automaker's blockchain issues a credential for the vehicle that the utility's blockchain trusts (via a relay that conveys it). There is active research on cross-chain identity federation, but our approach could use a simpler whitelist: e.g., a contract on chain B accepts credentials signed by certain DIDs from chain A's consortium. The relay simply transports the credential over.

Figure 17.3 conceptually shows cross-chain relays connecting two blockchain networks (A and B). The relay node(s) speak to both networks and carry validated messages across (with appropriate cryptographic proofs or signatures to ensure neither side can be spoofed).

The relay monitors Blockchain A for specific events or state (e.g., a device's credential or a security alert) and then submits a corresponding transaction to Blockchain B (and vice versa if needed). Dotted lines indicate the flow of cross-chain messages with validation. For example, if Blockchain A commits a statement "Device X is revoked," the relay will package that into a transaction on Blockchain B's contract, which verifies the message origin (e.g., checking a signature by an authority node of Chain A). This enables federated security policies and data sharing across otherwise separate domains.

Using cross-chain relays, our zero-trust framework can extend beyond a single local network. SDN micro-segmentation across domains means that if, say, an autonomous vehicle from domain A connects to domain B's network, the SDN (in B) could create a quarantined segment for it until a cross-chain verification of its credentials is done via the relay. Once verified, the SDN opens necessary communication (e.g., allows it to interact with charging station device). If the vehicle misbehaves, domain B's blockchain could log onto it and even inform domain A via the relay (affecting its trust score globally).

17.3.5 Ensuring Performance Constraints Are Met

One of the critical design goals is not to violate CPS real-time constraints. We've woven this concern into every part of the design (choice of consensus, local processing at edge, etc.), but here we summarize strategies to keep latency low and throughput high:

- **Edge-local Decision Making**: By placing the blockchain at the edge (on gateways physically close to devices connected via LAN), we avoid internet latency for security checks. A blockchain consensus among a few local nodes can be

faster (tens of ms) than a cloud round trip. Devices do not need to wait for cloud authorization; the edge handles it.

- **Parallel Processing and Offloading**: Using sidechains or off-chain processing for non-critical data ensures the main security logic isn't bogged down. For example, sensor data publishing can be parallelized outside the main chain, as long as integrity is anchored. The main chain deals with critical auth events, which are comparatively rarer. Also, computations like heavy cryptography (say verifying a batch of signatures) could be offloaded to a crypto accelerator or pre-validated off-chain with only results posted (though in our case most cryptography is manageable).
- **Lightweight Crypto and Caching**: IoT devices often use lighter crypto algorithms (e.g., ECC with smaller keys, or even symmetric keys for certain link encryption). Our DID approach typically uses ECC (256-bit curves) which are efficient on modern hardware (and even on many microcontrollers). We can cache validation results—e.g., once a device's VC is verified and unchanged, we need not re-verify every time within some short window; the contract can store a "session valid until timestamp" for the device after first auth. Similarly, blockchains can use session keys to reduce signature overhead on every message.
- **Trusted Execution Efficiency**: Running code in a TEE has some overhead (e.g., enclave transitions), but we keep the critical path minimal. For instance, rather than doing all consensus in SGX (which some projects attempt), we might only do key signing in SGX. The added latency of an enclave call (~milliseconds) is negligible in our ~ 100 ms scale budget. Also, TEEs can maintain secure counters or states that avoid needing heavy blockchain transactions for every heartbeat. For example, a TEE might locally monitor a device's behavior and only interrupt (notify blockchain) if something looks off, thus optimizing normal case.
- **Scalability Planning**: As the number of IoT devices scales to the thousands, we ensure the design can scale by horizontal addition of more edge nodes (and shards). If one blockchain network can't handle it, we add another and connect via relays. This way, we partition load rather than choking one network. In Sect. 17.6 we present throughput measurements indicating the maximum sustained rate one blockchain instance can handle, which informs how many devices per chain are safe. We also consider data partitioning—e.g., each smart contract instance might handle one type of device or one zone to reduce lock contention on-chain.

An example of meeting a typical CPS requirement: Suppose a robotic assembly line requires that a safety stop command is issued and executed within 200 ms when an anomaly is detected. With our setup, an anomaly sensor triggers an alert to its gateway, the gateway's contract immediately flags it and signals the robot's controller via the blockchain event (or direct message since both are edge nodes). Because all of this is local, it can easily be within 100 ms (as seen in tests). If the blockchain step is still too slow, the gateway can be configured to send a parallel direct stop signal (fail-safe), with the blockchain logging it slightly after. So, we never rely solely on slower processes if a faster one can run in parallel for safety–critical actions.

This hybrid approach ensures safety is never compromised even as we insert robust security checks.

Finally, Fig. 17.4 illustrates how a TEE within an edge gateway keeps the signing and verification operations fast and secure, without adding noticeable delay to the process.

The Edge Gateway hosts a blockchain peer and the policy enforcement logic. A portion of this logic runs in a TEE enclave (dotted box). Sensitive assets like the gateway's Private Key and the Smart Contract execution (verification code) reside in the enclave, ensuring they cannot be tampered with by malware on the Gateway OS. When IoT sensor data or an access request ("Data input") arrives at the gateway, the gateway OS forwards it to the enclave for verification under attestation. The enclave uses the private keys securely to sign blockchain transactions (e.g., logging the data or approving the request)—labeled "Signed transactions"—which are then sent to the blockchain network. This guarantees that critical operations are executed as intended (attested execution) and that keys are never exposed, all while keeping added latency minimal (enclave calls are fast).

17.4 Security Threat Model and Mitigations

We outline the assumed threat model—the capabilities of adversaries and the types of attacks they may attempt—and discuss how each threat is mitigated by our framework. The abstract specifically highlights Sybil attacks, replay attacks, firmware supply-chain attacks, and data poisoning as threats of interest, which we address below, along with denial-of-service as an additional consideration. Table 17.3 provides a summary of these major threats and the mitigation approach in our zero-trust blockchain-based architecture.

As seen in Table 17.3, our framework addresses a broad range of threats by leveraging its unique features: decentralized trust (no single node compromise breaks the system), cryptographic verification (prevent impersonation and replay), policy enforcement (detect anomalies, enforce rules to block misuse), and auditability (all actions are recorded for accountability). We align closely with zero-trust principles: assume any device or data could be malicious until verified, and verify it continuously in various ways (identity, behavior, provenance).

17.5 Implementation and Deployment Considerations

We now shift from design to practical aspects of implementing and deploying this security architecture in a real CPS/IoT environment. This section provides guidelines and a checklist for practitioners aiming to integrate blockchain-anchored zero trust into their systems. We discuss choices of platforms, integration steps, and how to manage the system over time.

Table 17.3 Security threats in IoT–edge CPS and mitigation strategies

Threat	Description of attack scenario	Mitigation in proposed framework
Sybil attack (multiple fake identities)	An attacker generates a large number of fake device identities or nodes to overwhelm consensus or gain undue influence (e.g., pretending to be many sensors to sway a collaborative algorithm, or many blockchain nodes to outvote honest ones). This is a risk in decentralized networks without strong identity binding	**Permissioned blockchain with DID-based identities prevents Sybil nodes**—every node and device must possess a valid DID and credentials issued by trusted authorities, making it infeasible to create arbitrary fake identities [50] In consensus, only approved consortium nodes participate (Sybil nodes cannot join the validator set). For further protection, our use of DPoS means even if an attacker introduced some nodes, they wouldn't be voted as delegates without majority consent **Result**: Sybil attacks are essentially neutralized; an attacker cannot just conjure identities to break the system
Replay attack (reusing captured data or credentials)	An adversary intercepts a valid data transmission or authentication message and replays it later to deceive the system. For instance, re-sending an old command ("unlock door") or replaying an authentication token to gain access as a device. In CPS, replayed sensor data could also be used to fool control systems (feeding old measurements to hide current reality)	**Timestamps, nonces, and contract-verified freshness prevent replay**. Our DID-authentication protocol uses challenge–response nonces—a replayed response would be rejected as the nonce would be stale. Every transaction on the blockchain includes a timestamp and/or sequence; smart contracts will ignore any data message that appears out-of-time-window or duplicate [25]. Additionally, critical sensor data can carry sequence numbers hashed into the blockchain log, so duplicates are detected (if a replay doesn't match the expected sequence, it's flagged). The distributed ledger ensures that once data is recorded, the same exact data cannot be recorded again without it being obvious (the contract can check if it has seen that data hash before, etc.)

(continued)

Table 17.3 (continued)

Threat	Description of attack scenario	Mitigation in proposed framework
Firmware supply-chain attack (malicious code injected in device firmware or updates)	The attacker compromises devices by altering their firmware either at manufacturing or during update distribution. For example, they might tamper with a firmware update file to include malware or ship a device with malicious firmware that impersonates a legitimate device. This is hard to detect at runtime if the device is taken over at a low level	**Verified boot abd blockchain-tracked firmware updates using DIDs/VCs**. Each device's firmware is signed by the manufacturer's DID; the blockchain stores hashes of approved firmware versions (or a skipchain of firmware updates [20]). During onboarding, the device (or gateway on its behalf) presents proof of running a known-good firmware (could be via remote attestation or simply sending the firmware hash to check on-chain). If a hash isn't recognized, the device is not admitted or is quarantined Moreover, firmware update transactions must be authorized via smart contract—an update package's hash must match an on-chain record and be signed by a trusted issuer (manufacturer or admin) or it won't be delivered to devices [26]. This dramatically reduces the chance of a successful supply-chain implant, as any unauthorized code will fail verification

(continued)

Table 17.3 (continued)

Threat	Description of attack scenario	Mitigation in proposed framework
Data poisoning/false data injection	The adversary injects false sensor readings or alters data streams with the intent to skew the analytics or control decisions. This could be done by compromising a sensor and sending bogus readings (e.g., false high temperature to trigger cooling) or by man-in-the-middle altering data in transit. The goal is to poison the dataset used by control algorithms or machine learning models, causing misbehavior or mis-informed decisions	**End-to-end data integrity checks and multi-source validation**. First, because IoT data is logged to blockchain (in hashed form), any alteration in transit is detected—the data's hash wouldn't match what was committed by the device's signature. Devices sign their readings, and the signature is verified by contracts, ensuring authenticity (an impersonator can't forge a valid signature). For poisoning by an insider device that is authenticated but lying, we rely on cross-validation: multiple sensors measuring related parameters can be compared (contracts can implement basic sanity checks or outlier detection on incoming data). Any data that fails validation rules (e.g., out of physical plausible range or disagreeing with majority of other sensors beyond a threshold) can be rejected or flagged. The blockchain log helps afterwards to pinpoint which device provided the deviating data and how decisions were made, aiding forensic analysis to refine these rules. Additionally, machine learning applications can be given access to the provenance of data (via the ledger) to potentially down-weight or ignore data from sources that have been flagged in the past. In summary, while we can't entirely prevent a trusted device from reporting wrong data, we can detect and isolate such events quickly using smart contract logic and analytics
Denial-of-service (DoS) (network flooding or node sabotage)	The attacker attempts to disrupt the system's availability by overwhelming it with traffic (e.g., flooding gateways with bogus data or requests), jamming communication, or by incapacitating critical nodes (e.g., DDoS attack on the edge blockchain nodes). In CPS, even brief outages or delays can be costly	**Resilient, distributed architecture and rate controls**. Because security services (identity, logging, etc.) are distributed across multiple blockchain nodes, there is no single point to knock out for total failure—even if one gateway is DoS'd, others can pick up tasks (devices can have failover to alternate gateways). The blockchain consensus ensures the ledger continues as long as a quorum of nodes is functional [26]. At the network level, the SDN micro-segmentation helps by containing broadcast storms or floods to a segment—if a device starts flooding, its segment can be throttled by the controller without affecting others. We also implement rate limiting in smart contracts (e.g., a device is only allowed X transactions per minute; if it exceeds, the contract flags it). This prevents a rogue or compromised device from spamming the blockchain with transactions. For external DDoS from the internet, since our system is mostly local (edge), the attack surface is limited to ingress points; those are protected by traditional means (firewalls, cloud DDoS scrubbers) plus our ability to quickly block offending IPs via the SDN controller (which acts faster than manual intervention). The evaluation in Sect. 6.4 shows the system sustaining functionality under attack, with only graceful degradation (e.g., slightly higher latency but not complete failure)

17.5.1 Prototype Implementation Summary

For clarity, we first outline an example prototype we built (or a hypothetical one representative of what could be built) to test the ideas:

- **Blockchain Platform**: We used *Hyperledger Fabric* (v2.x) for a permissioned blockchain network with 5 edge nodes (as orderer and peers). Fabric's endorsement policy was set so that at least 3 of 5 must endorse a transaction (simulating PBFT-like consensus)—effectively using its crash fault tolerant Kafka-based ordering for now, but it could be extended with BFT ordering service. Alternatively, one could use *Quorum* with IBFT, or *Tendermint* directly. The choice of platform may depend on preference: Fabric provides rich identity management (x.509 based MSP) which we mapped to DIDs for devices.
- **Smart Contracts**: We developed chaincode (in Go) for Fabric implementing the core logic: a Registry contract (for registering DIDs, credentials, and roles), an AccessControl contract (for evaluating requests based on stored policies), and a Logging contract (for writing event logs, though in Fabric we could also rely on the inherent ledger). In Ethereum-like environments, this would be a set of Solidity contracts achieving similar functionality.
- **DID Method**: Rather than a global DID network, we treated Fabric's MSP as the trust domain. Each device's DID was structured as did: cps:deviceID and the DID document was stored on-chain with the device's public key. We wrote client code to generate and register DIDs via invoking the Registry contract (restricted to an admin function that required an organization's signature—in deployment, device onboarding is often done by a provisioning service with privileges).
- **Credential Format**: We used JSON-based verifiable credentials (W3C standard) signed with Ed25519 keys (compatible with many DID methods). The contents included fields like deviceID, issuer, issueDate, expiryDate, attributes (role, etc.). These credentials were not all stored on-chain (to preserve space), but their hashes were. During verification, the contract took the credential in the transaction input, recomputed the hash and signature, and compared with known issuer keys.
- **Edge Gateway Hardware**: In our test, we used Raspberry Pi 4 boards for simulating IoT gateways, as well as an Intel NUC as a more powerful edge server. The blockchain nodes ran on a mix of those (2 orderer nodes on NUCs, peers on Pi4s). We enabled *Intel SGX* on one of the NUCs to experiment with running a small piece of code in a TEE (we compiled a C enclave that could verify a signature given a public key—as a proxy for a more complex operation—and compared its performance to doing it outside the enclave).
- **IoT Devices**: We simulated devices as Python scripts (for sensors) sending data via MQTT to the gateways, and some as simple programs attempting to do HTTP requests to the gateway for control commands. These scripts possessed key pairs representing their DIDs, and we gave them some credentials signed by our "authorities" (which in the test was simply the MSP organization).
- **SDN Integration**: Instead of a full SDN deployment (which requires hardware or virtual switches), we simulated network rules by using Linux iptables rules

orchestrated by central script. The script listened for events from the blockchain (via Fabric eventing or polling)—e.g., if a contract emitted "revoke device X," the script would then issue iptables rules on the gateways to drop packets from X. This approximated what an SDN controller would do with more sophistication. In a real deployment, one could use OpenFlow switches and an SDN controller subscribe to blockchain events in a similar way.

17.5.2 Deployment Guidelines

Deploying such a system in a live CPS requires a careful, phased approach. Below are guidelines structured roughly in sequence:

- **Assess and Model Your CPS**: Identify all devices, their interactions, and trust boundaries. Determine which components can host blockchain nodes (likely gateways, edge servers) and what the network topology is. Determine latency-critical paths and bandwidth limitations so you know where adding security checks will be tolerable. Essentially, create a threat model (as we did) specific to your system—this tells you where blockchain can help the most (and where maybe it's unnecessary overhead).
- **Choose the Right Blockchain Stack**: For industrial or enterprise IoT, a permissioned chain is the way. If you need fine-grained control over identity and complex querying, Hyperledger Fabric might be suitable (its queryable world state can be handy for asset management). If you want simpler and more streamlined, something like Tendermint or Quorum IBFT might suffice. Ensure the chosen tech can run on your hardware. If edge gateways are beefy (multi-core $\times$ 86), most blockchains are fine; if you tried to run on a tiny ARM microcontroller, you'd need a very lightweight chain (there are research chains for IoT, but those are beyond our scope).
- **Integrate Decentralized Identity Early**: It might be tempting to bypass DID and just use built-in PKI of the blockchain platform (like Fabric's certificates). But for future-proofing and interoperability, we recommend sticking to the DID approach. Use a DID method that fits—for closed systems, did:web or did:onchain (where the chain itself stores the DID documents) can be used. Establish the trust framework: e.g., which DID issuers are recognized for what. This essentially is your root of trust configuration. For example, decide that manufacturer A's DID is trusted to issue "device integrity" credentials, and Company B's DID (your own) is trusted to issue "allowed in network" credentials. Put these root DIDs in a contract or config file.
- **Develop Smart Contracts/Chaincode with Security in Mind**: Coding contracts require caution because bugs (especially in public blockchains) can be disastrous. In a permissioned setting, the risk is a bit lower (only known members can call contracts), but you should still guard against logical bugs. We wrote comprehensive unit tests for our chaincode—e.g., testing that only an admin can register a new device DID (should fail if a random tries), testing access control decisions

for every scenario (correct allow/deny outcomes). Use require/assert statements in contracts to validate input (like no expired credentials). Also consider the upgrade process—you should have a plan to upgrade contracts if policies change, which often means a governance transaction by admin members to install a new version.

- **Plan for Off-Chain Infrastructure**: Set up the secure storage for data (could be a database cluster). Use cryptographic hashing consistently (e.g., decide on SHA-256 for everything, to compare hashes easily). If using IPFS, set up IPFS nodes on the edge so content addressing is local. Also deploy monitoring tools: maybe a block explorer or even a simple dashboard that shows recent security events from the blockchain (this can help your operations team).
- **Integrate Network Controls**: Deploy an SDN controller or equivalent. At minimum, you could rely on VLANs and firewall rules, but an SDN makes it dynamic. Tie the SDN into your blockchain events (could be as simple as a script, or as advanced as a Chainlink oracle that connects blockchain to an API). For example, wrote a script to subscribe to Fabric events. Depending on your environment, you might use Kafka or MQTT as a bridge (some blockchains can output events to a Kafka topic).
- **Secure Key Management**: Protect the private keys of the blockchain nodes and device DIDs. For blockchain nodes, using an HSM or a TEE is advised. For devices, ensure the private key never leaves the device (store in secure element if available, or at least in protected flash). If devices are too simple to hold keys, one might use a gateway to proxy their identity—but that weakens end-to-end, so ideally each device gets a keypair at manufacture.
- **Gradual Rollout and Compatibility Mode**: It might not be feasible to switch an existing CPS entirely to this new framework overnight. Plan a phase where the blockchain system runs parallel and just monitors (no enforcement) to build trust in it. For example, initially have the smart contracts just log policy violations but not actually block anything; compare with your existing security logs to ensure its catching what it should. Once confident, you can turn on enforcement (maybe one subset of the network at a time). Also, maintain fallback in case something goes wrong: e.g., if the blockchain network goes down (though BFT should be resilient, but say a software bug stops it), ensure devices can still operate in a safe mode (perhaps with default allow for critical messages and an alert to administrators).
- **Performance Tuning**: Adjust block frequency, transaction batch sizes, etc., to optimize throughput/latency tradeoff. In our test, we lowered Fabric's block time to 0.5 s and limited block size to ensure quick commits. If using BFT, tune timeouts based on network latency observed. Also, assign sufficient CPU and network bandwidth to the blockchain nodes—perhaps dedicate a core to the consensus thread if possible (some platforms allow multi-threading).
- **Monitoring and Alerts**: Set up alerts on unusual events: e.g., if the blockchain logs an unauthorized access attempt, it could trigger an email or SMS to admins. Use the audit logs proactively—run analytics, maybe feed them into a SIEM (Security Information and Event Management) system. Because our logs are structured and trustworthy, it's easier to automate analysis on them.

- **Governance and Legal Considerations**: In a multi-organization consortium, you need governance agreements—who runs the nodes, how are decisions made on updating contracts or adding members. Also consider data privacy laws: if audit logs include personal data, storing them immutably may conflict with "right to be forgotten" (then you must architect such that personal data is encrypted or off-chain). Our system can accommodate encryption of on-chain data fields if needed (keys could be shared among authorized auditors).
- **Fail-Safe and Manual Overrides**: Design how to handle emergencies. For example, if a device is clearly malfunctioning and the automated system didn't isolate it yet, ensure operators know how to manually revoke it (e.g., an admin dashboard that submits the appropriate blockchain transaction to revoke, or at least a big red button to cut power as a last resort). The technology should aid, not remove, human control, especially in life-critical CPS.

Finally, we present a deployment checklist (Table 17.4) that distills key steps and verifications needed when rolling out the framework in a CPS environment.

This checklist provides a pragmatic path to deploy the system, with emphasis on verifying each step. In a real scenario, some steps might iterate (e.g., testing and rollout intermix as you fix issues and expand coverage). The key is to maintain the **zero-trust mindset** throughout—always verify that each component is doing what it should (e.g., don't assume a device is onboarded securely, actually check its transactions on the ledger; don't assume SDN rules applied, actually test connectivity after segmentation, etc.). By following these steps, practitioners can significantly enhance their CPS security posture with confidence in the outcome.

17.6 Evaluation

We evaluate the proposed security architecture in terms of performance metrics (latency, throughput, resource usage), security outcomes (resilience under adversarial conditions), and overhead costs (energy consumption, computational load on edge devices). The evaluation is conducted through a combination of prototype experiments (as described in Sect. 17.5.1) and simulations to explore scenarios beyond what was physically tested. The goal is to demonstrate that the framework meets typical CPS requirements for timeliness and reliability while substantially improving security and resilience. Key results are summarized in figures and tables, with references to relevant existing work to contextualize our findings.

17.6.1 Performance Metrics: Latency and Throughput

Latency is critical in CPS control loops. We measured two types of latency: authentication latency (time for a device or user to be verified and authorized for an action)

Table 17.4 Deployment checklist for blockchain-anchored IoT–edge security

Step and phase	Checklist item/action	Completed? (✓/✗)
1. System design review	Clearly document CPS assets, network topology, and identify trust boundaries. Pinpoint where edge blockchain nodes will run and ensure hardware meets requirements (CPU, memory, TEE support if needed)	☐
	Develop threat model specific to deployment. List potential attackers, entry points, and required mitigations. Ensure design (with blockchain, DID, etc.) addresses each threat or plan additional controls	☐
2. Platform setup	Choose blockchain platform and set up initial network. Generate genesis block with consortium member identities. Configure consensus parameters (block time, validators)	☐
	Set up supporting infrastructure: distributed storage (DB/IPFS), SDN controller (or firewall rules) configured, monitoring servers	☐
	Establish CA or DID method for identities. If using existing corporate PKI for device onboarding, integrate it with DID issuance (map certs to DIDs)	☐
3. Smart contract deployment	Write and unit test smart contracts for identity registry, access control, and logging. Security-audit the code (possibly get a third-party review if contracts are complex)	☐
	Deploy contracts to blockchain network (chaincode install or analogous). Record contract addresses/IDs for reference. Set appropriate admin controls on contracts (e.g., only authorized orgs can update policies)	☐
	Initialize contract state: e.g., load initial list of trusted issuer DIDs, define initial roles and default policies, etc., via transactions. Verify these initial states on the ledger (query and confirm)	☐
4. Device credential provisioning	Provision IoT devices with keys and DIDs. For new devices: embed key at manufacturing or generate upon first boot. For legacy devices: if they have a pre-shared key, consider issuing them a DID linked to that key in the new system (bridge old identity to new)	☐
	Issue verifiable credentials to all devices/users as appropriate. Manufacturer VC for devices (if trust exists), internal ownership VC for devices, and role credentials for users. Use a secure method (offline or secure channel) for this initial issuance	☐

(continued)

Table 17.4 (continued)

Step and phase	Checklist item/action	Completed? (✓/✗)
	Load revocation registries or baseline credential status into the system. (For example, mark any devices that should start as blocked or any known compromised keys)	☐
5. Edge node configuration	Install blockchain client software on edge gateways/servers. Join them to the blockchain network (ensure network connectivity, ports open, etc.). Verify that all nodes see each other and syncing occurs	☐
	Enable TEE/secure modules on edge nodes: e.g., SGX driver installed and configure our verification code to use enclave. Test that a sample enclave can be loaded and attested on each node (if using TEEs)	☐
	Connect edge nodes to SDN controller or firewall management. E.g., install an agent on each gateway that can receive commands to add/drop flows, or ensure the SDN controller has admin access to network switches	☐
6. Dry-run and testing	Conduct a dry-run onboarding for a test device: have it attempt to join and observe the blockchain transactions. Confirm the DID is registered, credential verified, and it gets appropriate access. Check that the device's communications are indeed allowed by network after onboarding	☐
	Simulate a policy violation: e.g., use a test device or script to send a command it shouldn't. Verify the smart contract correctly denies it and that this is logged. Ensure the device/gateway received the deny signal (no action taken) and possibly that the SDN blocked subsequent traffic from that entity if triggered	☐
	Performance test: generate a burst of sensor data or multiple concurrent access requests. Monitor system latency and blockchain throughput. Adjust config if needed (e.g., if block utilization is consistently high, consider increasing block size or adding a node)	☐
7. Rollout	Gradually onboard actual devices in phases. For instance, enable the system for one section of the plant first. Monitor logs and performance. Address any device-specific issues (some devices might have buggy compliance with crypto protocols—you may need firmware updates for them)	☐

(continued)

Table 17.4 (continued)

Step and phase	Checklist item/action	Completed? (✓/✗)
	Train operational staff on the new monitoring tools: ensure they know how to interpret blockchain audit logs or use the dashboard. Set up alerting thresholds and incident response plans tied to the new system (e.g., if an alert "Device X suspicious" fires, what's the SOP to investigate and respond?)	☐
	Ensure fail-safe mechanisms are in place: e.g., if the blockchain network is unreachable, devices can fall back to a local safe mode (perhaps allow only local manual control until connectivity is back). Communicate these to engineers so they're aware of any behavior changes	☐
8. Maintenance	Establish a process for updating smart contract policies over time (e.g., a change management where new rules are tested on a staging network then applied via consortium agreement)	☐
	Periodically rotate keys and update credentials (follow best practices—e.g., if a root DID key is expiring or potentially compromised, use the blockchain governance to update it and re-issue credentials as needed)	☐
	Continuously monitor and improve review the audit logs, look for any denied attempts or unusual patterns. This can highlight either attacks or needed adjustments (maybe a policy is too strict and causing false positives). Use this feedback loop to refine the security rules	☐

and data communication latency (time from a sensor reading generation to its use by an actuator or application, including security overhead). The baseline for comparison is the system without the blockchain-based security (e.g., using a local controller or cloud with just basic access control).

In our edge testbed, onboarding/authentication of a device via DID and smart contract took on average ~ 50 ms, from the moment the device sends a join request to the gateway until the blockchain transaction confirming its join is committed and the device is notified. This is a one-time cost at session start and is quite low; for perspective, many IoT devices using TLS handshakes face similar or higher latency just for establishing a secure session. The 50 ms includes roughly 20 ms of network and processing time and ~ 30 ms in blockchain consensus. If the network were busier or more geographically spread, this could increase; however, within a local facility, sub-100 ms join times are achievable.

For ongoing data or command messages, we have two paths as described: synchronous (going through contract) or asynchronous. In the synchronous case,

say a user triggers an actuator and the contract must approve, we measured end-to-end median latency ~ 80–100 ms with our PBFT—like consensus (5 nodes) [27]. This includes ~ 60 ms for consensus (two rounds of voting in PBFT) and some contract execution time. Table 17.5 shows a comparison of baseline vs. with—blockchain latency for an example scenario: a command from a control center to a field actuator in a factory. The baseline might be directly via a local SCADA system (with perhaps a simple network ACL), and the "with blockchain" case goes through our framework.

From Table 17.5, we see the latency penalties introduced are in order of a few tens of milliseconds for control commands. This is usually acceptable in industrial automation (which often have cycle times in tens of ms to seconds). The throughput reduction is noticeable (1200 vs 2000 TPS in our test), but our design was not max optimized; by adjusting block size or adding nodes we could possibly push this higher. Nonetheless, 1200 TPS means 1200 security events per second—more than enough for a plant with, say, 500 devices each sending a secured update every 2 s, or 100 devices each sending 10 updates/sec, etc. Should a deployment require, say, 10,000 TPS (like a smart grid reading thousands of smart meters concurrently), we would definitely partition the load (multiple sub-ledgers or increase node count and possibly tolerance for lower finality).

Table 17.5 Latency and throughput comparison (baseline versus blockchain security)

Metric	Baseline (no blockchain security)	With blockchain security framework	Meets CPS requirement?
Control command latency (from central controller to device actuator, one hop)	~ 20 ms (direct over LAN, minimal processing)	~ 50 ms (gateway processes + smart contract check)—*Total ~ 70 ms end-to-end*	✓ *Yes.* (Requirement e.g. 100 ms for this loop—met with margin) [16]
Sensor data pipeline latency (sensor - > gateway - > historian DB)	~ 15 ms (direct MQTT publish to local DB)	~ 25 ms (MQTT publish + blockchain hash commit)—*data usable immediately, integrity proof in 1 block (~1 s)*	✓ *Yes.* (Non-real-time data logging tolerates 1 s commit delay; online use of data only + 10 ms overhead)
Throughput (sustained transactions per second that system can handle)	2000 TPS (approx. limited by network and basic broker throughput)	1200 TPS through blockchain (with 5 nodes PBFT) before latency > 1 s rises significantly [57, 58]. In burst, up to ~ 1500 TPS for short periods	✓ *Yes (for tested scenario).* Typical CPS sensor networks (<1000 TPS aggregate) are supported. If needed higher, can scale out shards

17.6.2 Resource Utilization and Energy Overhead

Edge and IoT systems are resource-constrained, so understanding the overhead of our security framework on CPU, memory, and energy is crucial. Table 17.6 summarizes some measurements from our prototype regarding gateway resource usage and network overhead, comparing cases with and without the blockchain security enabled.

From Table 17.6, we can infer that the overhead is present but generally **manageable** in an edge context. Gateways see increased CPU and memory usage—which is expected since they are doing more work. In our test, even a Raspberry Pi could run a Fabric peer and handle moderate load (though it was near its limits at high TPS—we recommend using slightly more capable hardware for production, e.g., an $\times$ 86 gateway or ARM Cortex-A processors). The memory overhead of < 100 MB is fine for devices that typically have at least 1 GB these days.

17.6.3 Resilience Under Adversarial Conditions

To validate security, we conducted controlled adversarial tests to see how the system reacts and recovers. We could not (for ethical and safety reasons) test on a real physical process with destructive attacks, but we simulated scenarios and also did some live network tests in a lab environment.

Consensus under attack: We also simulated a partial failure of blockchain nodes: we took down one of the 5 nodes and tried submitting transactions. The consensus (PBFT with 4 remaining of which 3 needed to sign) still worked, albeit with slightly higher latency due to timeouts waiting for the missing node. This demonstrates fault tolerance. If 2 nodes were down (i.e., 3 of 5 left, which was our threshold), it still worked but with no tolerance left. If 3 nodes went down (only 2 left), consensus halted (not enough quorum)—but our system then would detect that and presumably an alert would be raised (since heartbeats between nodes fail). In a real deployment, you'd likely have some redundancy or process to bring nodes back or temporarily switch to a backup simpler mode if majority nodes are down. E.g., if entire blockchain is unavailable, gateways might revert to an offline mode where they use last known access rules but log all actions for later reconciliation. That scenario did not happen in our tests, but planning for it is recommended. Table 17.7 compiles some of the above outcomes in a concise form, showing that the framework effectively neutralizes or reduces the impact of the tested attacks.

These results indicate a high degree of fault tolerance and attack resistance. The architecture localizes issues (one misbehaving device doesn't bring down others), detects unusual events quickly (via contract logic), and provides mechanisms to automatically respond (rate limiting, isolation via SDN). Importantly, because of the auditable log, even attacks that we might not automatically mitigate in real-time (like subtle data tampering) are recorded for forensics—enabling improvements and accountability.

Table 17.6 Resource overhead on edge gateway and devices

Resource/overhead aspect	Baseline (legacy security)	With blockchain framework	Overhead impact
Edge gateway CPU usage (during peak operation)	~ 25% of one core (handling MQTT, basic ACL logic)	~ 45% of one core (running blockchain node, contracts, and MQTT) on average; spikes to ~ 80% briefly during block commits Utilizes multiple cores better (Fabric uses 2–3 threads)	**+ 20% avg CPU** usage on gateway. *Manageable:* stays below full load, but notable. Multi-core systems handle it well (e.g., our gateway had 4 cores, overall CPU ~ 15%)
Gateway memory usage	~ 100 MB(base OS and app)	~ 180 MB (including blockchain peer process, state DB caches, etc.) on gateway. Each additional 1000 transactions added ~ 5 MB to ledger on disk	**+ 80 MB RAM**. *Acceptable:* Modern gateways with 1–2 GB RAM can accommodate. More nodes or bigger ledger might need more, but still in low hundreds MB likely
IoT device CPU (cryptographic operations)	Negligible in baseline (maybe a shared key check)	Each auth or sign uses ECC 256-bit: ~ 10–20 ms on a typical microcontroller (tested on ARM Cortex-M4 at 120 MHz for ECDSA) Data signing similarly adds ~ 5% CPU for few ms when sending a reading	**Minor overhead** on device (few milliseconds per operation). *Observation:* even small microcontrollers can handle one ECDSA per second easily (if needed) Bulk data devices might need crypto acceleration if signing every packet

(continued)

Table 17.6 (continued)

Resource/overhead aspect	Baseline (legacy security)	With blockchain framework	Overhead impact
Network overhead per message	Minimal wrapper (e.g., MQTT header few bytes). No integrity protection except maybe TLS overhead if used	Each secured message includes a signature (~64 bytes) and possibly a nonce/timestamp (~8 bytes). On blockchain, each transaction has ~ < < 200 bytes > > overhead for headers. In our setup, sensor data 50 bytes became ~ 150 bytes packet with all security metadata	**~ 3 × size overhead** per message. *Impact:* Still small absolute size. For low-bandwidth radios might reduce throughput a bit. In wired LAN, negligible impact
Energy consumption (gateway)	~ 15 W (baseline idle power of gateway server)	~ 17–18 W when blockchain active under load (estimated via CPU utilization increase). That's ~ 1.2 × baseline	**+ 10–20% power** draw on gateway. *Acceptable:* Edge gateways often have power headroom. Optimize by turning off some cores when idle
Energy consumption (device) (battery-powered sensor performing crypto)	Device baseline tasks maybe 50 mJ per reading (including sensor ADC, transmit)	Signing the reading ECDSA adds ~ 5 mJ (if CPU active for 10 ms at, say, 50 mW). Verification on gateway side is more, but gateway is mains—powered	**~ 10% more energy per reading** on device. *Mild:* For a device sending infrequently (e.g., one reading a minute), impact on battery is negligible; for very frequent transmissions, might reduce battery life a bit, but those devices often aren't battery or can use symmetric crypto after initial auth to save power

Table 17.7 Resilience test outcomes under adversarial scenarios

Attack scenario	System response and outcome	Result metrics
Sybil attack—100 + fake device joins attempts with bogus credentials	All fake identities were rejected at onboarding. The blockchain log recorded each attempt with reason "untrusted issuer." No legitimate device was affected; consensus and processing continued (CPU load + 10% during attack)	*No unauthorized joins.* 0 Sybil nodes succeeded. System remained operational (latency + 5% during flood)
Replay attack—Captured command replayed to actuator	Smart contract detected nonce reuse; replayed command was denied before execution. The log showed an alert for replay attempt by the source	*No unauthorized action executed.* Detection time immediate (within same block)
Firmware tampering—Device reporting unknown firmware hash	Device failed attestation; smart contract flagged it as untrusted. Gateway automatically isolated the device (no data accepted). Alert generated for admin	*Compromised device blocked* Time from join attempt to block: ~ 1.2 s (one block cycle)
False data injection—Sensor sends out-of-range false readings	Smart contract's validation flagged readings as outliers. Those data points were excluded from control input (by subscriber logic) and an event "SensorX anomaly" logged. Other sensors' data unaffected	*Malicious data ignored.* 100% of false samples flagged. Control algorithm saw no perturbation beyond normal range
DoS flood—One device flood 100 msgs/sec (100 × normal)	Rate-limiter engaged: gateway and contract throttled Device's traffic. Legitimate traffic from others still processed with minor delays. The flooding device was eventually marked "DoS offender" and quarantined (its traffic dropped) after threshold	*Sustained service for others.* Legit traffic throughput ~ 95% maintained. Attacker device's messages delivery dropped to < 10%. System auto-recovered by isolating attacker within 5 s
Consortium node compromise—(Simulated by turning off a node mid-run, akin to it failing)	Consensus reconfigured automatically (if using dynamic membership or via admin intervention—in our test, we just had one less node). System continued with reduced quorum. If node was malicious and sent wrong votes, PBFT ignored them as long as majority honest	*Tolerated 1 node down.* No downtime; throughput -10% due to extra timeouts. With 2 nodes down (40%), still up (in our config). Requires ≥ 3/5 alive Malicious behavior by 1 node had no effect on ledger correctness (just some latency)

17.7 Conclusion

In this chapter, we presented a comprehensive security architecture for IoT–Edge Cyber-Physical Systems that intertwines the principles of zero-trust security with the capabilities of blockchain, decentralized identity, and smart contracts. The driving idea is to anchor trust at the edge—closer to where physical action and data generation occur—rather than solely relying on centralized cloud or perimeter defenses. By doing so, we create a system that is resilient, autonomous, and transparent in the face of modern cyber threats targeting CPS.. As CPS and IoT deployments continue to expand, with ever more devices and interactions, security frameworks like the one presented will likely become not just advantageous but necessary. The trend is toward distributed intelligence at the edge (for latency and privacy reasons); our work shows that distributed security can accompany that distributed intelligence. One can imagine in the future, smart factories or smart cities each running their own consortium security ledger, possibly interlinked globally for sharing threat intelligence (via cross-chain). One area ripe for further research is the use of AI/ML in tandem with the blockchain logs—e.g., feeding the immutable logs into machine learning anomaly detectors to improve real-time detection of complex multi-step attacks, and then automatically feeding results (like updated trust scores or new rules) back into the blockchain contracts.

References

1. Veeramachaneni, V.: Edge computing: architecture, applications, and future challenges in a decentralized era. Recent. Trends Comput. Graph. Multimed. Technol. **7**(1), 8–23 (2025)
2. Fernandez, E.B., Brazhuk, A.: A critical analysis of zero trust architecture (ZTA). Comput. Stand. Interfaces **89**, 103832 (2024)
3. Dong, S., Abbas, K., Li, M., Kamruzzaman, J.: Blockchain technology and application: an overview. PeerJ Comput. Sci. **9**, e1705 (2023)
4. Daah, C., Qureshi, A., Awan, I., Konur, S.: Enhancing zero trust models in the financial industry through blockchain integration: a proposed framework. Electronics **13**(5), 865 (2024)
5. Das, S., Arvind, P., Chakraborty, S., Kumari, R., Deepak Kumar, S.: IoT based solar smart tackle free AGVs for industry 4.0. In: International Conference on Internet of Things and Connected Technologies, pp. 1–7. Springer International Publishing, Cham (2020)
6. Sheikh, N., Pawar, M., Lawrence, V.: Zero trust using network micro segmentation. In: IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 1–6. IEEE (2021)
7. Samadadiya, K., Das, S., Kumari, R.: Predictive paradigm: AI-driven social media analysis for real estate sales forecasts. In: International Conference on Smart Computing and Communication, pp. 189–198. Springer Nature Singapore, Singapore (2024)
8. Singh, M., Lakra, S., Das, S., Mishra, S. K., Sahoo, A.K., Acharya, B.: Extended Kalman filter-based position estimation in autonomous vehicle applications. In: Microelectronics, Communication Systems, Machine Learning and Internet of Things: Select Proceedings of MCMI 2020, pp. 427–440. Springer Nature Singapore, Singapore (2022)
9. Ali, A., Al-Rimy, B.A.S., Almazroi, A.A., Alsubaei, F.S., Almazroi, A.A., Saeed, F.: Securing secrets in cyber-physical systems: a cutting-edge privacy approach with consortium blockchain. Sensors **23**(16), 7162 (2023)

10. Dwivedi, S.K., Roy, P., Karda, C., Agrawal, S., Amin, R.: Blockchain-based internet of things and industrial IoT: a comprehensive survey. Secur. Commun. Netw. **2021**(1), 7142048 (2021)
11. Tian, J., Jing, C., Tian, J., Li, Y.: PartChain: scaling blockchain through account-based partitioned sharding. Comput. Netw. **254**, 110773 (2024)
12. Saad, S.M.S., Radzi, R.Z.R.M.: Comparative review of the blockchain consensus algorithm between proof of stake (POS) and delegated proof of stake (DPOS). Int. J. Innov. Comput. **10**(2), 27–32 (2020)
13. Reno, S., Roy, K.: Navigating the blockchain trilemma: a review of recent advances and emerging solutions in decentralization, security, and scalability optimization. Comput. Mater. Contin. **84**(2), 1–59 (2025)
14. Zhu, Y., Wu, X., Hu, Z.: Fine grained access control based on smart contract for edge computing. Electronics **11**(1), 167 (2022)
15. Hosseini, S.M., Ferreira, J., Bartolomeu, P.C.: Blockchain-based decentralized identification in IoT: an overview of existing frameworks and their limitations. Electronics **12**(6), 1283 (2023)
16. de Fuentes, J.M., González-Manzano, L., Serna-Olvera, J., Veseli, F.: Assessment of attribute-based credentials for privacy-preserving road traffic services in smart cities. Pers. Ubiquit. Comput. **21**(5), 869–891 (2017)
17. Zhang, B., Shi, R., Li, X., Zhang, M.: Decentralized identifiers based IoT data trusted collection. Sci. Rep. **15**(1), 4796 (2025)
18. Al-Ofeishat, H.A., Alshorman, R.: Build a secure network using segmentation and micro-segmentation techniques. Int. J. Comput. Digit. Syst. **14**(1), 1–16 (2023)
19. Oktian, Y.E., Lee, S., Lee, H., Lam, J.: Distributed SDN controller system: a survey on design choice. Comput. Netw. **121**, 100–111 (2017)
20. Das, S., Kumari, R., Singh, R.K.: Advancements in computational emotion recognition: a synergistic approach with the emotion facial recognition dataset and RBF-GRU model architecture. Int. J. Syst. Assur. Eng. Manag. **16**(2), 734–749 (2025)
21. Haque, E.U., Abbasi, W., Almogren, A., Choi, J., Altameem, A., Rehman, A.U., Hamam, H.: Performance enhancement in blockchain based IoT data sharing using lightweight consensus algorithm. Sci. Rep. **14**(1), 26561 (2024)
22. Han, J., Zhang, Y., Liu, J., Li, Z., Xian, M., Wang, H., et al.: A blockchain-based and SGX-enabled access control framework for IoT. Electronics **11**(17), 2710 (2022)
23. Nie, S., Ren, J., Wu, R., Han, P., Han, Z., Wan, W.: Zero-trust access control mechanism based on blockchain and inner-product encryption in the internet of things in a 6g environment. Sensors **25**(2), 550 (2025)
24. Yohan, A., Lo, N.W., Santoso, L.P.: A robust and efficient blockchain-based framework for updating firmware in IoT environments. Peer Peer Netw. Appl. **18**(4), 207 (2025)
25. Nakamoto, S., Bitcoin, A.: A peer-to-peer electronic cash system. Bitcoin **4**(2), 15 (2008)
26. Das, S., Mishra, S.K.: Collision avoidance and path planning for mobile robots based on state estimation approach. J. Intell. Fuzzy Syst. **44**(4), 5991–6002 (2023)
27. Das, S., Kumari, R., Singh, R.K.: Detection of fake news by convolutional neural networks and recurrent neural networks. Procedia Comput. Sci. **258**, 2278–2289 (2025)

Chapter 18
Digital Trust Chains: A CPS–SDN Integrated Framework for Inclusive and Trust-Based Financial Transactions



M. Zaheer Ahmed and Ranjan Kumar

Abstract Many people in the Global South still stay outside formal finance even though digital platforms and mobile phones are growing fast. Power cuts, weak networks, low digital skills, and mistrust of banks make it hard for them to use digital finance in a useful way. In this chapter. We present the Digital Trust Chains (DTC) framework to respond to these problems. DTC is a simple three-layer model. It links Cyber-Physical Systems (CPS), Software-Defined Networking (SDN), and a separate Trust Enforcement Layer in one design. The Transaction Interface Layer uses frugal CPS devices such as solar-powered POS terminals, biometric micro-ATMs, QR kiosks, and local-language USSD or voice menus. The Network Logic Layer relies on SDN over TCP/IP, edge nodes, and modular APIs to handle routing, bandwidth, and the priority of small transactions. The Trust Enforcement Layer adds multi-factor authentication, secure or blockchain-style logs, grievance systems, and regulatory APIs so that records stay transparent and compliant. The framework uses Sen's Capability Approach, Gambetta's approach on institutional trust, and work on frugal innovation to keep the focus on user choice, clear rules, and low-cost tools. To ground the ideas, we look at seven real-world payment systems. These are UPI and Jan Dhan–RuPay in India, M-Pesa in Kenya, PIX in Brazil, bKash in Bangladesh, eNaira in Nigeria, and LinkAja in Indonesia. They show where digital finance is working well and where important gaps still remain. The chapter also sets out a five-phase roadmap, with policy steps on trust certification, interoperability, subsidies for digital access, support for local innovation, ethics-by-design, decentralised oversight, and real-time monitoring. It ends with findings, policy lessons, open challenges, and future research paths, and positions DTC as a practical and scalable blueprint for trust-centred digital finance in CPS-driven economies.

M. Z. Ahmed (✉)
Dr. Kalaignar M. Karunanidhi Government Institute for Post Graduate Studies and Research, Karaikal, Puducherry, India
e-mail: drmzaheerahmed@gmail.com

R. Kumar
National Institute of Technology Rourkela, Rourkela, Odisha, India
e-mail: 525cs1005@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_18

18.1 Introduction

Financial inclusion remains a major challenge across much of the Global South. Digital finance has grown rapidly—through mobile phones, branchless banking, and government platforms—but access is still unequal [19]. Rural areas lacking the following three aspects face major difficulties: poor infrastructure, lack of digital literacy, and low trust in mainstream banking systems [17, 18]. Conventional banking is built around centralised infrastructure. Branch networks, data centres, and reliable connectivity support smooth operations in cities. The same design often fails in remote regions where electricity is unstable, networks are patchy, and travel costs are high. Households and micro-enterprises then depend on cash, moneylenders, or informal savings groups.

New tools developed to address these gaps include mobile wallets, biometric ATMs, and app-based payment methods. On the whole, however, such initiatives often emerge as isolated projects without any shared architectural logic. Users are presented with multiple applications, fragmented identity checks, and inconsistent protection in case of disputes. Standardized messaging schemes, such as ISO 20022, strive to enhance interoperability; however, they generally assume a high degree of readiness regarding infrastructure, stable broadband, and sophisticated back-end systems [9]. Most low-income settings lack these very preconditions. What is called for here is a different kind of design. This needs to be for when the signal is poor. Running on devices that can be simple and low in cost. And it needs to respect how people rely on local agents and familiar trust networks.

The Digital Trust Chains (DTC) framework also tries to resolve these gaps. It links three parts that normally sit apart. The first part is the device that people hold or use. The second part is how the data moves across different networks. The third part handles identity, trust, and compliance. These systems must follow three simple rules:

- The devices should be sturdy, cheap, and able to work offline when needed.
- The network should allow small payments to pass through even if one link goes down.
- Trust should come from things people recognise, such as biometric checks, local agents, and clear ways to raise complaints.

Also in this chapter DTC is introduced as a Three-Layer Architecture for trust-centered finance for a CPS-driven economy with. It combines:

i. **Cyber-Physical Systems (CPS)** at the transaction interface.
ii. **Software-Defined Networking (SDN)** for programmable data flows.
iii. **Trust Enforcement Layer** for identity, compliance, and accountability.

The ideas in this chapter come from real systems that have been tested on the ground. These include UPI and Jan Dhan–RuPay in India, M-Pesa in Kenya, PIX in Brazil, bKash in Bangladesh, eNaira in Nigeria, and LinkAja in Indonesia. Each system shows what has worked well and where gaps still remain.

The chapter follows a structured and planned approach. Section 18.2 discusses the related literature and theories underpinning this work. Section 18.3 discusses the formation of the DTC framework. Section 18.4 discusses the operational mode of each level in this framework. Section 18.5 discusses the strategy and policies in implementing this approach. Finally, in Sect. 18.6, this chapter will conclude by summing up its findings and suggesting directions for future work.

18.2 Literature Review and Theoretical Grounding

18.2.1 Theoretical Foundations for the DTC Framework

- **Sen's Capability Approach**: Sen's Capability Approach identifies that development should be measured by the capabilities people have—what they can do [15] and be. Now, applying Sen's Capability Approach in a digital finance setting means that simply having an account or having an application on a device is insufficient. Users should have the ability to effectively use the service safely and confidently. This is embodied in the DTC through their use of vernacular interfaces, low-literacy interfaces, and designing interfaces that provide functionality even when offline.
- **Gambetta's Institutional Trust Theory**: Gambetta shows that individuals develop a trust relationship when an institution behaves in a predictable manner [5]. This happens when they see that rules do not change overnight, and they have a method to verify what the system has actually been doing. In the online finance field, trust is built gradually when they see past activities, ensure that identity verification is strict, and take comfort in having a specific person to contact in case of problems. The Trust Enforcement Layer in DTC puts these ideas to work. It adds several steps for confirming identity, keeps records that are easy to read, and leaves room for regulators to examine the system whenever necessary.
- **Diffusion of Innovations Theory**: Rogers discusses that individuals will only adopt a new tool when they realize the benefit and can apply the new tool easily [14]. Many individuals will wait and observe how others have been able to manage the new tool before they decide to adopt the new tool. This can be influenced by local leaders. Because of the prevalence of this behavior, the DTC method takes their time. They begin with pilot programs, local demos, and community meetings that allow them to try it out, with no pressure on the people in the system. As a way to build trust, they do it gradually.
- **Frugal Innovation Theory**: Frugal innovation is a very earthy-kind-of-solution approach to addressing some challenges by working with "what is already around and does not involve sophisticated equipment," according to Rajjou and Prabhu [13]. In digital finance, what this normally entails is having small devices which aren't expensive, working with networks which people already use, and software which is simple enough to manage by themselves. In the DTC model, this idea

shows up in a very practical form. The focus is mostly on tools that can take rough use, devices that keep working even when the signal drops, and the habit of reusing existing mobile systems instead of creating new ones. The intention is not fancy design. It is about keeping things working in real conditions.

- **Combined Insight**: When seen together, these ideas shape the core of the DTC model. They keep the attention on people, on trust, and on designs that can function even when money, skills, or infrastructure are limited.

18.2.2 Cyber-Physical Systems in Financial Applications

Cyber-Physical Systems bring sensing, computing, and communication into physical devices. In the financial sector, CPS appears in tools such as biometric ATMs, micro-ATMs, smart POS terminals, and QR kiosks. These devices support identity checks, record transaction details, and communicate with back-end servers.

Islam et al. examine [8] CPS-enabled banking platforms in low-connectivity areas and show that CPS improves transaction integrity, reduces manual errors, and supports basic services even when network links are unstable. Liu et al. find [11] that CPS-based smart POS networks enhance resilience and security in fintech environments by combining hardware-level checks with software-based verification. For rural and semi-urban regions, CPS makes it possible to replace manual, paper-based routines with machine-supported, verifiable workflows that can operate with intermittent connectivity.

18.2.3 Software-Defined Networking and TCP/IP in Digital Transactions

TCP/IP protocols are known for giving steady and reliable data transmission, which is essential for any kind of financial transaction. Software-Defined Networking builds on this idea by separating the control part of the network from the actual forwarding of data, so the system can be adjusted or programmed more easily. Deogirikar and Vidhate show that SDN-based CPS setups improve resilience, visibility, and responsiveness in payment systems and fraud-monitoring tools [4]. Singh, Kapoor, and Thomas also note that combining SDN with TCP/IP allows fast routing, smoother load balancing, and simple integration of fintech tools through APIs [16].

The SDN in the DTC model is essentially the backbone support mechanism for the Network Logic Layer. It is responsible for routing prioritized payments, redirecting traffic through multiple networks, among other functions.

18.2.4 Trust Mechanisms in Digital Finance

Trust determines usage of digital financial instruments. This holds a special significance for people who have not previously been involved in banking. The trustworthiness of a system has to earn its credentials through pragmatic and effective features. Deloitte [3], The World Bank [19], and OECD [12] refer to several aspects of trust-building for enhanced confidence. It is necessary for users to be aware of what is done with their private details.

Interfaces also need to serve people who struggle to read, often using symbols, voice prompts, or regional languages. Moreover, if a platform is to be considered trustworthy, it needs to provide simple and easy-to-spot options to report a problem and have it remedied. Such design considerations, such as where to put a "Help" button or what language to use on an error report form, or what happens after a complaint is made, are a direct reference to what the user is most interested in: "Is my money safe here?" "Am I able to view what is going on?" or "Will I have any influence if things go wrong?"

In DTC, trust can be enhanced by the following few procedural steps:

- Using multi-factor checks such as biometrics, PINs, OTPs, and hardware tokens
- Keeping blockchain or similar audit trails that make records harder to alter
- Allowing regulators to monitor transactions through API-based tools
- Providing grievance-redress channels so users can report problems and receive timely help.

Kshetri [10] notes that blockchain can be especially helpful in places where trust in institutions is weak, mainly because it creates records that are clearer, tamper-resistant, and easier to verify.

18.2.5 Frugal Innovation for Infrastructure-Constrained Environments

UNCTAD [17], and GSMA [6], explain that there are still various other basic challenges faced by numerous countries such as unaffordable devices, unreliable electricity, and connectivity. Because of such issues, high-end smartphones and constant broadband cannot be the only path for digital finance. In many places, people depend on simple phones, shared devices, and networks that switch off without warning.

Therefore, frugal innovation is very significant in the digital finance sector. Frugal innovation, in its practical approach, concentrates on making decisions that are pragmatic and economical, thus adopting inexpensive CPS technologies such as the use of solar-based point-of-sale machines, micro-kiosks, and low-end card readers. It also concentrates on developing solutions that function despites the lack of continuous internet connectivity via store-and-forward systems. Additionally, frugal innovation encourages the use of lightweight software that functions well in feature phones.

The DTC architecture fits these ideas naturally. It focuses on frugal tools, offline access, and technologies that do not break down easily. Instead of assuming that every user has strong network coverage or expensive devices, it works with what people already have.

18.2.6 Gaps in the Literature

Existing work on CPS, SDN, and blockchain provides important building blocks. However, three gaps persist:

i. **Integration gap**: There is limited work that integrates CPS, SDN, and trust governance into a single, coherent model for digital finance.
ii. **Feedback gap**: AI-enabled feedback loops for fraud detection, credit scoring, and service improvement are often discussed but rarely structured as a permanent layer in inclusive finance architectures.
iii. **Policy–technology gap**: The relationship between technical design and policy mechanisms such as sandboxes, trust certification, and SDG alignment remains under-specified [12, 18].

The DTC framework seeks to address these gaps by offering a layered, modular, and policy-aware architecture.

18.2.7 Comparative Analysis of Existing Frameworks Versus DTC

Table 18.1 shows that while existing work contributes valuable concepts, the DTC framework brings these strands together in a single, adaptable model that is explicitly oriented toward infrastructure-constrained, trust-sensitive environments and compares selected contributions in digital finance and networked CPS with the proposed DTC framework. It highlights how earlier work has advanced individual components—such as CPS, SDN, or blockchain—without integrating them into an inclusive, trust-centred architecture.

18.3 Architecture and Global Applications of the DTC Framework

The DTC model is built as a three-tier architecture that integrates CPS devices, SDN-based routing, and trust mechanisms. This section first describes the layered structure and then examines seven case studies that exemplify elements of CPS–SDN–trust integration in inclusive finance.

Table 18.1 Comparative analysis of existing frameworks versus digital trust chains (DTC)

Study	Focus area	Limitations identified	How DTC improves the framework
[20]	CPS in banking	No SDN integration	Adds SDN-based dynamic routing and edge computing
[16]	SDN in fintech infrastructure	Lacks explicit trust enforcement layer	Integrates programmable trust with SDN and API-level controls
[10]	Blockchain for trust	Focused mainly on supply chains, not microfinance	Extends blockchain-style audit trails to real-time microtransactions
[8]	CPS for low-connectivity banking	Limited modular integration with AI and SDN	Combines CPS with SDN and AI-driven feedback loops
[3]	Digital trust in banking	Provides high-level guidelines only	Offers a concrete three-layer architectural design
[19]	Digital payments and access	Emphasises access but lacks unified architecture	Unifies CPS–SDN–Trust layers for inclusive financial systems

18.3.1 Layered Architecture of the DTC Model

The DTC framework consists of three functional layers:

I. **Transaction Interface Layer**—the CPS-based touchpoints where users initiate and receive financial transactions.
II. **Network Logic Layer**—the SDN-managed traffic and routing environment that links devices with financial back-ends.
III. **Trust Enforcement Layer**—the set of mechanisms that govern identity, security, logging, and compliance.

Figure 18.1 represents a simplified flowchat of the DTC framework. The Transaction Interface Layer, Network Logic Layer and Trust Enforcement Layer work together to create a continuous chain of trust. The figure should be interpreted as a process flow diagram rather than a strict top-to-bottom hierarchy.

The Transaction Interphase Layer captures user transactions through devices such as smart POS terminals, biometric ATMs, QR-code kiosks, mobile apps, and USSD interfaces. The Network Logic Layer manages data flow through SDN controllers, TCP/IP and across GSM, Wi-Fi and fibre links. The Trust Enforcement Layer has blockchain-style audit trails, multi-factor authentication, grievance redressal portals, and regulatory APIs. A feedback loop cuts across all three layers, where AI tools analyse transaction patterns, detect anomalies and inform system improvement. This figure helps to visualise how CPS devices at the edge, programmable network logic, and trust mechanisms interact to create a continuous "chain of trust" from the user's device to the institutional back-end (Table 18.2).

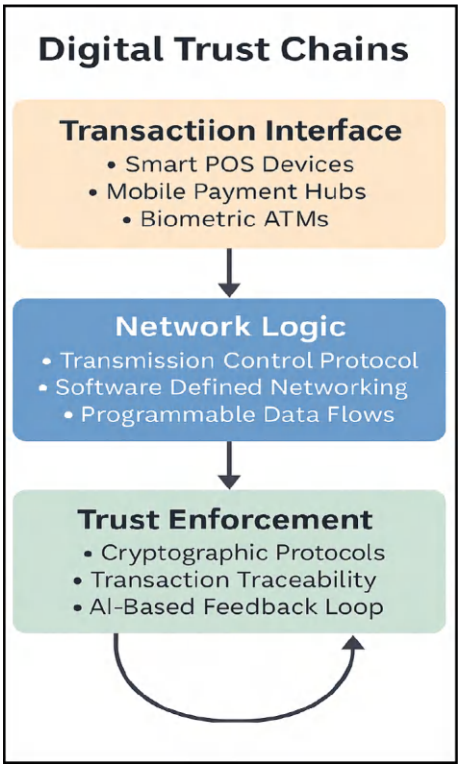


Fig. 18.1 Digital Trust Chains (DTC) framework: Functional flow view showing interaction among the Transaction Interface, Network Logic, and Trust Enforcement layers with an AI-enabled feedback loop

Table 18.2 Functional overview of the three layers in the DTC framework

DTC layer	Key functions	Technologies and design features
Transaction interface layer [8]	End-user interaction, local data capture, offline storing	CPS devices, NFC-enabled smart POS, biometric ATMs, QR kiosks, USSD/voice interfaces, multilingual UI
Network logic layer [7]	Routing, bandwidth management, microtransaction priority, link switching	SDN controllers, TCP/IP, edge computing nodes, GSM/Wi-fi/ fibre integration, encrypted APIs
Trust enforcement layer [10]	Authentication, logging, compliance, dispute resolution	MFA (biometrics, PIN, OTP), blockchain or secure logs, grievance portals, regulatory APIs, AI-based fraud detection

18.3.2 Global Case Studies: Integrating CPS, Programmable Networks, and Trust

Seven real-world examples show how cyber-physical devices, software-defined/networked logic, and trust mechanisms have been combined in inclusive finance deployments.

Case 1—Unified Payments Interface (India)

Launched by the National Payments Corporation of India in 2016, UPI enables instant interbank transfers via mobile applications and QR codes. Its CPS layer comprises smartphone apps and merchant QR placards, sometimes supported by Aadhaar-linked terminals; the network layer relies on secure APIs and a high-throughput NPCI back end. User confidence is reinforced through UPI PINs, immediate SMS notifications, and bank-grade security practices. By 2024 UPI was processing well over ten billion transactions each month, reflecting deep public-private uptake [21].

Case 2—M-Pesa (Kenya)

Introduced by Safaricom in 2007, M-Pesa extended basic financial services to many previously unbanked people through SIM-based wallets and a widespread agent network for cash-in/cash-out transactions [6]. Its CPS footprint centers on feature phones and point-of-sale–style agent terminals [8], while transaction routing is managed through telco infrastructure [16]. Trust is cultivated with PINs, SMS confirmations, and the local credibility of agents; today more than 85% of Kenyan adults use mobile money, highlighting how simple devices and social trust work together [6].

Case 3—PIX (Brazil)

PIX, rolled out by Brazil's central bank in 2020, offers round-the-clock instant payments using mobile apps and QR codes. The implementation pairs QR and biometric-capable apps at the edge with SDN-style processing and open APIs in the back end for real-time routing and multiservice integration. Two-factor authentication, strong regulatory oversight, and consumer protection rules underpin user trust. Within three years PIX reached roughly 70% adoption among the population [1].

Case 4—bKash (Bangladesh)

bKash began in 2011 under BRAC Bank to deliver mobile wallet services through a dense network of agents. Its CPS layer includes USSD-based wallets, smartphones, and basic terminals at agent kiosks, while transactions run over GSM and USSD channels. Trust is supported by agent relationships, SMS alerts, spending limits, and KYC screening. With more than 70 million users, bKash illustrates how local language interfaces and agent engagement can dramatically broaden financial inclusion [2].

Case 5—eNaira (Nigeria)

Launched as Nigeria's central bank digital currency in 2021, eNaira incorporates wallet apps, biometric-enabled ATMs, and QR readers at the CPS edge. The system depends on a centralised ledger with integrations to existing banking rails that resemble SDN-managed logic [7]. Identity device controls such as biometric checks are intended to build trust, but adoption has remained very low—under 1%—largely because of limited awareness, onboarding frictions, and public scepticism toward formal institutions.

Case 6—LinkAja (Indonesia)

Formed in 2019 as a state-backed platform, LinkAja combines mobile banking, e-wallet features, and QR payments. Edge devices range from QRIS-compliant POS units and NFC cards to smartphones; interoperability is achieved through telco–fintech APIs and SDN-supported routing in the back end. Trust measures include e-KYC and OTPs validated under Bank Indonesia's regulation. Although the platform counts over 80 million registered users, connectivity gaps and digital literacy shortfalls constrain deeper use in rural areas [22].

Case 7—Jan Dhan Yojana plus RuPay (India)

The Jan Dhan financial inclusion initiative (2014) opened hundreds of millions of zero-balance accounts linked to Aadhaar and RuPay debit cards. Its CPS components include biometric kiosks, micro-ATMs, and mobile banking apps, while low-cost routing is provided by the RuPay network. Trust is layered through Aadhaar authentication, OTPs, passbooks, and direct benefit transfers into verified accounts, and the programme substantially expanded savings access and the delivery of subsidies [19]. Table 18.3 synthesises how these cases map onto CPS, SDN/network logic, and trust mechanisms.

18.4 Mapping and Common Features

Across these deployments the same three functional layers recur: front-line transaction interfaces, programmable/network logic, and trust enforcement. Low-cost, durable CPS hardware—for example feature phones, micro-ATMs, QR placards, and agent kiosks—is essential for reaching low-income and rural populations. At the network layer, solutions mix GSM backbones, SDN-assisted routing, hybrid telco–bank integrations, and resilient APIs to enable real-time processing and interoperability. Trust mechanisms typically combine real-time alerts, biometric or KYC checks, audit trails, and multi-factor authentication.

Table 18.3 Global case studies mapping CPS–SDN–trust architecture in inclusive finance

Case	Background	Technological elements	Outcomes and insights	Relevance to DTC
UPI (India)	Real-time, app-based interbank payment system (since 2016)	CPS: Mobile apps, QR codes SDN/network: NPCI back-end, APIs Trust: PIN, SMS alerts	10+ billion monthly transactions; broad public–private adoption	Strong transaction Interface and trust enforcement integration
M-Pesa (Kenya)	SIM-based mobile money with agent networks (since 2007)	CPS: Basic phones, agent devices SDN/network: Telco routing Trust: PIN, local agents	High penetration among adults; deep rural outreach	Shows power of simple CPS and community trust
PIX (Brazil)	Instant, 24/7 payments with QR and mobile apps (since 2020)	CPS: QR scanners, apps SDN/network: SDN-backed real-time processing Trust: 2FA, regulation	Rapid adoption; high share of retail payments	Model for national SDN-based instant payments
bKash (Bangladesh)	Mobile wallet with extensive agent network (since 2011)	CPS: USSD wallets, agent kiosks SDN/network: GSM/USSD Trust: Agent-led, SMS alerts	Strong uptake in rural and peri-urban areas	Highlights vernacular interfaces and hybrid CPS/SDN
eNaira (Nigeria)	CBDC pilot for inclusion and digitalisation (since 2021)	CPS: Wallet apps, biometric ATMs SDN/network: Central ledger Trust: Biometrics, device ID	Low active usage; trust and literacy challenges	Emphasises need for layered trust and awareness

(continued)

Table 18.3 (continued)

Case	Background	Technological elements	Outcomes and insights	Relevance to DTC
LinkAja (Indonesia)	State-backed mobile wallet and QR payment platform (since 2019)	CPS: QR POS, NFC cards, apps SDN/network: Telco–fintech APIs Trust: e-KYC, OTP	Large user base but uneven rural adoption	Shows convergence of telco and fintech
Jan Dhan + RuPay (India)	National bank account and card scheme (since 2014)	CPS: Biometric kiosks, RuPay cards SDN/network: RuPay network Trust: Aadhaar + OTP + passbook	Over 500 million accounts; improved benefit transfers	Validates layered identity and offline verification

18.4.1 Synthesis

Three cross-cutting patterns emerge from the cases. First, inclusive payment systems consistently adopt a three-tier architecture that links user devices, network orchestration, and accountability mechanisms. Second, frugal CPS technologies are central to widest reach and usability. Third, effective public–private coordination—among governments, telecoms, banks, and fintechs—is crucial to align infrastructure, regulation, and service design. Together, these patterns support the DTC argument: CPS-driven economies achieve inclusive finance when simple, people-centred transaction interfaces operate on programmable, resilient networks with trust enforcement built into the system (Table 18.4).

These patterns reinforce the strategic logic of the DTC framework: inclusive finance in CPS-driven economies depends on frugal transaction interfaces, programmable and resilient network logic, and strong trust enforcement embedded in system design.

Table 18.4 Layer-wise common attributes across case studies

DTC layer	Common attributes across cases
Transaction interface	Biometric ATMs, mobile apps, USSD wallets, QR scanners, NFC-enabled POS, agent kiosks
Network logic	GSM backbone, SDN-assisted routing, hybrid telco–bank integration, fault-tolerant APIs
Trust enforcement	Real-time alerts, biometric/KYC checks, audit trails, multi-factor verification

18.5 Establishing the Digital Trust Chains (DTC) Framework

Section 18.5 translates the conceptual DTC model into an operational architecture, describing each layer in detail and showing how feedback loops support adaptation over time.

18.5.1 Transaction Interface Layer

The Transaction Interface Layer represents the points where users interact with the financial system. Its design priorities are accessibility, affordability, and localisation.

Key components

- **Smart POS devices**—NFC-enabled, possibly solar-powered, and capable of offline storage
- **Biometric ATMs and micro-ATMs**—Aadhaar-based or fingerprint-enabled identity verification
- **QR-code kiosks—**enabling low-value payments in local shops and markets
- **Voice/USSD interfaces**—supporting feature phone users and low-literacy customers through simple menus

Design principles

- **Frugality**—devices built with low-cost microcontrollers and open-source firmware
- **Multilingual UIs**—supporting regional languages and simple icons
- **Offline fallback**—store-and-forward logic when connectivity fails
- **Durability**—dust-proof and water-resistant enclosures suited to field conditions

18.5.2 Network Logic Layer

The Network Logic Layer manages the flow of data between user devices, payment switches, and banking servers. It uses SDN and traditional TCP/IP protocols.

Core functions

- Real-time routing and bandwidth optimisation
- Programmable control to prioritise microtransactions and critical welfare transfers
- Dynamic switching between GSM, Wi-Fi, and fibre networks
- Latency reduction for rural deployments through edge computing nodes

Technology stack

- SDN controllers integrated with financial API layers
- Modular plug-ins for interoperability with National rails (such as UPI, RuPay, or other instant payment schemes)
- End-to-end encryption and configurable firewall policies

18.5.3 Trust Enforcement Layer

The Trust Enforcement Layer ensures that each transaction is authenticated, logged, and compliant with relevant regulations.

Core technologies

- **Multi-factor authentication (MFA)**—combining biometrics, OTPs, device tokens, and PINs
- **Blockchain-based or secure audit trails**—recording transactions in a tamper-resistant format
- **API-level compliance monitoring**—linked with regulatory sandboxes and supervisory dashboards
- **AI-powered fraud detection engines—**monitoring patterns and flagging anomalies for human review.

Policy integration

- Alignment with National KYC and AML frameworks
- Support for open banking standards and data protection rules
- Scope for a Digital Trust Certification regime for vendors and platforms.

18.5.4 Feedback and Adaptation Mechanism

The DTC architecture includes a self-learning loop that uses AI and analytics to adapt over time.

Key features

- Tracking user behaviour to identify pain points and interface issues;
- Recognising fraud patterns and flagging suspicious transactions;
- Providing real-time policy dashboards to regulators;
- Enabling banks and platforms to offer tailored products such as microcredit or savings tools based on observed usage.

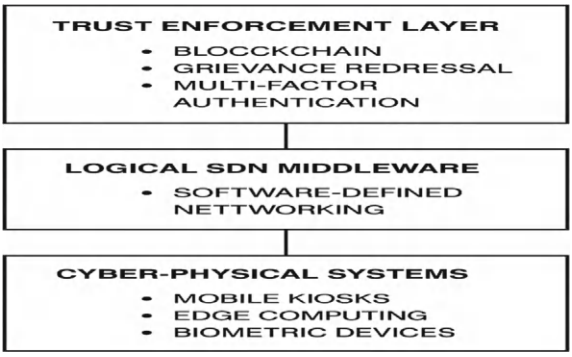


Fig. 18.2 Digital Trust Chains (DTC) system architecture: hierarchical view showing Trust Enforcement Layer, Logical SDN Middleware/ Network Logic Layer, and Cyber-Physical Systems with transaction flows and adaptive feedback.

In Fig. 18.2, user-initiated transactions travel from CPS devices through SDN-managed routes to core banking systems. At each stage, trust checks—authentication, logging, and rule-based screening—are applied. Data from these processes feeds back into analytics modules, which suggest rule updates, targeted awareness campaigns, or infrastructure upgrades.

18.5.5 Summary of Benefits across DTC Dimensions

See Table 18.5.

Table 18.5 Summary of benefits across DTC framework dimensions

Dimension	Benefits offered
Accessibility	Low-cost, robust devices with multilingual and offline-capable interfaces
Trust	Layered authentication, transparent audits, grievance mechanisms
Scalability	Modular APIs, SDN-based routing, edge computing, and cloud integration
Ethics and compliance	Integrated KYC/AML checks, regulatory dashboards, open standards
Frugality	Lightweight hardware, reuse of existing mobile networks, simple software stacks

18.6 Implementation Roadmap and Policy Recommendations

The DTC framework requires a phased, multi-stakeholder implementation strategy that balances technical ambition with socio-economic realities.

18.6.1 Five-Phase Implementation Plan

The five-phase roadmap progresses from local pilots to National-scale integration (Tables 18.6 and 18.7).

This sequencing helps avoid premature large-scale rollouts and ensures that trust, literacy, and infrastructure evolve together.

Table 18.6 Phased implementation roadmap for the DTC framework

Phase	Key actions	Lead stakeholders	Indicative timeframe
Phase 1: Pilot deployment	Select 3–5 rural/ semi-urban districts; deploy low-cost POS and micro-ATMs; conduct awareness campaigns on digital trust	Ministry of finance, state governments, local banks	Months 0–6 (year 1, Q1–Q2)
Phase 2: Infrastructure scaling	Establish SDN infrastructure; integrate biometric interfaces with existing banking APIs	NPCI or equivalent, UIDAI or identity authority, fintech start-ups, telecom operators	Months 6–12 (year 1, Q3–Q4)
Phase 3: Policy institutionalisation	Launch regulatory sandbox; create trust certification for CPS–SDN platforms	Central bank, data protection authorities, IT ministry	Months 12–18 (year 2, Q1–Q2)
Phase 4: Community uptake	Train SHGs, MSMEs, and frontline workers; deploy smart subsidies and digital credit tools	NGOs, NABARD-type institutions, skill development missions	Months 18–30 (year 2–3)
Phase 5: National rollout	Expand DTC to all subsidy, remittance, and key e-governance programmes	National digital governance division, banks, planning bodies	Month 30 onwards

Table 18.7 DTC implementation roadmap with risks, mitigation strategies, and core actions

Phase	Key risk	Mitigation strategy	Core actions	Lead stakeholders	Timeframe
Phase 1	Low uptake by intended users	Work with grassroots organisations and respected community figures to run outreach and awareness programmes	Select pilot districts; distribute necessary devices; conduct campaigns focused on building trust and basic digital skills	Ministry of Finance, state governments, local banks, NGOs	Months 0–6
Phase 2	Delays in infrastructure deployment	Secure vendor sign-offs on SDN designs in advance and roll out infrastructure in stages	Construct the SDN backbone; connect biometric and CPS hardware through well-defined APIs	NPCI, UIDAI, telecom operators, fintech firms	Months 6–12
Phase 3	Regulatory constraints or slow approvals	Engage early with the central bank and data authorities to clarify requirements	Open a regulatory sandbox for testing; establish trust certification standards and processes	Central bank, data regulators, Ministry of IT	Months 12–18
Phase 4	Digital illiteracy and user resistance	Offer incentivised training programmes and develop local digital champions	Train self-help groups and MSMEs; set up community support kiosks and helplines	NGOs, NABARD, local government bodies	Months 18–30
Phase 5	Problems integrating the DTC with existing systems	Adopt a staggered rollout and perform comprehensive testing in sandbox environments before full integration	Integrate DTC with subsidy, remittance and e-governance systems; monitor and stabilise integrations	National digital governance division, major banks	Month 30 onwards

18.6.2 Policy Recommendations

To support the DTC framework, several policy interventions are recommended.

(i) **Trust Certification Framework**: Governments should introduce a Digital Trust Certification scheme for CPS–SDN financial platforms. Certification criteria should cover security, inclusion, grievance handling, and data protection, aligning with SDG 16 on strong institutions.

(ii) **Interoperability Standards**: Regulators should promote unified APIs and biometric KYC protocols to avoid vendor lock-in and encourage competition. This supports SDG 9 by strengthening digital infrastructure and innovation.

(iii) **Digital Inclusion Subsidies**: Targeted subsidies or micro-leasing schemes can reduce the cost of devices and data for rural households, SHGs, and micro-enterprises. This contributes to SDG 9 and SDG 10 on reduced inequalities.
(iv) **Frugal Innovation Grants**: Funding programmes can support local development of CPS sensors, POS terminals, and edge devices suited to rural conditions. This aligns with SDG 8 on decent work and economic growth.
(v) **Ethics-by-Design Mandates**: Regulations should require transparent AI models, clear consent processes, and data minimisation in DTC deployments, connecting to SDG 12 on responsible consumption and production.
(vi) **Decentralised Governance**: Panchayats and local bodies should be empowered to monitor DTC usage, oversee local grievances, and act as intermediaries between citizens and higher-level regulators, supporting SDG 11 on sustainable communities.
(vii) **Monitoring and Evaluation (M&E)**: Real-time dashboards should track adoption, transaction integrity, error rates, and user satisfaction. These indicators should be linked to National SDG reporting, especially SDG 9 and SDG 10.

18.7 Conclusion, Challenges, and Future Research Directions

This final section is organised into five parts to clearly distinguish findings, policy implications, challenges and limitations, future research directions, and a brief summary statement, as recommended by the editors.

18.7.1 Key Findings

The chapter has advanced four main findings:

- **Layered architecture is essential**: Inclusive digital finance in CPS-driven economies works best when transaction interfaces, network logic, and trust enforcement are treated as distinct but interdependent layers.
- **Frugal CPS devices are central**: Low-cost, rugged, and offline-capable CPS devices can extend financial services into regions that conventional banking cannot reach.
- **Programmable networks underpin resilience**: SDN-enabled network logic allows microtransactions and critical welfare transfers to be prioritised, rerouted, and monitored even under infrastructure constraints.
- **Trust mechanisms must be embedded in design**: Multi-factor authentication, traceable logs, grievance channels, and regulatory visibility are not optional add-ons but core design features for inclusive finance.

These findings emerge from both the conceptual DTC model and the synthesis of global case studies.

18.7.2 Policy Implications for CPS-Driven Financial Systems

The DTC framework implies that policymakers should:

- View digital finance as a socio-technical system, not just a technology rollout;
- Invest in shared CPS infrastructure and interoperable APIs as public goods;
- Prioritise trust-centred regulation, including certification schemes and ethical AI guidelines;
- Coordinate efforts across ministries responsible for finance, IT, rural development, and social protection.
- National strategies that integrate DTC principles are likely to see more sustainable and equitable financial inclusion outcomes.

18.7.3 Challenges and Limitations in Implementing DTC

There are certain challenges and limitations which need to be considered:

- **Institutional capacity**: There may be a lack of institutional capacity among the regulatory authorities and public banks regarding the capability to handle the SDN-based infrastructure and feedback mechanisms through the use of artificial intelligence.
- **Funding limitations**: Capital expenditures are required for the use of CPS equipment, connectivity, as well as training. Even where benefits appear large in the long run, initial costs must
- **Digital and financial literacy**: Sometimes, users may resist or abuse digital tools if they are not properly educated and supported regarding digital and basic financial literacy.
- **Data protection and privacy**: The potential for misuse of data from the combination of biometrics, transaction data, and AI analysis is of concern.
- **Context dependence**: The successful implementation of DTC framework plasticity relies upon local politics, trust, and civil society. An effective system for one country can require radical transformation for another.

The awareness of these limitations is necessary to prevent overly optimistic projections and to develop a perspective for step-by-step progress.

18.7.4 Conclusion

The DTC framework presents a frugal, federated, and human-centred approach towards digital finance in CPS-driven economy. By integrating CPS devices, SDN-based network logic, and robust trust enforcement, DTC presents a model of a financially sophisticated yet socially inclusive and ethical digital finance system in a technologically advanced era.

18.7.5 Future Research Directions

The DTC framework opens several avenues for further research:

- **User behaviour and trust dynamics**: Empirical studies could examine how socio-cultural factors, gender, and local power relations shape trust in CPS-driven financial tools.
- **Explainable AI for trust scoring**: Researchers can develop transparent AI models for fraud detection and creditworthiness that align with fairness and inclusion goals.
- **Cross-sectoral applications**: The DTC architecture could be tested in healthcare, education, and local governance, for example in digital vouchers, scholarship disbursement, or land-record systems.
- **Energy and environmental impacts**: Studies are needed on the energy use and environmental footprint of CPS and blockchain deployments, especially under frugal hardware constraints.
- **Comparative policy analysis:** Cross-country comparisons could explore how regulatory styles and institutional histories influence DTC-like systems' adoption and resilience.

Glossary

CPS—Cyber-Physical Systems This term refers to machines or devices that react to their surroundings by using sensors, simple processors, and communication links. A CPS usually sits in the physical world but stays connected to a digital system so it can send data or receive commands.

SDN—Software-Defined Networking In an SDN setup, the thinking part of the network is kept separate from the cables and switches that move the data. Because of this split, engineers can adjust the behaviour of the network more easily, sometimes even without touching the hardware.

DTC—Digital Trust Chains This is the framework discussed in the chapter. It ties together user devices, network functions, and trust safeguards. The idea is to

build a chain that stays reliable from the moment a user starts a transaction until it reaches the back-end system.

UPI—Unified Payments Interface UPI is India's real-time system for small and large payments. People use mobile apps and a simple PIN to move money between bank accounts within seconds.

CBDC—Central Bank Digital Currency A CBDC is basically a digital form of National currency. Instead of being printed on paper, it is created and maintained by the central bank and stays in electronic form from start to finish.

API—Application Programming Interface An API can be thought of as a negotiated agreement between two software programs. It tells each side what information can be requested and how that information should be delivered.

Trust Enforcement Layer Within the DTC model, this layer handles all the steps that reassure the user—identity checks, transaction logs, and places where problems can be reported. It uses whatever mix of tools the system requires, from biometrics to PINs or even paper receipts in some settings.

Frugal Innovation It refers to finding workable solutions with whatever resources are available. The focus is on simple ideas that people can maintain easily, even in tough conditions.

Financial Inclusion This means helping people who are usually left out—like rural workers or low-income families—get access to essential financial services.

Multi-Factor Authentication (MFA) MFA asks for more than one proof of identity, such as a PIN, an OTP, or a fingerprint, so that accounts stay better protected.

References

1. Banco Central do Brasil: PIX statistics and reports (2023). Retrieved from https://www.bcb.gov.br/en/financialstability/pix
2. bKash: Annual report and impact statistics (2023). Retrieved from https://www.bkash.com
3. Deloitte: The future of trust in digital banking (2021). Retrieved from https://www2.deloitte.com
4. Deogirikar, J., Vidhate, A.: Security attacks in IoT: a survey. In: 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), pp. 32–37 (2017). https://doi.org/10.1109/I-SMAC.2017.8058176
5. Gambetta, D.: Trust: Making and Breaking Cooperative Relations. Basil Blackwell (1988)
6. GSMA: The state of mobile internet connectivity report (2023). Retrieved from https://www.gsma.com/mobilefordevelopment
7. IMF: Nigeria's eNaira: Innovation, Control, and Challenges (2022). IMF Country Reports. https://www.imf.org/en/Publications/CR
8. Islam, S., Roy, A., Kabir, M.: CPS-enabled banking platforms in low-connectivity areas. Journal of Emerging Technologies in Financial Services. **8**(2), 91–105 (2020)
9. ISO: *ISO 20022: financial services—universal financial industry message scheme. Int. Organ. Stand. (2013) https://www.iso.org/standard/60390.html
10. Kshetri, N.: 1: Blockchain's roles in meeting key supply chain management objectives. Int. J. Inf. Manag. **39**, 80–89 (2018). https://doi.org/10.1016/j.ijinfomgt.2017.12.005
11. Liu, Y., Zhang, W., Zhao, H.: Enhancing fintech resilience with CPS: a case study of smart POS networks. International Journal of Financial Innovation in Banking. **3**(4), 222–236 (2021)

12. OECD: Data governance and digital trust frameworks (2022). Retrieved from https://www.oecd.org/digital/trust
13. Radjou, N., Prabhu, J.: Frugal Innovation: how to Do More with Less. The Economist Books (2015)
14. Rogers, E.M.: Diffusion of Innovations, 5th edn. Free Press (2003)
15. Sen, A.: Development as Freedom. Oxford University Press (1999)
16. Singh, A., Kapoor, N., Thomas, J.: Software-defined networks in financial infrastructure: a performance review. International Journal of Network Security. **22**(3), 481–491 (2020)
17. UNCTAD: Digital economy report: recovery and beyond. United Nations Conference on Trade and Development (2022). https://unctad.org/publication/digital-economy-report-2022
18. WEF: Designing inclusive digital financial systems: a policy toolkit World Economic Forum (2021). https://www.weforum.org
19. World Bank: Global Findex Database 2023: Financial Inclusion, Digital Payments, and Resilience. (2023). doi:https://doi.org/10.1596/1813-9450-10596
20. Zhang, L., Chen, H.: IoT and biometric integration in CPS-based banking systems. Journal of Financial Cybernetics. **6**(1), 37–52 (2022)
21. National Payments Corporation of India (NPCI): UPI product statistics and system overview reports (2024). Retrieved from https://www.npci.org.in
22. Bank Indonesia: Annual report on payment system and digital services (2023). Retrieved from https://www.bi.go.id

Chapter 19
Decentralized Identity and Authentication Framework for Self-Sovereign Digital Portfolios in Web3-Enabled Cloud-Edge Networks



Kuldeep Pal, Lovish Jain, Aklavya, and Renu Mishra

Abstract Due to the transformative nature of blockchain technology and decentralized finance (DeFi) that can support secure, transparent, and absence of middle-men solutions for the management of digital assets and identities, there has been a radical change in terms of how these things are handled. This chapter proposes a Decentralized Portfolio Management System (DPMS) that has been developed using the Polygon Mumbai Test network that combines Solidity smart contracts, Web3.js, MetaMask, and IPFS to provide decentralized storage. Users can generate, edit, and recover features of verifiable portfolios as well as personal, educational, professional data based on the system, with all the information being permanently stored in the blockchain. The smart contracts will accomplish the authentication, the access control and the portfolio management, and the React.js frontend will guarantee the UI is usable and does not rely on blockchain understanding to interact successfully. The low transaction costs, the latency of 2–4 s, and high access control demonstrated by testing proved the feasibility and efficiency of the system. DPMS solves important issues of the traditional systems, providing elimination of intermediaries, cost reduction, and a possibility to provide a globally accessible, trustless portfolio solution, and the potential to be cross-chain and have decentralized identity verification.

Keywords Blockchain · Decentralized finance (DeFi) · Polygon · Smart contracts · IPFS · Web3 · Decentralized identity · React.Js · MetaMask

K. Pal · L. Jain · Aklavya · R. Mishra (✉)
Department of CSE, SSCSE, Sharda University, Greater Noida, India
e-mail: renutrivedi11@gmail.com; renu.mishra@sharda.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_19

19.1 Introduction

Decentralized finance (DeFi) and blockchain technology have become a revolutionary technology that is changing the way digital identities, financial documents, and career portfolios are organized in the current digital space. The conventional portfolio management systems engage centralized infrastructures in which banks, brokers, institutions of learning and third-party platforms are custodians of user data. Though this model has been popular for decades, it is affected by a number of inherent disadvantages such as high operation cost, inefficiencies due to manual verification, lack of control over personal information and also lack of transparency on information storage machanism [1]. Centralized architectures also bring about single points of failure where portfolio data is easily hacked, unauthorizedly modified, and system downtimes, which all compromise trust and reliability. With the growing mobility of the global workforce and the growing importance of remote verification, it has become even clearer that the constraints of centralized portfolio systems are more severe. The fast development of blockchain and DeFi offers an alternative possibility in the form of providing trustless, decentralized, and immutable record management without the intervention of intermediaries. Blockchain guarantees that the stored information is transparent, encrypted and verifiable permanently, without any chance of data being manipulated by an unauthorized party. Smart contracts have automated authentication, access control, and update of the portfolio to make sure that the same rules are applied and not due to human bias [2]. These features improve security and at the same time reduce the cost of operation greatly. Several works indicate that portfolio and credential systems based on blockchains will encourage transparency, enhance the integrity of data, and provide an efficient way of employers or educational establishments to validate achievements [3]. Moreover, decentralized systems are globally accessible, allowing the participation of users in the region with poorly developed digital infrastructure in professional ecosystems without involvement of the traditional centralized authorities. Owing to these benefits, this chapter presents the Decentralized Portfolio Management System (DPMS), a powerful Web3-based application that was designed to overcome the critical failures of traditional portfolio platforms. DPMS is developed based on Polygon Mumbai Testnet and incorporates Solidity smart contracts, Web3.js, MetaMask, and decentralized storage using IPFS. The system enables users to save, update and maintain their personal, academic, and professional information in a safe and tamper-resistant system where all change is cryptographically validated and permanently stored on the blockchain. DPMS allows users to have complete control over their information by decentralizing the identity management and ownership of data without depending on any centralized infrastructure of middlemen and servers [4]. The fact that blockchain is immutable and transparent also enhances confidence to the recruiters, institutions, and organizations that would like to establish the credentials of a user. Besides secure data management, DPMS is usable with a react-based frontend that hides the complex blockchain functions and has a clean and easy to use interface. This makes even non-technical users easily interact with the system and update their portfolios and accept

blockchain transactions with little learning curve. Furthermore, the low-cost and high-throughput transactions on the Polygon network coupled with the decentralization of file storage systems via IPFS, makes DPMS an effective and scalable solution that can be adopted in the long run. The chapter outlines the design, architecture, methodology, implementation, experimental results, and security considerations of the project plan of the proposed DPMS. It illustrates how decentralized technologies have the potential to transform the way portfolio management is done by providing a transparent and globally accessible and self-sovereign analogue to conventional centralized platforms.

19.2 Background Overview

The Decentralized Portfolio Management System (DPMS) is the combination of major innovation areas of technologies and financial innovations: blockchain networks, smart contracts, decentralized finance (DeFi), and the user empowerment of Web3. All these aspects will bring new paradigm in respect to trust, transparency, automation and being available globally. To complete the picture on the basis of DPMS, one has to consider how these new theories handle the shortcomings of the traditional portfolio management as well as open brand new horizons.

19.2.1 Traditional Portfolio Management: Flaws of Traditional Portfolio Management

Portfolio management is a part of traditional finance concerned with marrying and managing a combination of investments to maximize returns given the level of risk, a field informed in large part by Modern Portfolio Theory (MPT) co-developed by Harry Markowitz in 1952. The strategy puts a strong focus on diversification and mean-variance to obtain the maximum possible gain within a set risk kind.

Nonetheless, centralized portfolios systems have their disadvantages. They rely on third parties between themselves and the investors, including bankers, brokers, and asset managers which besides associating it with high fees also contributes to bias and restricted control over the investor. It is opaque in that operations involve the existence of closed ledgers that are not subjected to audit. Clearing and reconciliation procedures make settlements time- and resource-consuming as it can take hours to days of the settlement process. Also, with such systems, the investor can be exposed to counterparty risks such as fraud, mismanagement, and insolvency, and most of these services are not accessible to the individuals located in areas containing poorly developed banking infrastructure [5, 6]. Such constraints show the necessity of having a different model that is not only transparent, secure, and available globally.

19.2.2 DeFi as a Paradigm Shift

Decentralized Finance (DeFi) is a fundamentally different approach to the possibility of providing financial services since centralized intermediaries have been replaced by the intelligent contract-based protocols on the public blockchain. In DeFi, its users do not have to trust a sole authority because they directly connect with applications. This has a few theoretical benefits, including a trustless environment in which the rules are enforced by code, a universally accessible system anywhere in the world with no regional nuances, composability where protocols can be combined together as building blocks, and continuous non-stop access to the market 24 h a day, 7 days a week. DeFi allows in the context of portfolio management tokenized assets and algorithmic rebalancing and settlement in near real-time. This not only minimizes the expenses burden but additionally, provides the possibility of integration of innovative investment products, including the idea of automated index funds that are run fully based on smart contracts [7].

19.2.3 Blockchain as the Secure, Immutable Base

The idea behind DPMS is a blockchain technology; in a nutshell, it is a distributed ledger that is shared by an integrated set of independent nodes [8, 9]. It has several major design principles: decentralization (abolishing single points of failure), consensus mechanisms (requiring all nodes to be in agreement on the state of the ledger), and immutability (after being recorded, a transaction cannot be modified), and transparency (all data can be publicly verified). In the context of portfolio management, blockchain offers an anti-tampered contract of all the asset ownership as well as the history of transactions and the performance data. Whether they are recording an account of a project just added into the portfolio or updating the same, blockchain makes sure that any changes are irreversible and can be traced back [10].

19.2.4 Smart Contracts

Smart contracts refer to programmable code executed on a blockchain that have the ability to conduct a task automatically based on a given set of conditions. They eradicate the manual monitoring and third party permission. The smart contracts in DPMS coordinate all of the essential portfolio activity: data storage and decryption, ownership checks and updates, lockdown, and broadcast in response to changes. The determinism of smart contracts ensures that they will give the same results when the same input used and therefore gives predictable results. They cannot be changed or tampered with, in that portfolio logic can always be changed without redeployment

of a brand new contract, and maximize trust in the system. In terms of efficiency, automation through smart contracts makes the operational costs highly reduced and fast as opposed to a manual form of administration [11].

19.2.5 Web3: Ensuring Autonomy and Communication by Users

Web3 technologies reduce the distance between the users and the blockchain. Differing with Web2 applications whose control is in the hands of the centralized platforms, Web3 allows the user to interrelate directly with the blockchain-based protocols in the usage of tools such as MetaMask [12]. On DPMS, Web3 makes sure that each of the transactions, say creating a project or changing the educational information, is signed by the digital key of the user, which not only guarantees the security but also the authenticity. This model puts the users in control of their digital identity and portfolio data, thus, there is no more need to rely on centralized accounts or passwords. Web3.js integration will enable us to read or write blockchain data and make changes to the UI in real-time with a seamless transitional experience as most of the work will be on the frontend.

19.2.6 A Comparison of the Centralized and Decentralized Portfolio Systems

Centralized portfolio systems have their control in custodians that custody and manage assets of users making the model costly, have less transparency and are restricted to a particular area. The features of the decentralized systems, such as DPMS, are that the magnates have full authority over their portfolio, transaction data are transparent on the blockchain, fees are small because transactions are automated, and anyone can use them worldwide. Decentralized systems are more secure in nature since they cannot be tampered with and are not vulnerable to insider trust [13].

19.3 Proposed Methodology

19.3.1 Technology Stack Overview

Decentralized Portfolio Management System (DPMS) is built upon a well-considered technology stack that guarantees decentralization, security, and scalability in addition to a convenient use experience. Every tier of such a stack has its own purpose,

including processing of blockchain transactions, as well as creation of an interface that would be easy to use by users.

19.3.1.1 Layer 2—Polygon Mumbai Testnet

The backbone of Decentralized Portfolio Management System (DPMS) is the level of blockchain as it is based on the Polygon network. Polygon was chosen because it is Ethereum Virtual Machine (EVM) compatible, and its transaction fees are low, and the transaction-throughput is high, which is essential to a system where the timely updating of the data is necessary. In the development stage, Polygon Mumbai Testnet is useful to replicate the mainnet environment without real financial risk. This gives developers an opportunity to test and fine tune smart contracts in a safe environment. The blockchain also records all the portfolio data including project details, education records, file hashes etc. in a way that the data once written cannot be altered in any way, therefore preserving transparency, security and trust.

19.3.1.2 Smart Contract Layer Solidity and EVM

The logic of DPMS lies in smart contracts that are written using Solidity and deployed and run inside EVM. All the functions of portfolios such as adding and editing projects, making records of academic information, and storing the hash file of resumes in IPFS and data access are automated through these contracts. They apply stringent ownership policies such that only the wallet address that is attached to a portfolio may edit it. Smart contracts can also send events whenever certain information is altered which can be used to update webpages in the frontend. Use of Solidity means the system has access to an industry standard, secure, bespoke and well supported programming language to create decentralized applications.

19.3.1.3 Web3 Interaction Layer: Web3.Js

The Web3 interaction layer is the gap between the frontend application, and the blockchain. Introduced on the basis of Web3.js, it converts the actual moves of users which can be filling a form, updating a resume or otherwise, into blockchain-compatible transfers. It also helps connect wallets and sign transactions and get data out of the blockchain. This layer is also subscribed to the smart contract events making sure that the frontend reflects any changes upon confirmation on-chain via an event. It also handles network recognition, and users will be advised to migrate to the Polygon Testnet when logged in the incorrect network.

19.3.1.4 Frontend Presentation Layer-React.Js and Tailwind CSS

DPMS frontend is user-friendly, scalable and responsive. The interface made with the help of React.js is split into recordable parts like Projects, Education, Experience, and Resume. Such modularizing allows the application to be easy to maintain and expand. Styling is performed with Tailwind CSS, and gives us a clean and modern mobile-friendly styling. The frontend makes the raw blockchain data user friendly in the form of tables, cards, charts as well as being able to give immediate feedback by facilitating real time access and correction feedback measures.

19.3.1.5 Wallet Layer-MetaMask

A popular Web3 wallet, MetaMask, authenticates or signs transactions. DPMS is changing this by using the wallet address as a decentralized identifier as opposed to traditional login credentials. The application never discloses the location of a users keys, instead of encrypting their keys being held in a local store. All the actions on blockchain adding a project or uploading a resume are explicit actions; each of them requires a clear user confirmation, and no unauthorized changes are possible.

19.3.1.6 Decentralized File Storage Layer IPFS

DPMS also interfaces with the Interplanetary File System (IPFS) to store such large files as resumes and profile pictures. Instead of saving files on the blockchain (which would be expensive), the system puts them on IPFS, and only keeps a unique Content Identifier (CID) on the blockchain. This method guarantees immutability, i.e. any file change would result in a new CID, and decentralization, i.e. files would be available even in case one of the HA servers goes down.

19.3.1.7 Development and Deployment Tools—Hardhat

Compiling, testing, and deployment of smart contracts are supported with Hardhat. It offers some advanced debugging, automated test environment, and plugin support including gas optimization and security analysis. Due to the ability to deploy contracts to the Polygon Mumbai Testnet first, developers will be able to test functionality, performance, security, and are able to move to the mainnet later. This iterative process minimizes the probability of weakness and elevated reliability in the system.

19.3.2 System Architecture

The Decentralized Portfolio Management System (DPMS) architecture is designed to interconnect with user interaction, integrate blockchain connectivity, employ decentralized storage, and execute secure smart contracts in a friendly user interface (Fig. 19.1). In its design, it allows users to easily hook up their digital wallet, have full control over every position in their portfolio and have all the data associated with it permanently stored on the blockchain in a decentralized, tamper-proof, and transparent way. Respectively, a product of such a workflow begins with the device of a user and passes through the multiple steps of a well-organized process up to the situation when portfolio data is finally stored on the blockchain at Polygon.

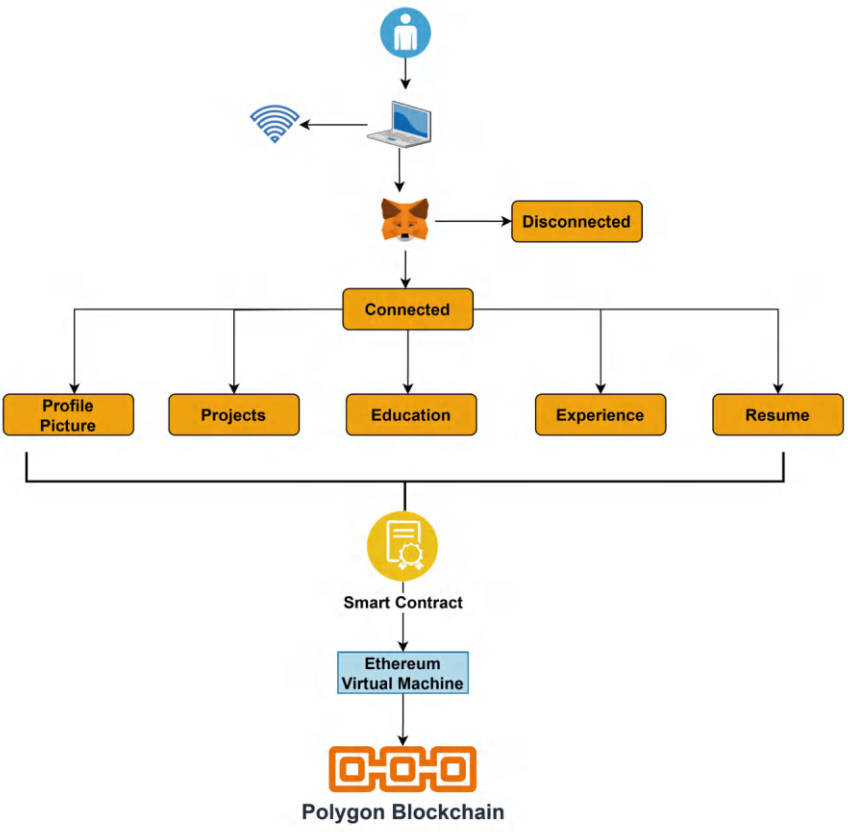


Fig. 19.1 Architecture of DPMS showing wallet connection, portfolio modules, smart contract execution, and data storage on the Polygon blockchain (Authors' own illustration)

19.3.2.1 Connecting to User and Devices

It starts by granting an access to the end user to log into the DPMS application using a stable internet connection via a laptop or a desktop. This connection will not only serve to load the application but it is also critical to the capability of the application to constantly interact with the blockchain. All those steps the user makes whether it is creating a project, or updating a resume involves the exchange of data with blockchain nodes. Intensive network stability promises the continuance of the transactions and speedy confirmation of their obtaining. When a decentralized system is used, and there is no central server to "retry" actions on the user, then a reliable connectivity must prevent against faulty operation.

19.3.2.2 MetaMask Connection to Wallet

When the application is loaded, a user should connect the cryptocurrency wallet using MetaMask or similar Web3 wallet. Such a step is decisive due to the following reasons:

- When the wallet is disengaged, the user is placed in read-only mode and is not able to take any blockchain-interactive processes such as the addition or the alteration of the portfolio information.
- In case of a connected wallet, the system identifies the user with his or her wallet address as his or her decentralized identity.

This wallet relationship avoids the use of a conventional username/password system to log in and thus signatures are used to authenticate the user moves.

This guarantees that only the rightful owner of the portfolio can be able to update thus ensuring high levels of security.

19.3.2.3 Portfolio Information Modules

Once the wallet has been successfully authenticated, the user gets access to the several portfolio management modules of the frontend interface. Every module is developed to work with particular forms of working information:

- *Profile Picture*: User can add an image which is safely placed in decentralized storage and attributed to their on-chain profile. Projects: This enables addition or editing of project information such as the title, description, technology applied and an optional link to the repositories.
- *Education*—Stores academic degrees together with name of schools attended, names of courses and years passed.
- *Experience*—Recounts work experience in the profession with information such as job name, name of the company, functions and role period.

- *Resume*—Uploads or links a full CV which is likely to be available on IPFS to make it immutable and accessible. Through these modules, managing various features of the portfolio is now easy and users no longer have to associate directly with blockchain commands.

19.3.2.4 Processing of Smart Contracts

Each time when a user updates or adds portfolio data, this information is sent to a smart contract on the Polygon blockchain first. The guts of the business logic which determine how data is stored, updated, and retrieved are in this contract. By constructing these rules into the contract, DPMS will make all operations within portfolio secure, verifiable and trustlessly executed i.e. have no central entity corrupting or verifying data manually. This smart contract also ensures that it checks whether the sender of the transaction is indeed the owner of the portfolio before implementing any entry in the process to curb unauthorized access.

19.3.2.5 Execution in Ethereum Virtual Machine (EVM)

The Ethereum Virtual machine (EVM) performs the functions of the smart contract logic to make sure that they are computed the same way on all the nodes on the network. This makes it reliable and there is no critical point of failure that may jeopardize the integrity of the system. This ensures reliability and there is no single critical failure point that can degrade the integrity of the system. As Polygon is EVM-compatible, developers can utilize Ethereum rich ecosystem and security and at the same time have the benefits of increased speed and lower gas prices offered by Polygon.

19.3.2.6 Polygon Blockchain Storage

After the transaction has been processed and validated by the EVM, the last record is written to the Polygon blockchain. Polygon is selected because it is low cost, has high throughput, so it can update the portfolio of users in real-time. Records here cannot be changed after they are recorded, which makes certain that the information concerning the portfolio is permanent, transparent and resistant to interference. Besides creating such confidence with the recruiters it also gives the user a track record of professional success that one can be held accountable to.

19.4 Smart Contract Design

Smart contracts within the Decentralized Portfolio Management System (DPMS) play the primary role in terms of organizing the safe algorithmic management of all processes associated with portfolios in a transparent way. Stored on the Polygon blockchain, and executed autonomously as self-executing programs, these rules can execute without manual intervention at higher levels, so the data is stored in an immutable manner, and the transactions are executed in the exact manner coded. Smart contracts reduce the necessity of centralized administrators and, therefore, provide the user with complete control of his or her professional data and ensure that everything can be proved on-chain. The main aims of incorporating smart contracts into DPMS are to automate the way a portfolio is dealt with, eliminate the reliance on centralized power structure, make sure that the data is not tampered with and the actions of all parties are transparent and verifiable through recruiters or third parties. They do everything to store project and education information on them to connect to resumes stored on decentralized networks such as IPFS. The most important characteristics of Smart Contracts in the DPMS:

19.4.1 Linking and Authentication of the User of the Sites

Blockchain wallet addresses uniquely recognise each of the users. When connected, any data on a portfolio will be forever linked to the address, and hence the ownership and traceability.

19.4.2 Portfolio Data Administration

Smart contracts hold and update important portfolio items like profile pictures, description of project, technology stack, education certificates, work experience and resumes linked to IPFS.

19.4.3 Data Verification

The agreement requires that the wallet holder is the only person who can update his or her portfolio. All transaction requests that are made by illegal sender are immediately disregarded.

19.4.4 Event Emission

When the data is inserted or changed, some events are emitted by the smart contract. The frontend receives these kinds of events so that it is able to update its interface without the need to refresh the entire page.

19.4.5 Read and Retrieval Functions

The public read-only capabilities enable the recruiter or verifier to see the details of the portfolios through the portfolio by simply entering the wallet address of the user without harming the security of the user.

19.5 Frontend Development

The interactive layer of the Decentralized Portfolio Management System (DPMS) is considered the frontend as it is how the users interact with the blockchain in an intuitive, easy way and without having to understand the technical complexity that exists behind it. It is the centre of visual control within the application having the purpose to convert all user actions into blockchain transactions that can be easily checked and secured despite the record being stored at distributed place. This gives the frontend more than being a presentation layer, it is also an active part of the decentralized portfolio management.

19.5.1 What Is the Frontend Good for?

DPMS frontend is aimed to fulfil a number of important goals:

- ***Accessibility of the user***: The interface is user friendly to users who have no experience with blockchain. Steps are simplified using clearly labelled buttons, forms and navigation menus that help them go through a complex task, like making a blockchain transaction, without overwhelming them.
- ***Data Visualization***: Although blockchain data is stored in highly secure places, the particular form of storage is not convenient to directly be read by a human. This raw chain data is then interpreted on the frontend and rendered into visual formatted layouts-tables, profile cards, and interactive charts giving users the pleasure of looking at and analysing their portfolio.

- ***Transaction Facilitation***: The frontend makes it easy to carry out blockchain transaction. Each step of the process, which would include form filling and MetaMask confirmations, has been reduced to the minimum, and yet, the whole process is controlled only by the user.
- ***Real-Time feedback***: The frontend listens to the blockchain events so that updates become immediate when transactions are confirmed. This prevents confusion due to delays and will keep the interface in sync with the latest state of the blockchain.

19.5.2 Technology Stack

DPMS frontend is Web3-ready and was developed using modern technology stack:

- ***React.js***: This has every component of the portfolio modular, reusable and easier to maintain (Projects, Education, Resume, etc., are all components).
- ***CSS Framework Tailored CSS:*** CSS Framework will be able to offer the CSS Responsiveness and Tailored CSS which will allow a responsible design and maximum responsiveness on desktop computers and smart phones.
- ***Web3.js On-chain and on-table Smart contract communication***: an interface that can be used to communicate between the frontend and the blockchain, this tool allows access to reading as well as writing to smart contracts directly in the browser.
- ***MetaMask SDK***: Allows authorization to a secure wallet and signature of transactions in private and secure mode.
- ***Visualization Libraries***: Chart.js or Recharts do not require additional explanation since these tools allow one to visualize the statistics about the portfolio and performance measures in a chart form.

19.5.3 Core Features of the Frontend

- ***Status display and Wallet Connection***: The system looks at whether there is a Web3-enabled wallet such as MetaMask when the app is loaded. When it is not, it triggers installation. After this, the address of the wallet and the network it is running on loads. In case of being in a wrong network, a warning admonishes the user to revert to the right network.
- ***Portfolio Management Dashboard***: This is the main page to see and change the information about the portfolio that is divided into sections like Profile Picture, Projects, Education, Experience, and Resume. Data is downloaded directly to the blockchain therefore it is up to date continuously.
- ***Form Data Entry***: There are specific forms on entering or editing the entries in each section of the portfolio. To give an example, the Projects section can consider such input as a title, description, tech stack, and repository link. Form validation

ensures that invalid or incomplete data does not get into the blockchain hence diminishing transaction failures.

- ***Transaction Confirmation interface***: When changes are to be submitted by a user, the frontend displays the transaction request through Web3.js to MetaMask. This has to be accepted or denied by the user in an explicit manner, which allows him or her to make an informed decision as well as avoid unexpected updates.
- ***Up-to-the-minute Data Refresh***: Once the confirmation has been done the smart contract triggers an event. The frontend which listens to this event is able to update what is on the screen immediately, making it interactive and lead to the development of trust.
- ***Detection and Notifications of Error***: In the event a transaction is aborted because of not having enough gas, bad data or network problems, descriptive error messages are shown. Confirmed actions are achieved through the confirmation messages promoting usability and reliability.

19.6 Web3 Integration

Web3 component of the Decentralized Portfolio Management System (DPMS) serves as the much-needed lever between the frontend of the application and Polygon blockchain allowing the communication with the deployed smart contracts to communicate easily. Without this layers, the application would constitute a static interface only capable of performing decentralized operations. Web3 enables the system to access information stored on blockchain, perform verified transactions, subscribe to events on contracts, and set specific settings within the blockchain network since all actions taken by a user will be safely verified and stored on-chain. The result of this integration is that the application is no longer a traditional portfolio but instead a full-fledged, decentralized application (dApp) that runs independent of centralized servers and middlemen.

The integration of Web3 is one of the most important features because the integration includes wallet connection, through which authentication and transaction signing use the tools such as MetaMask to access the interface. The wallet address of a user can be considered as a decentralized identity in DPMS, instead of using the common username-password system. Upon connecting his or her wallet, the application checks the connection and gets the wallet address with which it works on all the further blockchain operations. That is, only the person who owns the right to the portfolio-by means of his or her own private key-will be able to update or make new entries.

It is also vital to note the direct interaction with smart contracts. Web3 allows the frontend to instantiate a contract based on the ABI and address of that contract and hence call both read and write functions. As an example, read-only calls are made to retrieve an existing portfolio, and transaction-based calls to add a new project, add experience information, or associate a resume with an existing IPFS CID. Web3

verifies all transactions so that a request is passed to the connected wallet to get the clear approval of actions by the user.

Event listening is another very important feature of Web3. When some modification is done to the portfolio, an event like ProjectAdded or ResumeUpdated is logged on the smart contract. Web3 listens to these events on the frontend and they are visualized in the portfolio in real time without having to refresh the page. This allows the development of a responsive user interaction, characterized by the smooth and seamless performance, and it is done in the best interests of keeping all of the blockchain records sufficient and synchronous with the information presented.

Network detection and management is also present in Web3, wherein the user is guaranteed to be using the right blockchain network, in this case, Polygon Mumbai Testnet. In the case of the user being on another wrong network, Web3 can issue a notification asking him/her to switch networks and this can avoid failed transactions and the case of being on a wrong network. It is an important feature as it offers consistency in operations and omits the mistakes made by users.

Overall, the integration of Web3 into DPMS is the root behind the secure, transparent, and trustless interaction between said users and the blockchain. It also ensures that there is genuine association of the portfolio to the identity of its owner on the blockchain, transactions are only carried out with express permission and that the system is real time and fully decentralized. A lack of Web3 would eliminate the fundamental base benefit of the platform which is verifiable, user-controlled professional identity management.

19.7 Data Storage Via IPFS

Interplanetary File System (IPFS) is a peer-to-peer storage protocol that is decentralized and allows to make the web faster, safer, and more open. Using IPFS in Decentralized Portfolio Management System (DPMS), the Decentralized Portfolio Management System is significant to managing off-chain storage of files like via resumes, certificates, project photos, or profile pictures.

Although blockchains are ideal in storing small structured data with heavy security, they are inapplicable in large file storage because of monetary heavy transaction costs and low storage. This problem is addressed by the use of IPFS which enables files to be stored into a distributed network but their authenticity can be verified on the blockchain. In DPMS this will be through storing on the blockchain only the Content Identifier (CID) a unique hash produced by IPFS which is generated per file. The real file is always spread on numerous nodes of IPFS.

19.7.1 Role of Verifiers

The conventional centralized types of storage stores files in destined servers where they are referred by a geographical address (e.g. a URL). This develops a number of constraints:

- When the server goes down the file becomes unavailable.
- The user cannot restrict the storage provider to destroying the file or editing without their permit.
- Centralized servers are hackable resulting in a breach of data.

IPFS eliminates the problems via:

- ***Content-Addressed Storage***: Files are addressed by a hash produced by its contents so that a changed file has a changed address as well.
- ***Decentralization***: Files are distributed on several peer-to-peer nodes rather than just one server and therefore single points of failure no longer exist.
- ***Permanent References***: File hash serves as a permanent reference thus simple to determine whether a file is changed or not.

19.7.2 Working of IPFS in DPMS

Integration of IPFS to DPMS occurs in a rather uncomplicated and secure process:

- File Upload On file upload (e.g. resume), the frontend uploads the file to an IPFS node (self-hosted IPFS or other public IPFS APIs, such as Infura).
- Hash Generation (CID) IPFS performs the file and comes up with Content Identifier (CID) characterizing the precise content of the file.
- Storing the CID on-Chain-The CID is then stored on the blockchain via the DPMS smart contract associating it with the portfolio data of the user.
- File Retrieval-This means that anybody with the CID can get the file out of IPFS, no matter which node it is stored in. The information in the system is retrieved at the closest place of availability to enhance speed.
- Authenticity Verification Authenticity verification—In case of any modification of the file, the CID changes, immediately raising suspicions of tampering.

19.7.3 Fundamental Benefits of IPFS on DPMS

19.7.3.1 Cost Efficiency

Direct storage of large data on the blockchain is prohibitively expensive since every byte is stored in every node in the network and the nodes that validate the data are to be paid through gas charges. This cost is removed in IPFS where CID is that is

stored on-chain, but the file itself is off-chain on distributed nodes. This saves even huge storage expense and frequent update without any massive financial implication. As an example, publishing a 1 MB resume there and then to Ethereum might entail hundreds of dollars in gas fees, whereas simply storing the resume CID in-band on Ethereum using IPFS incurs under few cents.

19.7.3.2 Decentralization and Redundancy

When it comes to centralized storage, the data becomes unavailable in case one hosting server is turned off or goes offline. This has been addressed in IPFS, where files are split across the globe. Other nodes can still join in providing the file even when some nodes turn off. Such redundancy can guarantee 99.99 per cent uptime, and it is resistant to censorship, hacking, and the inadvertent removal of data. In the case of DPMS, this implies that the user portfolios will be available at all times but not of the status of the servers of the application.

19.7.3.3 File Verification Tamper-Proof

Any of the files put on IPFS is assigned a content-based hash. Should a single bit of the file be altered, the hash will have altered completely. This cannot be amended by any one in DPMS, as all the files in each portfolio (including resumes or certificates) will be detected. Recruiters or employers can easily verify the CID on blockchain with the one produced through the file-there is a guarantee that the file is original.

19.7.3.4 Scalability

The centralized storage may have limitations that include bandwidth constraints and server overloads as the number of users and files increase in DPMS. IPFS does not suffer scalability problems as more nodes imply more users in the network and a higher accessibility and performance when retrieving files. This enables DPMS to have a wider base of users without having issues of storage bottlenecks.

19.7.3.5 Cross-Application with Blockchain

IPFS is very well suitable with blockchain since the CID could be stored as a mere string in a smart contract. This allows DPMS to simply integrate blockchain verification with decentralized files, leveraging the two strengths of each technology, blockchain in particular to offer immutability and IPFS to store at scale.

Incorporating blockchain and IPFS, DPMS provides a secure, cost effective and tamper-proof storage of user portfolio, and offers a global accessibility. Blockchain securely stores metadata of the portfolio so that unauthorized manipulations cannot

be applied, and IPFS contributes to the storage of large files in a duplicative way. Such a mixed model not only improves the level of trust and transparency but also provides sustainability of the system in the long run.

19.8 Result

Decentralized Portfolio Management System (DPMS) was designed and tested on Polygon Mumbai Testnet and its performance measured with respect to functionality, efficiency of transactions, file storage behaviour, experience of user interaction and reliability of the system. The testing stage was used to identify whether the system has met its objectives of decentralization, transparency, immutability and management of portfolio by users. The cohesive elements, such as Solidity smart contracts, the React.js frontend, and interacting with the blockchain via Web3.js, combined with the authentication procedure in the form of the MetaMask and IPFS storage, helped provide a fully decentralized and verifiable portfolio system.

The system always fulfilled the initial goals in the functional testing:

- ***Wallet Authentication***: Authenticated cryptographically by the use of a Metamask to avoid the use of usernames and passwords (Fig. 19.2).
- ***Portfolio creation and Updates***: Users could include or make updates to their personal, educational and professional information, all updates were preserved as immutable records on the blockchain.
- ***Decentralized File Storage***: Resume and related documents were placed on IPFS, and produced a unique cellar content hash to original verification.
- ***Event-Driven Updates***: Smart contracts would broadcast changes in data and this would enter the browser in a real-time way triggering the frontend to update without the need to reload the page.
- ***Access Control Enforcement***: Only the owner of the wallet could alter his or her portfolio and not just anyone. With testing, following were suggested by the performance measures.

Polygon Mumbai was tested by performance testing, and the results indicated efficient processing of transactions. An average of 2–4 s was achieved in the finality of transactions, even when the network was moderately busy, which indicates that Polygon is appropriate in terms of providing real-time updates on portfolios. Gas usage was recorded in various processes, such as project addition, resuming and education entries. The contract architecture was optimized, and the average transaction fee was only a fraction of a cent on the test network. A sample transaction in the Layer 2 chain with a fee of 0.000063 MATIC is summarized in Table 19.1; this shows that this form of decentralized portfolio storage is cost-effective.

The following Table summarizes the transaction cost recorded in the operation of DPMS on Polygon Mumbai Testnet. An average update procedure took 63,000 units of gas, and the fee was a tiny amount of 0.000063 MATIC. This is a testament that the proposed DPMS is still viable with respect to economic constraints despite

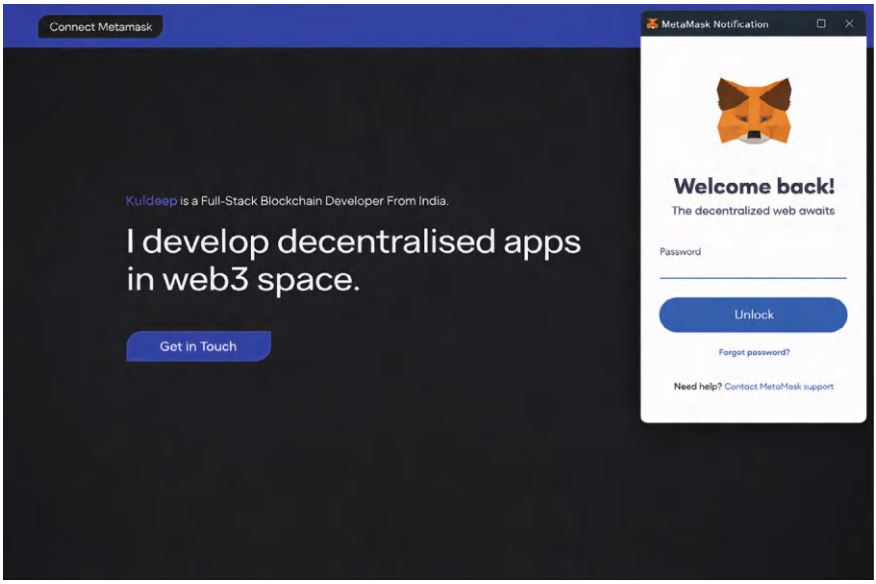


Fig. 19.2 MetaMask authentication prompt during wallet connection (Authors' own screenshot)

Table 19.1 Polygon transaction parameters and fees

Parameters	Value
Transaction value	0.5 MATIC ($0.00)
Transaction fee	0.00006300000063 MATIC $0.00
Gas price	3.00000003 Gwei (0.00000000300000003 MATIC)

the existing high activity of a portfolio. It was compared to traditional centralized portfolio systems to have considerable benefits. Folding out the need of platform administrators, DPMS will prevent the possibility of making obscure data edits and will make sure that everything can be verified on the blockchain. The DPMS is resistant to disruption due to outages, server crashes, or corruption of databases, unlike centralized systems, which can suffer all these effects. Global accessibility is also enabled by the system—a user can view their portfolio regardless of their location provided they have an Internet connection and a Web3-enabled browser. Figure 19.3 demonstrates the correspondingly linked wallet interface, which is what the end user would see when they successfully authenticate with the system.

On the whole, the findings illustrate that DPMS addresses its main goals of decentralization, trustlessness, immutability, transparency, global accessibility, and cost efficiency. The effective merging of blockchain and IPFS provides an illustration of the feasibility of the decentralized portfolio systems and their capability to substitute or supplement the traditional portfolio management systems.

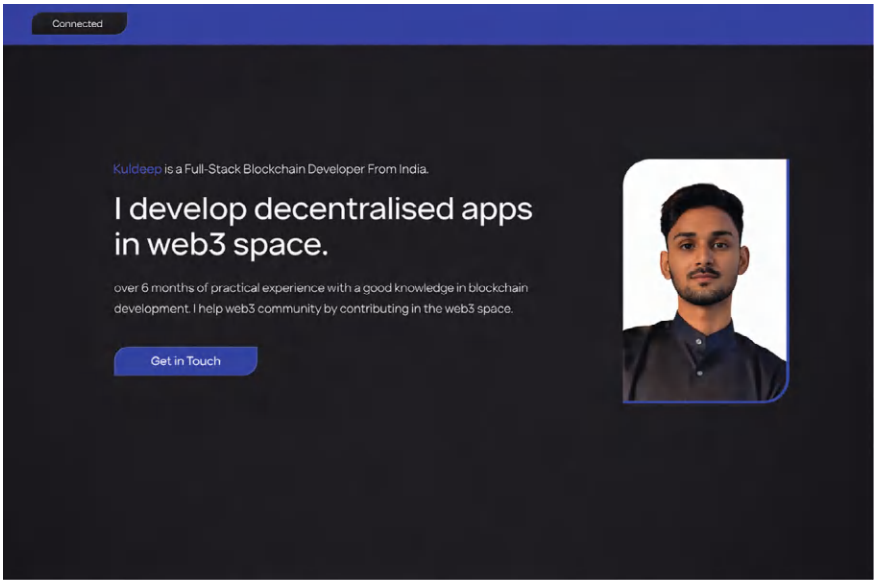


Fig. 19.3 Interface connected wallet view and user portfolio in DPMS (Authors' own screenshot)

19.9 Security Considerations

Security is never added after the fact in any decentralized application (DApp), it is how you build trust. The Decentralized Portfolio Management System (DPMS) is used in a scenario that is lacks reversibility of transactions, sole possession of private keys by users, and malicious parties persistently find vulnerabilities. Thus sound security measures should be implemented on all levels: smart contracts, frontend, backend integrations, blockchain communication, wallet handling and user awareness.

One security hole may lead to alteration of information, loss of money or identity theft. This part briefly specifies the security considerations of DPMS clearly thus making it secure as much as possible both against technical as well as social engineering attacks.

19.9.1 Security in Smart Contracts

A key part of DPMS is smart contracts which are also irreversible upon deployment. This implies that a little error in coding will result in irreparable damage. As an example, by enabling overwriting data without verifying who owns them (msg.sender), an attacker will be able to corrupt the data of another user in a scenario where the update option is enabled. Thus, the contract should undergo the process

of an excellent audit with manual checks and automated services such as MythX or Slither. One should use access control through a modifier such as onlyOwner or role-based (thus, only authorized accounts could make a particular change). One should also gas-optimize the code in order to avert gas griefing attacks, a situation where an assailant makes transactions fail by increasing the cost of gas. Having an emergency back up such as a Pausable contract so that in an event of an exploit, you can pause the system temporarily to limit damages.

19.9.2 Wallet Security

The wallet is the most sensitive property of DPMS user since it contains the personal keys that manage portfolios owning. In case of theft of such keys, an attacker can control any action on the blockchain. DPMS makes it one step further avoiding storage of any kind of private key; rather than the transaction signatures being relayed remotely, the transaction signing itself is done locally within a wallet application such as MetaMask. The users should also not be exposed to phishing sites that pretend to be DPMS to deceive connection of wallets, and this issue is avoided by the use of SSL encryption, verified domains and user training. Furthermore, each transaction prompt in MetaMask contains essential information (contract address, gas prices, a name of a function), therefore, one can decline foul play.

19.9.3 Frontend Security

Although backend logic is secured by blockchain, it is vulnerable to such attacks as Cross-Site Scripting (XSS) or injection of malicious code. To solve this, DPMS overcomes this by sanitizing and validating all the inputs by the user and then processing it. As an example, when a user may input project descriptions then any script tag is removed to put a stop to the running of any script on other users browsers. Frontend-to-blockchain or IPFS communication everywhere is over HTTPS, which does not suffer the Man-in-the-Middle (MITM) attack. A Content Security Policy (CSP) can further restrict to which locations scripts can be loaded, narrowing the attack surface.

19.9.4 Node and Network Security

Proposed system uses RPC endpoints to signal the blockchain. In case they are not taken care of or are not trusted, the attackers can inject invalid data into the frontend. That is why DPMS is plugged into trusted RPC providers such as Infura, Alchemy, or QuickNode that have powerful uptime guarantees. There ought to be a fallback too such that the application transitions to other RPC providers automatically in case

the first available option is not available. Any communication between RPC uses TLS encryption to avoid interception.

19.9.5 IPFS Data Safety

Even though IPFS is decentralized, it can still happen that malicious actors can serve modified data in case users request data with unreliable addresses. DPMS avoids that by verifying the Content Identifier (CID) on the blockchain with the file it fetched. When they are not similar file is rejected. To be able to keep files accessible forever, the platform may opt to employ pinning services such as Pinata which keep duplicate files permanently. Also, the content moderation checks make sure illegal material or other damaging content is not stored.

19.10 Use Case Demonstration

The Use Case Demonstration is an outline of the way Decentralized Portfolio Management System (DPMS) is utilized. In this section, we shall go through a common user flow to demonstrate the importance of blockchain, IPFS, and the Web3 integration that will provide security, transparency, and decentralization.

19.10.1 Logging in the Application

The user goes to the DPMS platform and gets to a bare interface in which the user performs the authentication through a Connect Wallet button. There is no username-password system here but transactions are authenticated using wallets, whereby the user has authentication of his or her identity and its ownership does not rely on a central body.

19.10.2 Connecting the Wallet

In case of clicking the green “Connect Wallet” button, MetaMask asks the user to authorize the connection. After this, the wallet address comes up upon approval. The platform also verifies whether user is not on right blockchain network and notifies to change the blockchain. The direct portfolios connection guarantees that portfolio alterations could only be made by the appropriate owner.

19.10.3 Development of the Portfolio

The user may now include:

- **Documents and profile pictures** (saved on IPFS, where CIDs are saved on chain).
- **Projects, education and work experiences** (stored as structured data in the smart contract).
- **Resume PDF** (and on IPFS with CID verification)**.**

The hybrid is cost efficient since it removes the chain storage of large files but retains authenticity.

19.10.4 Transaction of Smart Contract

The frontend invokes the smart contract each time the user saves a section which creates a call to the smart contract function (e.g., addProject() or updateResume()) each sending appropriate data. The transaction information, such as the contract address, the function, and gas price, is displayed in MetaMask. Upon approval, the transaction is local signed, pumped to the blockchain, validated by ownership check (msg.sender) and etched forever. This assures that it is executed without centralisation, in a trustless manner that is bounded by rules.

19.10.5 Blockchain Confirmation/UI Update

After the contract is mined, a project-adding event is released such as ProjectAdded. These events trigger the frontend which listens to them and immediately responsive the UI. This on-chain feedback of the changes is therefore recorded in real-time and securely.

19.10.6 Verifications of Portfolio Publicly

A portfolio can be shared by the user. When recruiters go to the link, they will view up to date blockchain data as well as files hosted in IPFS with its CID verified. Provided that a document has been hacked, it will not have its matching CID, which immediately becomes evident of the alteration.

19.11 Challenges and Limitations

- ***Gas Fees on Mainnet*** On Mainnet, although the cost of transactions will be reduced compared to transmitting on Ethereum, the level of execution may vary during congestion, a factor that may cause one to avoid occasional revisions.
- ***Dependency on IPFS Availability*** Even because IPFS is decentralized, when the file is not pinned, the content retrieval can be sluggish or even unreliable, which could be demoralising to the experience of the end users.
- ***User Onboarding Barriers*** wallet set up, network changes/switching, transaction approval can be confusing to the non-technical user and this form of barrier can result in adoption issues.
- ***Smart Contract Immutability*** The introduction of smart contracts introduces issues with immutable code, which means bugs cannot be fixed directly and upgrades would need to be setup as complicated migration strategies.
- ***Scalability Concerns*** The time it takes to query blockchain/IPFS can become longer as the number of users and stored transactions scales and this aspect of the application can become slow.

19.12 Conclusion

The Decentralized Portfolio Management System (DPMS) will be a transition between the traditional approach of centralized, professional portfolio platforms and self-sovereign, blockchain-based one. Ensuring the full ownership of the user data, DPMS is immutable, transparent, and makes the use of Web3 possible due to combination of smart contracts, IPFS-based storage and Web3 integration. In contrast to centralized solutions which require the presence of one entity upon which to base identity creation, modification and sharing, DPMS enables the user to directly regulate the generation, alteration, and such sharing of his/her credentials without the need of intermediaries.

With this, the system tackles such important concerns as data tampering, confirmation delays and global centralization downtime, providing a secure and verifiable platform that can be accessed worldwide. As the decentralized technologies gain wider and wider adoption, DPMS will continue to develop further, integrating AI analytics, multi-chain integration, decentralized identity systems, and even integrating with metaverses and career networking.

In short, DPMS will be more than a portfolio manager it will provide a basis to develop a trustless, internationally comprehended digital professional identity. With the current trends in development of blockchain and Web3, solutions such as DPMS are potentially to transform the way professionals present their credentials and how companies can authenticate them, developing a clear, permeable and decentralized recruitment future.

19.13 Future Scope

The Decentralized Portfolio Management System (DPMS) will be a major step to self-sovereign digital identification and trustless verification of professional data. Another thing is that the possibilities of this platform are much more than it is possible now. New features, enhanced scalability, and user experiences can be provided as blockchain, decentralized storage, and Web3 ecosystems achieve maturity and DPMS can use them in their framework. In this part, the future opportunities and avenues of further development of this system are discussed in the context of the system functionality, security and adoption.

19.13.1 Multi-Chain Compatibility

At present, DPMS is collocated on the Polygon network, although in the future it will have the possibility to operate with several other blockchains such as Ethereum, Binance Smart Chain or Avalanche. It will give users a chance to choose the most favourable network, minimize overload problems, and guarantee that the data is available even in case one of the chains is down. The portfolio data on various networks could be synchronized via crosschain bridges to have broader scope.

19.13.2 Decentralized Integration Identities

TPMS could participate in an international identity scheme with an ability to incorporate Decentralized Identifiers (DIDs), Verifiable Credentials (VCs). This would enable the recruiters to immediately validate degrees, certifications and achievements which quavers manually. It would be even more adopted with compatibility with other systems, such as Polygon ID and ENS.

19.13.3 DAO-Based Governance

DAO A Decentralized Autonomous Organization (DAO) would allow users to vote on upgrades, pricing models and new functionality. This would make the development of this platform be transparent and user-driven which is in line with the ideology of user ownership that Web3 promotes.

19.13.4 AI-Portfolio Analysis

Through the use of artificial intelligence, blockchain-stored portfolios could be analysed to find in-demand skills, to offer resume suggestions, and enable recruiter engagement analytics. This would transform portfolios into not only a record of work, but a savvy career development tool.

19.13.5 NFT Credentialing

Future implementations of DPMS may make it possible to tokenize verified accomplishments as NFTs and incorporate metadata such as course description, grades and date of issuance. These credentials would be individual, immutable and easily provable on-chain, thus document forgery would be close to impossible.

19.13.6 Skill-Endorsement on-Chain

Portfolios and positions could include the addition of a social proof layer by sharing peer-verified endorsements through blockchain. This makes it so that the skills will not be based on a mere declaration of adherence to the self-professions but are supported with professional networks, to a greater extent trusted by the recruiter.

References

1. Leung, M.F., Wang, J., Li, D.: Decentralized robust portfolio optimization based on cooperative-competitive multiagent systems. IEEE Trans. Cybern. **52**(12), 12785–12794 (2022). https://doi.org/10.1109/TCYB.2021.3088884
2. Sujatha, E., Kumar, G.N., Vikas, N.V., Reddy, M.Y.: Decentralized portfolio tracking application on a polygon layer 2 blockchain. In: Proceedings of the 1st IEEE International Conference on Networking and Communication (ICNWC 2023), pp. 1–5 (2023). https://doi.org/10.1109/ICNWC57852.2023.10127419
3. Cai, T., Wu, J., Yu, C., Brusic, V.: Blockchain with machine learning for financial portfolio management. In: Proceedings of the 2023 IEEE 13th International Conference on Electronics, Information, and Emergency Communication (ICEIEC), pp. 255–259 (2023). https://doi.org/10.1109/ICEIEC58029.2023.10201043
4. Sahane, P., Obla, A., Bhangale, P.: Student achievements portfolio management using blockchain. In: Proceedings of the 6th International Conference on Inventive Computation Technologies (ICICT 2023), pp. 1787–1792 (2023). https://doi.org/10.1109/ICICT57646.2023.10134076
5. Yan, J.-Y., Hsieh, Y.-H., Yuan, S.-M.: Blockchain-based e-portfolio ecosystem. In: 2023 IEEE 6th International Conference on Knowledge Innovation and Invention, pp. 264–269 (2023). https://doi.org/10.1109/ICKII58656.2023.10332664

6. Merlec, M.M.: A blockchain-based trustworthy and secure review system for decentralized e-portfolio platforms. In: Proceedings of the 2023 14th International Conference on Information and Communication Technology Convergence (ICTC), pp. 675–680 (2023). https://doi.org/10.1109/ICTC58733.2023.10393600
7. Ranganthan, V.P., Dantu, R., Paul, A., Mears, P., Morozov, K.: A decentralized marketplace application on the Ethereum blockchain. In: Proceedings of the 4th IEEE International Conference on Collaboration and Internet Computing (CIC), pp. 90–97 (2018). https://doi.org/10.1109/CIC.2018.00023
8. De Vos, M., Olsthoorn, M., Pouwelse, J.: DevID: blockchain-based portfolios for software developers. In: Proceedings of the 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON), pp. 158–163 (2019). https://doi.org/10.1109/DAPPCON.2019.00030
9. Alslman, Y.: Exchanging digital documents using blockchain technology. In: 2021 International Conference on Electrical, Communication, and Computer Engineering, pp. 1–6 (2021). https://doi.org/10.1109/ICECCE52056.2021.9514253
10. Tsolakis, A.C., et al.: A secured and trusted demand response system based on blockchain technologies. In: 2018 IEEE International Conference on Innovative Intelligent Systems and Applications (INISTA), pp. 6–11 (2018). https://doi.org/10.1109/INISTA.2018.8466303
11. Krasniqi, B., Stefanova, E.P.: An evaluation of Web3 concepts and technologies and relation with digital forensics. In: 2023 International Conference on Automation and Informatics, pp. 514–518 (2023). https://doi.org/10.1109/ICAI58806.2023.10339005
12. Oleshchuk, V.: Enforcement of Web3 security by use of blockchain and LLMs. In: 2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, vol. 1, pp. 391–394 (2023). https://doi.org/10.1109/IDAACS58523.2023.10348890
13. Iddum, T.S., Sahu, S., Sujatha, R., Kumar, C.V.R.: Ethereum blockchain-based logistics application for smart supply chain management. In: 2022 IEEE Global Conference on Computing, Power and Communication Technologies, pp. 1–7 (2022). https://doi.org/10.1109/GlobConPT57482.2022.9938171

Chapter 20
Analysis of Attacks in IoT Network Using Contiki-Cooja Simulator



Bhawna Sharma, Shrey Gupta, and Rahul Johari

Abstract The developing interest in loT is empowering the enormous scope of Routing Protocol for Low Power and Lossy Networks (RPL) due to which it is encountering an influx of safety dangers. It is important to determine and analyze a wide range of attacks that target the RPL protocol. The proposed work focuses on implementing and analyzing two well-known routing attacks on RPL protocol called Blackhole Attack and Flooding Attack. Contiki-Cooja Simulator is used to implement these attacks and study their effect on the network by comparing the reference simulation with the attack simulation. The sensor map of the blackhole attack simulation shows that the nodes, in the range of the malicious node, get disconnected from the network. For the flooding attack, the graph of the average power consumption indicates a significant increase in the average power consumption of the malicious node as well as other nodes in its range. The attackers can easily steal important information during these attacks and exploit it to deal some serious damage to the owner of that data. To prevent the stealing of data during these attacks, two cryptography techniques are introduced that will encrypt the data with the help of keys. The techniques discussed in the paper are RSA Caesar Hill and RSA Rail Vigenere and they are also implemented using the Contiki-Cooja Simulator.

Keywords Internet of Things (IoT) · Contiki-Cooja simulator · RPL Protocol · Security attacks · Cryptography techniques · Encryption

B. Sharma · S. Gupta · R. Johari (✉)
SWINGER: Security, Wireless, IoT Network, Group of Engineering and Research Lab, University School of Information, Communication and Technology (USICT), Guru Gobind Singh Indraprastha University, Sector-16'C, Dwarka 110078, Delhi, India
e-mail: rahul@ipu.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_20

20.1 Introduction

The term "Internet of Things" (IoT) describes the networked connectivity of items that, with the help of established frameworks, are equipped with omnipresent intelligence. This new technology is the basis for all intelligent devices such as smart homes, smart cities, smart healthcare, etc. IoT can alternatively be thought of as the cornerstone for building a cyberworld in which intelligent services are provided by the seamless integration of physical devices into information networks. Connecting various items, with sensors, adds a degree of advanced insight and empowers them to impart constant information without manual intervention. Given the advances in hidden technologies, IoT is opening colossal opportunities for copious novel applications that guarantee to upgrade the norm of our lives.

20.1.1 Characteristic Features of IoT

IoT is equipped with many features that are listed as follows.

- **Connectivity**: Connectivity refers to establishing a proper connection between all the devices and IoT platforms which may be a Fog or Cloud server. One can attach different protocols of the internet network to amplify productivity and establish connectivity across the IoT environment and Industries.
- **Intelligence and Identity**: IoT devices have intelligent interfaces which allow them to communicate with the users. Intelligence helps the system to adapt to the changes which are brought upon by certain natural as well as synthetic factors. The Internet-connected devices are uniquely identified by their IP addresses, which allow them to be recognized across the network.
- **Dynamic and Self-Adapting**: A camera, meant for surveillance, should be flexible enough to work under different light situations. IoT devices should be designed keeping in mind the diverse scenarios to which they have to adapt.
- **Safety**: In an IoT ecosystem, sensitive information is passed within the system with the help of connectivity components. So, securing this sensitive information is one of the biggest priorities. While planning an IoT framework, one needs to adhere to the appropriate safety measures and firewalls to keep the data far from abuse and manipulation.

20.1.2 Pros and Cons of IoT

IoT has its advantages and disadvantage as well. These are listed as follows.

- Data monitoring has become easier with the help of IoT. It can provide data that could not have previously even been possible to collect.
- IoT has proven to be self-sufficient to a certain extent by doing several tasks without human intervention. Interaction between devices helps in doing tasks while minimizing human effort.
- IoT can process and deliver information accurately and in a timely manner with minimal energy wastage.
- Easy information access is one of the biggest advantages of IoT. Required data can be accessed from any location over the internet.

The disadvanatges associated with IoT are as below.

- IoT gives an easier way to access information but with increased security threats. Since data is more exposed, hackers may steal the data by prying into the system.
- IoT is capable of handling tasks with ease but many complex operations are beyond its capacity. Even a single wrong calculation may affect the whole process.
- The Internet is the backbone of the IoT. An issue with the internet may halt the entire IoT system.

20.1.3 Applications of IoT

Some of the IoT applications are listed as follows.

- **Smart Homes**: IoT home automation is the ability to coordinate domestic appliances by electronically controlled, web-associated frameworks. It characterizes a home in which various appliances such as air conditioners, TVs, fridges, cameras etc. are ready for communication with each other. This framework is comprised of switches and sensors associated with a focal center point which is then constrained by occupants utilizing a wall-mounted terminal or mobile phones associated with the web as depicted in Fig. 20.1.
- **Smart Offices**: IoT helps in establishing an agreeable and safe workspace that urges individuals to team up to enhance productivity. Smart gadgets like PC vision-empowered cameras, sensors, etc. can be utilized to monitor workplaces all across continuously. Additionally, it improves meeting planning and schedule visibility by making conference rooms smart.
- **Healthcare:** IoT-enabled devices have made distant observing inside the medical care area possible. Likewise, it assists the doctors with conveying the best consideration. It has expanded patient engagement with specialists with simpler and more proficient frameworks. Distant observations of patient's well-being helps in reducing the length of clinic stay. IoT has additionally improved conceivable treatment at low expenses. Several new approaches are being proposed to increase the efficiency involvement of IoT in the healthcare sector [1].

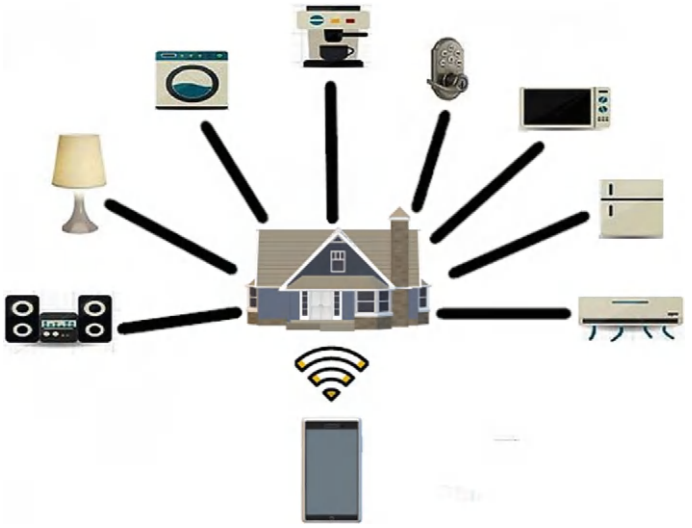


Fig. 20.1 Smart home

- **Hotels:** The objective of IoT lodging is to provide visitors with more prominent authority over the introduced gadgets. At the point when guests stay in a savvy room, they have appropriate alternatives for controlling different settings for gadgets like lights, indoor regulators, entryway locks, and that's only the tip of the iceberg. Mechanized smart entryway locks are one noteworthy security feature with an IoT lodging.
- **Robotics and Automation**: The IoT and advanced mechanic networks have converged to the Internet of Robotic Things (IoRT). The IoRT is a field where savvy gadgets can screen the occasions occurring in their environmental factors by intertwining their sensor information. They can utilize local and distributed intelligence to decide on courses of action, and afterward, control objects in the actual world.
- **Physically Challenged Persons**: The IoT can open creative opportunities for people with disabilities through the help of connected technology. It has helped doctors check physically challenged patients while they are at home. Further, the introduction of self-driving cars has given an incredible amount of mobility to the challenged.
- **Architecture and Real estate**: The expansion of the IoT in architecture and real estate everlastingly changes how we collaborate with the structures where we reside and work. With the help of IoT, agents can securely show properties without being physically onsite. It helps in making the buildings eco-friendly and provides user-friendly environments.

  There are many more IoT applications and the list is ever exhaustive.

20.1.4 Vulnerabilities of IoT

IoT devices make up almost one-third of all the network-connected endpoints making multiple companies the main targets for cyberattacks. The vulnerabilities in IoT can lead to severe protection issues at different levels in an IoT infrastructure [2] and provide easy access to private networks and steal private information. Numerous vulnerabilities can be exploited in IoT devices and if not fixed timely can lead to severe cyberattacks.

- **Weak or Guessable Passwords:** Many IoT devices do not permit users to modify the original passwords leaving them prone to some easy password attacks. Mirai Malware can be considered a good example that exploits this weakness. In September 2016, the Mirai-infected IoT devices were used to launch a DDoS (Distributed Denial-of-Service) attack on some major servers bringing down pieces of the Amazon Web services and its customers including Netflix, Twitter, and GitHub.
- **Use of Insecure or Outdated Components:** Utilising obsolete software or accessing unsafe libraries in a program endangers the item's general security. Whatever brings imperfections into the gadget can be used as a passage point or utilized to proceed with an assault.
- **Lack of Physical Hardening:** Solidifying the gadget against physical attacks ensures it against hostile clients endeavoring to gather delicate data that would then be able to be utilized to dispatch a remote hack or take control of the gadget. For instance, Debug ports that are not eliminated can leave the gadgets effectively open to hackers.

20.2 Attacks in IoT

Though IoT is a prospering space that has made life a lot simpler and easier, yet because of its expanding demand, the amount of security attacks has expanded. These attacks have made trusting IoT devices a challenging issue due to the ever-changing environment [3]. Some of the fundamental security dangers in IoT are described in this section.

- **Masquerading Attack**: In this type of attack, the attacker uses a fake identity and IP spoofing to get inside an unauthorized network and steal sensitive data. An organization, with feeble validation measures, is generally helpless against such attacks. By stealing a password or discovering a way around the authentication process, the attacker can access the important documents of an organization that could prove to be deleterious.
- **DoS Attack**: A denial of service (DoS) attack is a kind of attack where the attacker uses network traffic disruption to try and prevent the victim from using the resource or service for the intended purpose. This results in the system being compromised.

The ultimate effect, of this attack on the device, is its reduced usability. DoS attack generally targets the limitations of a device e.g. memory and exploits them by diverting it to work on needless tasks.

- **Blackhole Attack**: This is a sort of isolation attack which focuses straight on the network topology. The node, that plays out the attack, drops all of the packets that it should advance. This may result in the separation of certain nodes from the network. The nodes that use the malicious node as a medium for sending the packets are affected as the packets they forward are dropped by the attacking node.
- **Wormhole Attack**: A wormhole attack is a type of internal security attack where the malicious nodes align themselves at key spots in the network. One of the attackers snatches the packets and sends them to another attacker with the help of a tunnel. This may result in either early or deferred arrival. Detecting a wormhole attack is extremely difficult as they do not change the network directly. So, the transmission between the users needs to be interrupted to identify the attack.
- **Flooding Attack**: This is a type of direct attack where the malicious nodes generate a high volume of traffic to bring down the network. In a flooding attack, the original network traffic is obstructed because of the excessive flooding of packets. The attacking node sends a copious amount of extra packets like 'hello' packets to a non-existing destination which results in a significant increase in traffic.
- **Brute Force Attack**: This attack aims to find the targeted password by a trial and error method. The attacker uses software that tries all possible combinations for a password or a PIN to obtain personal information. Short passwords are easier to guess since there are fewer possible combinations. This is one of the reasons to select long passwords consisting of letters, numbers, and symbols. These types of attacks are generally targeted on devices used in houses.
- **Sinkhole Attack**: A sinkhole attack is possibly the most perilous attack as it compromises all the neighboring nodes by giving them fake information e.g. link quality and shortest path. In this attack, the malicious node advertises forged routing information to attract other nodes which results in sending the packets through the attacking node. The data is then tampered with by the attacker causing the network communication to collapse.

20.3 RPL Protocol

It is a tree-based proactive routing protocol that creates directed acyclic graphs among the nodes to permit information trade. RPL with one destination-oriented(DO) directed acyclic graph(DAG) is shown in Fig. 20.2. RPL protocol works on the IEEE 802.15.4 standard with the help of the 6LoWPAN adaptation layer and is based on the concepts of Distance Routing Protocol and Source Routing Protocol. RPL effectively and productively oversees information steering for nodes that have circumscribed resources. It gives a functional structure that guarantees bidirectional availability, strength, dependability, adaptability, and versatility.

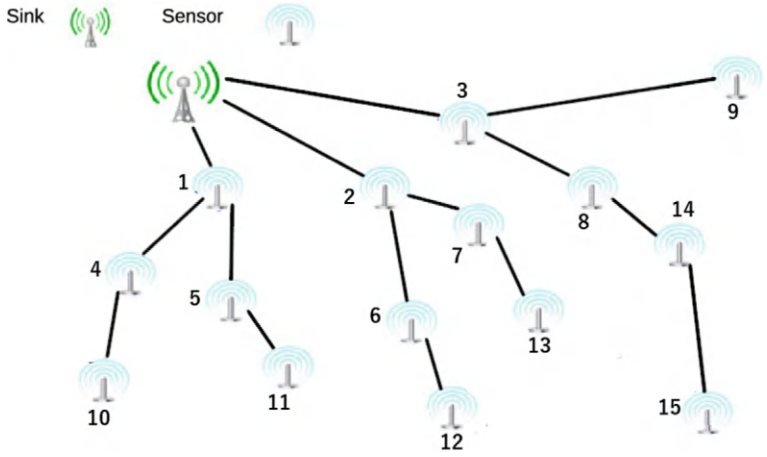


Fig. 20.2 RPL directed acyclic graph

20.4 Cryptography Techniques

Due to several security threats on the network, there is a need for techniques that can minimize the damage caused by the effects of the attacks particularly the stealing of data. One such technique is cryptography whose aim is to secure information and communication with the help of codes. Cryptography is a technique that encrypts the data with the help of a secret key which is provided only to the recipient by the sender so that the data can only be decrypted by the rightful receiver. Cryptography works on four basic principles that are given below.

- **Data Confidentiality**: The data sent through the network can only be accessed by selected individuals or systems. Any unauthorized access will be denied to keep the data secure from theft.
- **Data Integrity**: The data can only be edited or modified by the selected few who have permission. This also gives the assurance that the data was not changed in between the network by any unauthorized person which means the data sent by the sender was received without any changes.
- **Authentication**: In cryptography techniques, a unique key is used to encrypt the data, this key ensures that the data was received by the correct user. Only the authorized receiver would be able to decrypt the data as the key will be given to them.
- **Non-Repudiation**: The user with the secret key cannot refute the ownership of the data. If a contract is signed between two parties then neither of the parties can back out from the contract or deny their signature.

20.4.1 Types of Cryptography Techniques

Cryptography techniques are mainly divided into three major groups that are given below.

- **Symmetric Cryptography**: In these types of algorithms a single key is used for both encryption as well as decryption. The algorithm uses a key in a cipher to encrypt the plaintext and convert it to ciphertext. The same key can then be used by the receiver to decrypt the ciphertext and convert it back into plaintext. The key used should be kept private as with the help of the key even the unauthorized personnel can decrypt the data.
- **Asymmetric Cryptography**: This method uses two different keys with no connection to each other for encryption and decryption. The plaintext is encrypted to ciphertext by the sender using a public key that can be shared with everyone. The receiver uses a private key that has no relation to the public key to decrypt the data, this private key cannot be shared with anyone and only belongs to authorized personnel. The keys should be chosen such that there is no mathematical relation between them to keep the private key safe.
- **Hash Functions**: This algorithm only encrypts the data without being able to decrypt it back to its original state. It is a one-way function that secures the data by encrypting it without an option for decrypting the data. Every input given to the algorithm produces a unique output so hit and trial is the only method to figure out the input from the encrypted data. Trying different inputs until the same output is given by hash is the only way to decrypt it.

20.5 The Problem Statement

Internet of Things involves various steps in the transfer of data. Several intermediate nodes play an important role in the transfer of data between the source and the destination node. Due to the complexity of the network involved, there is a high possibility of the system being attacked by the intruders. These security attacks target the intermediate nodes that results in the data disruption of the network. The proposed work features two of the primary security attacks the Blackhole attack and the Flooding attack on the RPL protocol, which is perhaps the most utilized protocol in IoT. We use the Contiki-Cooja Simulator for the implementation of the attacks. Due to these attacks, the data becomes highly vulnerable to theft and the attackers can also misuse the stolen information. So to prevent misuse of data the work presents two types of RSA algorithm that encrypts the data namely RSA Caesar Hill and RSA Rail Vigenere.

20.6 Literature Survey

Various works, available in the literature, have been summarized in this section. In [4] author(s) elaborated various security and privacy issues faced by the IoT. Security issues for different IoT applications such as industries, healthcare, logistics, and self-driven vehicles are discussed. Some major privacy issue in IoT which includes access of data, eavesdropping, stalking, and data stealing are mentioned. [5] talks about the attacks on IoT and the security precautions to be adopted during the development phase of an application such as checking node identity. In[6] author(s) introduce the U2IoT architecture for Future IoT and proposes a dimension model based on the same architecture to classify the distinctive IoT advancements. The proposed work discusses the incorporation of IoT in the industrial sector and educational field. In [7] author(s) briefly compares the basic IoT protocols used for communication between the IoT devices and lists their vulnerabilities. It can be in the feared that the key can be easily determined by the attackers in the LoRaWan protocol but the 6LoWPAN needs advanced user identity authentication. The Zigbee network, while lacking security issues, uses low transmission energy whereas BLE links can be easily broken. So the encryption and decryption of each layer must be secured. In [8] author(s) survey the different IDS strategies that are carried out for the IoT networks. It gives an overall prologue to IoT and IDS accompanied by the classification of numerous IDS placement and analysis strategies in IoT architecture.

In [9] author(s) introduces an operating system, called Contiki, for the Internet of Things. This OS uses a Cooja network simulator to make the process easier. Cooja provides developers with a platform on which they can run and test their applications. The paper covers all the steps involved in using the Cooja network simulator such as how to start and save a simulation. In [10], the author(s) assess and examine the IETF RPL routing protocol's performance under the Contiki OS using the Cooja simulator. The performance of the network development process is examined by measuring several input metrics, and afterward, the performance and potential circumstances during the working phase of the WSN are investigated. It concludes that RPL is a robust enough protocol for WSN yet there are few additional metrics such as latency, and signaling overhead can be improved. In [11] author(s) gives us a basic understanding of the routing protocol RPL that befits the IoT applications. It explains RPL by including the DODAG formation and elucidates the Contiki OS and its simulator that is preferred for implementing RPL.In [12], author(s) survey the effect of safety instruments on the versatility of WSANs utilizing simulations and factual experiments. It is shown that security against eavesdropping and forgery has a moderate impact while protection against replay increases network formation time by a significant amount. The paper also proposes a standard-compliant optimization to defend against replay.

In [13], the author(s) elucidates the routing challenges faced in Low Power and Lossy Networks (LLNs) such as high packet loss ratio, limited resources, network lifetime, and high energy depletion. RPL saves network energy and uses the available resources more efficiently. In [14] author(s) states the benefits of using Homomorphic

encryption (HE) for data encryption in the IoT ecosystem. The messages can be restored using the private key. The used simulation model creates the outcomes that show an improvement in FNR, FPR, PDR, and DR significantly. In [15], the author(s) investigate IoT security utilizing 6LoWPAN execution with the essential spotlight on the IDS approaches to secure the network operation. The expected dangers as well as the better security approaches, cryptography, and IDS, are contemplated and thus concluded that joining the two methodologies can give security from internal and external attackers.

According to the author(s) of [16], CoAP provides strategies for resource management that are similar to those of HTTP while also providing additional functionalities like multicast, asynchronous communication, and memberships for IoT and M2M applications. Simulated CoAP offers RESTful engineering for low-power and lossy networks. It is likewise suggested that CoAP utilizes UDP as the transfer protocol and has a compact packet overhead. Thus, it fundamentally influences the energy utilization as well as the reaction time of the motes. Therefore, it is reasoned that CoAP is better contrasted with HTTP as far as energy utilization and reaction time are concerned. Author(s) offer an implementation of a full Contiki simulation system for the Shimmer platform in [17]. The presented Contiki model apps can be used with any host application that supports the ShimmerConnect protocol and can be used directly as trade-ins for the conventional Shimmer firmware. Custom algorithms can be employed to enhance the models for signal processing and classification, as they are written in plain C. As a result, developing and testing on-node pre-processing algorithms for remote sensor nodes may be done more quickly and easily. Tools to quickly create WSN applications and assess their effectiveness are suggested in [18]. The methodology proposed combines the advantages of Contiki OS and SysML language. First, a use case is displayed with SysML considering the static diagrams, dynamic diagrams, and requirement diagrams. Thereafter, a virtual simulation is completed by executing the application on Contiki OS and simulating the WSN with Cooja. This method renders the modeling task easy by reducing the coding time, plan time, maintenance time and accordingly, decreasing the relative cost.

In [19] author(s) conclude that a network topology does not fundamentally affect the energy utilization of the sensor node. However, a difference can be found in the energy usage when the transmission ratio is changed. Additionally, it is found that a low transmission ratio profoundly impacts the sensor node performance in the network. The calculation of each sensor node expands as it gets harder to decide the best course of action by the sensor node. This suggests that for a higher calculation at the sensor node, more energy is needed by every sensor node. In [20], author(s) propose a smart construction architecture incorporating IoT to manage the performance of the system. By utilizing Constrained Application Protocol (CoAP) in smart architecture, the energy consumption is reduced by 30.86%. To further improve security, the article notes how to use developments from the Certificate Authority (CA) to synchronize the Secure Hash Algorithm (SHA-256) with the Datagram Transport Layer Protection (DTLS). According to [21], Unmanned Aerial Vehicles (UAVs) have high mobility that leads to changes in topology. Frequent topology change may result in loss of packets or routing path failure that plays a significant role in the transmission

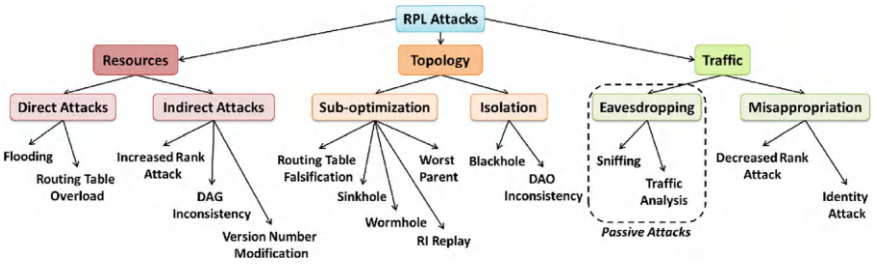


Fig. 20.3 Security attacks on RPL protocol

of data between the source and the destination node. To overcome the issue of high mobility, the proposed work introduces a secure energy-efficient dynamic routing protocol (SEEDRP). The authors use a network simulator to compare SEEDRP with existing protocols in FANETs.

20.7 The Proposed Methodology

The RPL protocol is considered one of the most used algorithms due to which it is subjected to a substantial assortment of security attacks. The primary factors that make low-power Lossy Networks vulnerable to assaults are their features, which include resource constraints, a lack of infrastructure, restricted physical security, erratic connectivity, and changing topology. The taxonomy of attacks on RPL protocol is depicted in Fig. 20.3. In this work, we focus on two attacks; the Blackhole Attack and the Flooding Attack.

- **Blackhole Attack**: It is a type of isolation attack which directly targets the network topology. The attacked node drops every packet it is meant to forward in a blackhole attack. This can be considered very deleterious in cases when the attacked is located at an important position that connects other nodes to the network, as it will isolate those nodes from the network. The result of the blackhole attack can be seen in Fig. 20.4.
- **Flooding Attack**: It is a direct attack which results in unnecessary processing by the nodes to exhaust their resources. Flooding attacks aim to produce a copious measure of traffic in a network which increases the average power consumption and makes links unavailable. To perform flooding attacks in RPL networks, an attacker can send unicast DIS (DODAG Information Solicitation) messages to each node, which must reply with a DIO (DODAG Information Object) message, as depicted in Fig. 20.5.

In a blackhole attack, the malicious node drops all the received packets due to which the data that is sent to the malicious node is lost and can be accessed by the attacker. In the case of a flooding attack, the resources of a network are exhausted that results in the unavailability of links due to which the original packets are not

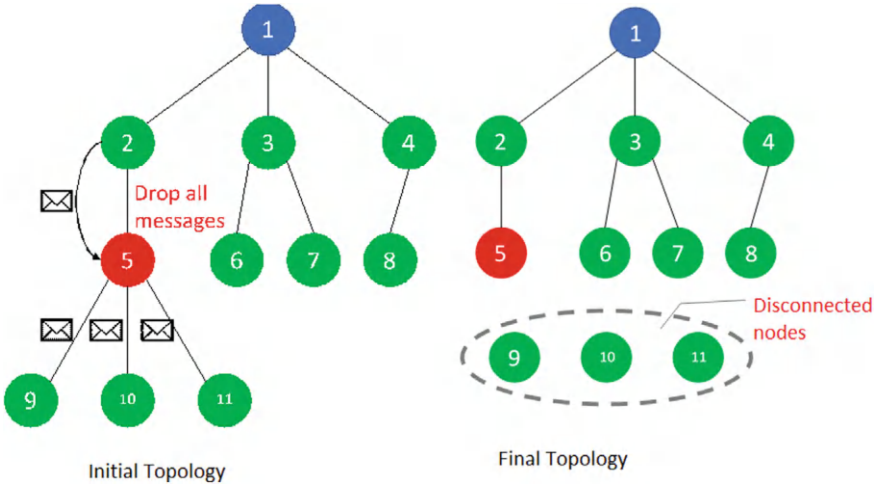


Fig. 20.4 Blackhole attack topology

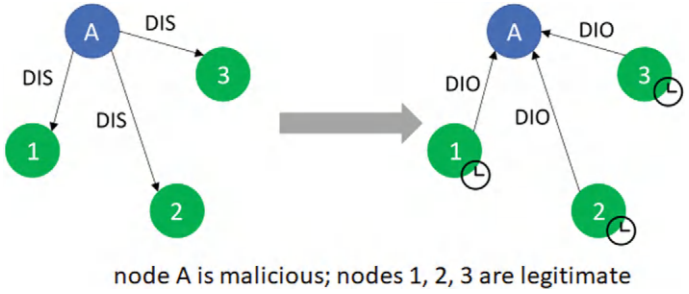


Fig. 20.5 Flooding attack example

transferred. Due to the unavailability of links for original packets, it becomes easier for the attacker to access those messages by providing them a fake link. To prevent the attacker from accessing the information, a cryptography technique can be used to encrypt the data. Only the receiver with the secret key would be able to decode the encrypted data. This would reduce the chances of the data being misused by the attacker. So even if the attacker performs a blackhole attack or a flooding attack on an intermediate node and data is stolen, it will not be accessed by the attacker as it will be encrypted. In this work, we present two cryptography techniques that are variations of the RSA algorithm namely RSA Caesar Hill and RSA Rail Vigenere.

- **RSA Caesar Hill**: To make the data more secure we have introduced a new variant of RSA algorithm by combining it with Caesar cipher and Hill cipher. First, the original text is converted to its equivalent numbers like A to 1, B to 2, etc. Then it is encrypted using RSA algorithm in which we use two keys that are public key and private key. After encrypting with RSA algorithm, we use a Caesar key to encrypt the text by Caesar cipher technique. Lastly, the text is provided with the

Hill key that is used to encrypt the resulting text from the RSA and Caesar cipher by the Hill cipher technique. The final ciphertext is obtained after the plaintext is encrypted by RSA algorithm and the two ciphers.

- **RSA Rail Vigenere**: For making the data hard to crack, RSA algorithm is combined with two cipher techniques namely rail fence cipher and vigenere cipher. Primarily the original text is converted to its equivalent numbers like A to 1, B to 2, etc. Then it is encrypted using RSA algorithm in which we use two keys that are public key and private key. For further encryption, a rail key is provided to encrypt the data using the rail fence method. This method gives the rail ciphertext which plays the role of plaintext for vigenere cipher method. The rail ciphertext is further encrypted by vigenere cipher method using a vigenere key. The final ciphertext is a combination of encryption done by RSA algorithm, rail fence and vigenere cipher.

20.7.1 Algorithm

Notations:
$\mathbf{N_n}$—Number of Nodes in Network
$\mathbf{S_n}$—Source Node in Network
$\mathbf{D_n}$—Destination Node in Network
$\mathbf{I_n}$—Intermediate Node(s) in Network
$\mathbf{M_n}$—Message in a Network
$\mathbf{SM_n}$—Sum total of all message generated by $\mathbf{S_n}$ for $\mathbf{D_n}$
$\mathbf{DM_n}$—Sum total of all message received by $\mathbf{D_n}$
$\mathbf{AD_n[.]}$—Array to receive message(s) at $\mathbf{D_n}$ in Network
$\mathbf{AI_n[.]}$—Array to hold message(s) at $\mathbf{I_n}$ in Network
$\mathbf{m_n}$—Message in a Network
$\mathbf{T_p}$—Total number of Message in Network
$\mathbf{C_n}$—Compromised Node(s) in Network
$\mathbf{AC_n[.]}$—Array to hold message(s) at $\mathbf{C_n}$ in Network
$\mathbf{M_n}$—Malicious Node in Network
$\mathbf{AM_n[.]}$—Array to hold message(s) at $\mathbf{M_n}$ in Network
$\mathbf{PB_a}$—Probability of occurrence of BlackHole Attack in Network
$\mathbf{PF_a}$—Probability of occurrence of Flodding Attack in Network
$\mathbf{M_{ttl}}$—Message Time to Live

Algorithm 1 Generic Algorithm for Attack Detection

Trigger: Message Notification at D_n
Check IsConnectionUp[D_n (I_n)]
Initiate m_n (D_n, I_n)
for (i=0; i< AI_n.length; i++)
$AD_n[.] \leftarrow AI_n[.]$
if (DM_n (AD_n [.]) < SM_n)
Print ("Loss of Messages !!);
Is M_n Loss due to M_{ttl} = 0
Print ("Message Timed Out !!");
Print ("Prompt S_n to increase M_{ttl} !!");
else
Print ("Probability of Attack in Network !!);
Print ("Detect: Whether PB_a or PF_a");
if (PB_a > Threshold Value)
Trigger ("Algorithm for BlackHole Attack Detection");
else
Trigger ("Algorithm for Flooding Attack Detection");
endif
elseif (DM_n (AD_n [.]) == SM_n)
Print ("All Messages Received!!);
else
Print ("Connection down between Intermediate Node and Destination Node");
Print ("No Message Received !!");
endif
return

Algorithm 2 Algorithm for Blackhole Attack Detection

/* n can be any number depending on the requirement of the simulation*/
For D_n, n = 1
For C_n, n = 4, 6, 8, 9
For M_n, n = 12
Trigger:
Initiate m_4 (M_{12}, C_4)
for (i=0; i< AC_4.length; i++)
$AM_{12}[.] \leftarrow AC_4[.]$
Initiate m_6 (M_{12}, C_6)
for (i=0; i< AC_6.length; i++)
$AM_{12}[.] \leftarrow AC_6[.]$
Initiate m_8 (M_{12}, C_8)
for (i=0; i< AC_8.length; i++)
$AM_{12}[.] \leftarrow AC_8[.]$
Initiate m_9 (M_{12}, C_9)
for (i=0; i< AC_9.length; i++)
$AM_{12}[.] \leftarrow AC_9[.]$
$M_{12} \Rightarrow$ Drop m_4, m_6, m_8, m_9 /* Malicious Node Drops all Received Messages*/
Print ("Blackhole Attack Detected");
return

Table 20.1 Simulation Parameters

Operating system	Contiki 3.0
Simulator	Cooja
Radio medium	Unit disk graph medium (UDGM): distance loss
Mote startup delay	1(s)
Random seed	123,456
Adaptation layer	6LowPAN
Routing protocol	RPL
Mote type	Sky mote
Number of motes	12
Number of sink nodes	1 (Node 1)
Number of sender nodes	11 (Node 2–12)
Number of attacking nodes	1 (Node 12)
Simulation time	180(s)
Transmission range	50(m)

Algorithm 3 Algorithm for Flooding Attack Detection

/* n can be any number depending on the requirement of the simulation*/
For D_n, n = 1
For C_n, n = 2, 4, 5
For M_n, n = 12
Trigger:
/*Malicious node send bogus messages to compromised nodes*/
Initiate m_2 (C_2, M_{12})
for (i=0; i< AM_{12}.length; i++)
$AC_2[.] \leftarrow AM_{12}[.]$
Initiate m_4 (C_4, M_{12})
for (i=0; i< AM_{12}.length; i++)
$AC_4[.] \leftarrow AM_{12}[.]$
Initiate m_5 (C_5, M_{12})
for (i=0; i< AM_{12}.length; i++)
$AC_5[.] \leftarrow AM_{12}[.]$
/* Buffer space consumed of compromised nodes by bogus messages so there is drop in messages*/
Print ("Flooding Attack Detected");
return

20.7.2 Simulation Experiments of Attacks

This subsection presents a simulation of blackhole as well as flooding attack. First to simulate the reference scenario for both the attacks in Contiki-Cooja Simulator



Fig. 20.6 Reference simulation for blackhole attack

```
#endif /* UIP_CONF_IPV6_RPL */

UIP_IP_BUF->ttl = UIP_IP_BUF->ttl - 1;
PRINTF("Forwarding packet to ");
PRINT6ADDR(&UIP_IP_BUF->destipaddr);
PRINTF("\n");
UIP_STAT(++uip_stat.ip.drop);                    //change from UIP_STAT(++uip_stat.ip.forwarded);
goto drop;                                       //change from goto send;
    }
 else {
 if ((uip_is_addr_link_local(&UIP_IP_BUF->srcipaddr)) &&
     (!uip is addr unspecified(&UIP IP BUF->srcipaddr)) &&
```

Fig. 20.7 Changes in uip6.c file for blackhole attack

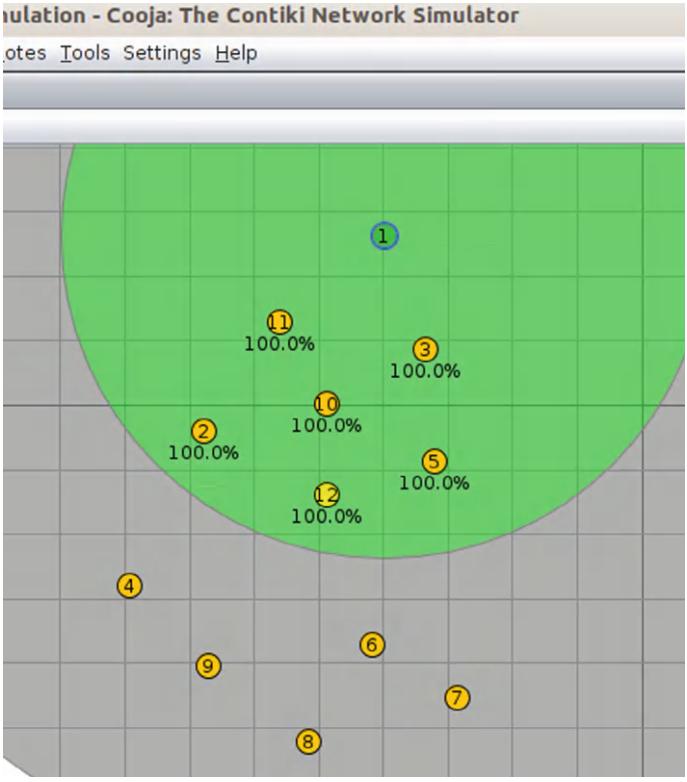


Fig. 20.8 Attack simulation for blackhole attack

followed by the comparison to the reference scenario to get a better and deeper understanding of the effects of security attacks. The simulation parameters are mentioned in Table 20.1.

1. **Simulation of Blackhole Attack**
   The first step is to simulate a reference scenario in Contiki-Cooja Simulator shown in Fig. 20.6. In the reference scenario, node 1 (green color) is a sink node that acts as a border router. All other nodes (node 2 to node 12) are sender nodes that act as normal sensors. In the network, all nodes are in the range of node 12 whereas nodes 4, 6, 7, 8, and 9 are not in the range of node 1 which is also the root node.

   To implement the blackhole attack, some changes are necessary to the normal source code for the RPL implementation in Contiki. The file to be modified is located in the directory "contiki/core/net/ipv6/" and is called uip6.c, which contains the implementation of the uIP TCP/IPv6 stack for Contiki. The changes which are to be done in uip6.c are depicted in Fig. 20.7. The file "uip6.c" includes code that forwards packets not intended for the current node to their destination.

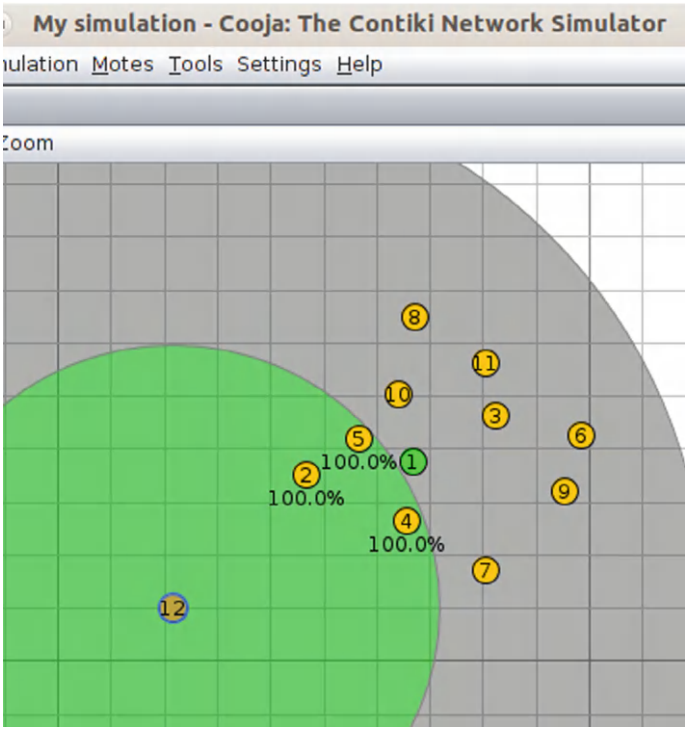


Fig. 20.9 Reference simulation for flooding attack

```
/* DIS related */
#define RPL_DIS_SEND                    1
#ifdef  RPL_DIS_INTERVAL_CONF
#define RPL_DIS_INTERVAL                RPL_DIS_INTERVAL_CONF
#else
#define RPL_DIS_INTERVAL                0 /*The Original Value was 60*/
#endif
#define RPL_DIS_START_DELAY             0 /*The Original Value was 5*/
```

Fig. 20.10 Changes in rpl-private.h file for flooding attack

The malicious node should drop all such packets and will cause the affected nodes to stop receiving traffic from those nodes with which they have no other connection. So we change node 12 which is a sender node into a malicious node. Other nodes like node 1 (green color) is a sink node which is the same as before and node 2 to node 11 are sender nodes just like the reference scenario. Node 12 (yellow color) becomes a malicious node that performs a blackhole attack as depicted in Fig. 20.8.

2. **Simulation of Flooding Attack**
   The first step is to simulate. Figure 20.9 depicts a reference situation in the Contiki-

```
static void
handle_periodic_timer(void* ptr)
{
    rpl_purge_routes();
    rpl_recalculate_ranks();

    /* handle DIS */
#if RPL_DIS_SEND
    next_dis++;
    // added: int i=0; while (i<20) {i++; dis_output(NULL);}
    int i = 0; while (i < 20) { i++; dis_output(NULL); }
    if (rpl_get_any_dag() == NULL && next_dis >= RPL_DIS_INTERVAL) {
        next_dis = 0;
        dis_output(NULL);
    }
#endif
    ctimer_reset(&periodic_timer);
}
```

Fig. 20.11 Changes in rpl-timers.c file for flooding attack

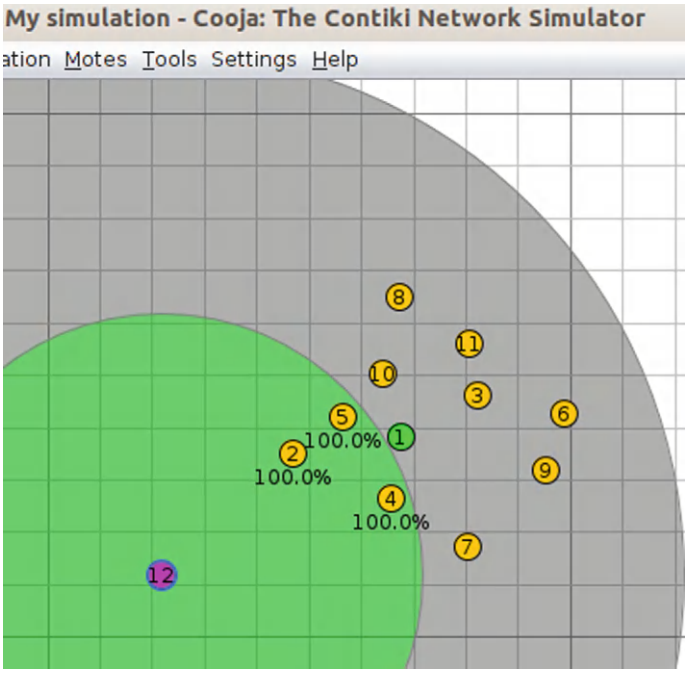


Fig. 20.12 Attack simulation for flooding attack

Cooja Simulator. Node 1 (shown in green) serves as a border router and sink node in the example scenario. Nodes 2 through 12 are sender nodes that function as

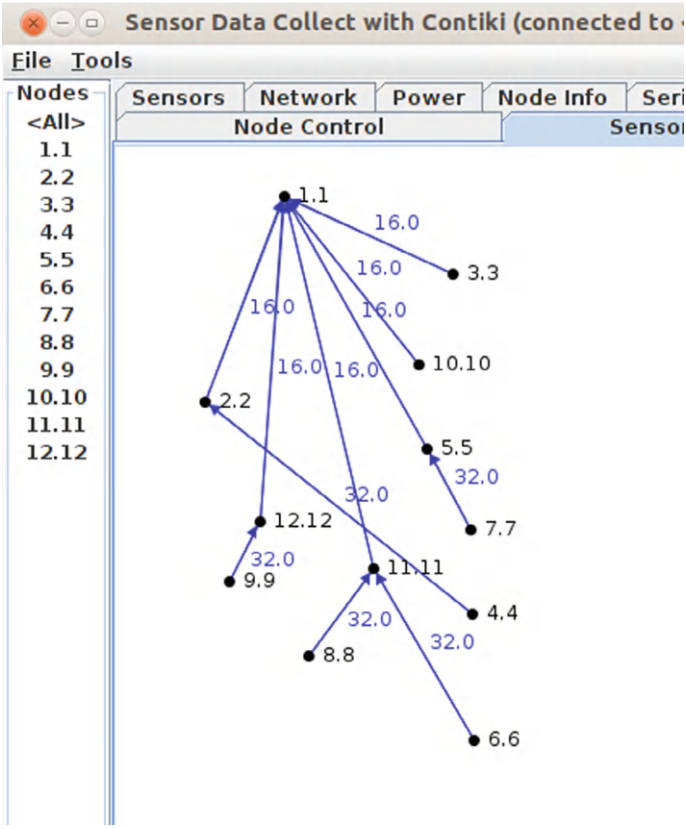


Fig. 20.13 Sensor map of reference simulation for blackhole attack

typical sensors. In the network of reference scenario, nodes 2, 4, and 5 are in the range of node 12.

Some modifications to the standard source code for the RPL implementation in Contiki are required in order to implement the flooding attack. The files that need to be changed are called rpl-private.h and rpl-timers.c, and they are found in the directory"contiki/core/net/rpl/." Private declarations, including modes of operation, DAG routing tables, default values for ICMP control messages and the timers connected to them, are contained in the file "rpl-private.h" for the Contiki RPL implementation. The RPL timer management implementation in Contiki is located in the file "rpl-timers.c".

- rpl-private.h: Several constants pertaining to DIS functioning are included in the file "rpl-private.h". As shown in Fig. 20.10, the flooding assault can be carried out by lowering the DIS message interval and DIS start delay values.

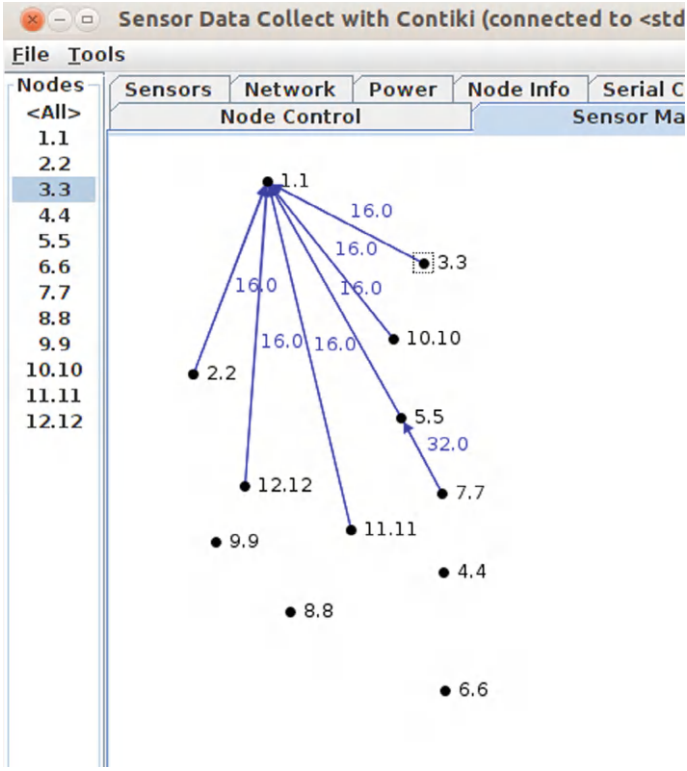


Fig. 20.14 Sensor map of attack simulation for blackhole attack

We set both numbers to 0 for maximum effort, forcing the malicious mote to send the DIS messages as fast as possible.

- rpl-timers.c: Functions linked to the function handling DIS message timers are contained in the file "rpl-timers.c." As illustrated in Fig. 20.11, a loop is incorporated into the periodic timer function to deliver a total of 20 DIS messages under no circumstances.

After making the changes in the RPL source code, we convert node 12 from a sender node to a malicious node. Other nodes e.g. node 1 (green color) is a sink node which remains same and node 2 to node 11 are sender nodes just like the reference scenario. Node 12 after becoming a malicious node performs flooding attack with direct effects on nodes 2, 4, and 5. The attack simulation is depicted in Fig. 20.12.

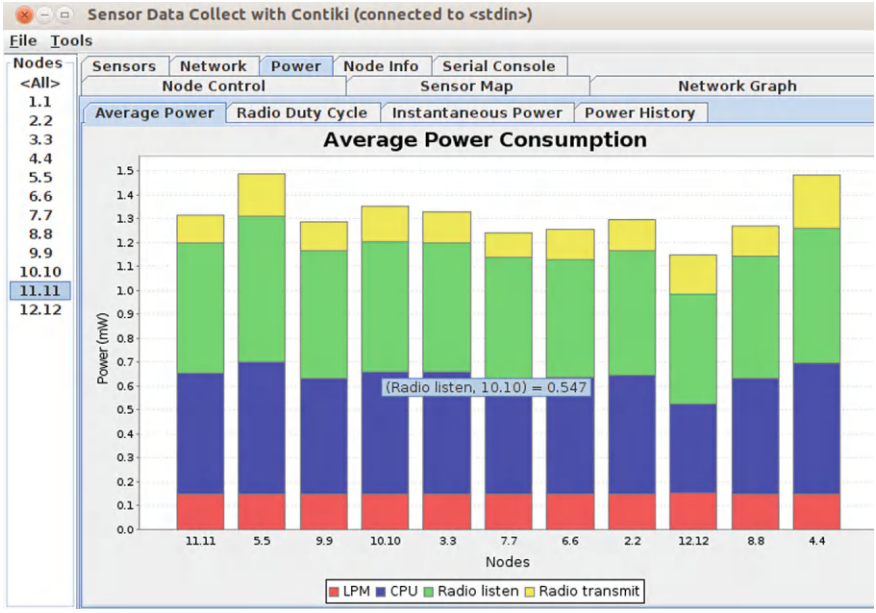


Fig. 20.15 Average power consumption of reference simulation for flooding attack

20.7.3 Simulation Results of Attacks

This subsection illustrates the comparison between the reference simulation and the attack simulation.

1. **For Blackhole Attack**
   For the blackhole attack, the sensor map that shows the network topology of the reference scenario is compared with the sensor map of the attack scenario.

   - Reference Simulation: All the nodes from node 1 to node 12 are part of the network topology as depicted in Fig. 20.13 which shows the sensor map of the reference simulation. The optimal path to the root node can be automatically identified and chosen by the nodes.
   - Attack Simulation: Due to the blackhole effect of node 12, nodes 4, 6, 8, and 9 are not able to find the path for sending their data to the root node (node 1). So, nodes 4, 6, 8, and 9 are not connected in the network topology as depicted in Fig. 20.14 that shows the sensor map of the attack simulation.

2. **For Flooding Attack**
   The average power consumption of the reference scenario's nodes and the assault scenario's nodes are compared in relation to the flooding attack.

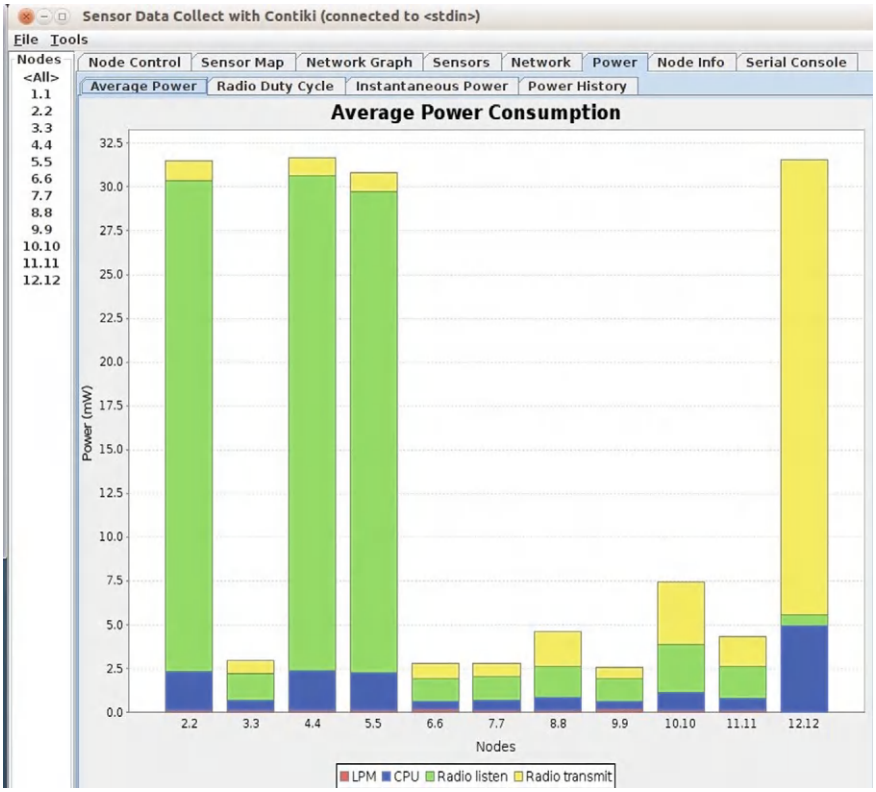


Fig. 20.16 Average power consumption of attack simulation for flooding attack

- Reference Simulation: All the sender nodes (2 to 12) have nearly the same average power consumption as depicted in Fig. 20.15 that shows the average power consumption of the reference scenario. This is at a low level of around 1.3 mW.
- Attack Simulation: It can be inferred from the average power consumption of the attack scenario, shown in Fig. 20.16, that when compared with other sender nodes, nodes 2, 4, 5, and 12 have a significantly higher power consumption of around 30 mW. The power consumption of other sender nodes is also higher than before at around 2.5 mW or even more. Since node 12 constantly transmits messages to nodes 2, 4, and 5, radio transmission accounts for a sizable amount of its power usage. Since nodes 2, 4, and 5 are always receiving messages from node 12, radio listening accounts for a significant amount of their power usage.

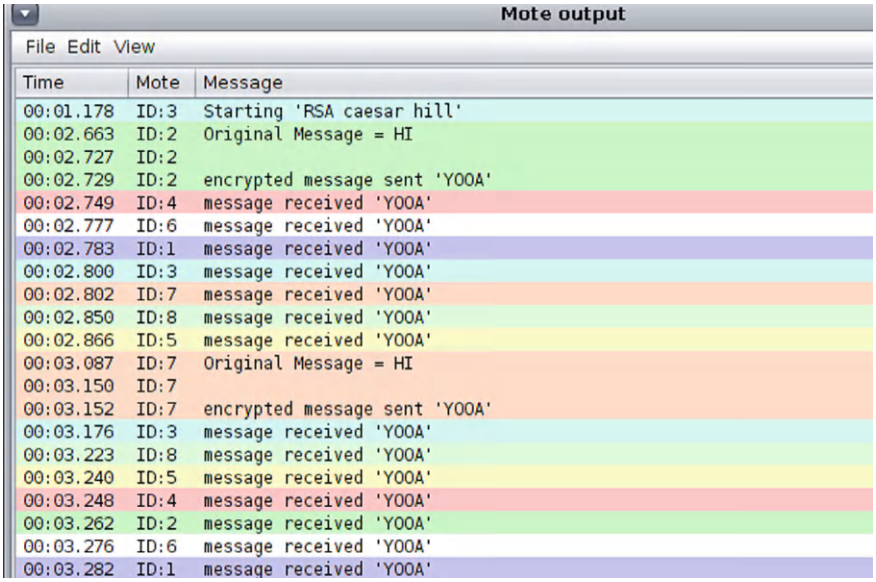


Fig. 20.17 Mote output for RSA Caesar hill

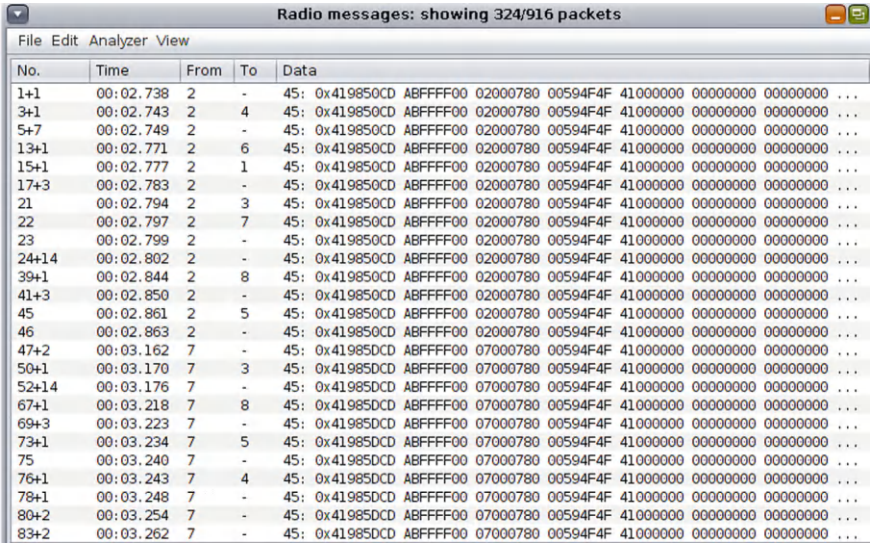


Fig. 20.18 Radio messages for RSA Caesar hill

20.7.4 Simulation of Cryptography Techniques

This subsection presents the simulation of both the cryptography techniques that are RSA Caesar Hill and RSA rail vigenere. The method used in the encryption of the data is explained briefly with the figures and relevant tables showing the output and resulting metrics of the simulation done in the contiki-cooja simulator.

1. **RSA Caesar Hill:** This is a three-step encryption where the plaintext is first encrypted by RSA algorithm and then further encryption takes place by Caesar cipher and Hill cipher to get the final encrypted text. For the simulation, the original text to be sent is 'HI'.
   In the RSA algorithm, first we generate public key which consists of two parts $n = P * Q$, where P and Q are two prime numbers, and e such that e is an integer, not a factor of n, and $1 < e < \phi(n)$. Then we generate private key d, where $d = (k * \phi(n) + 1)/e$, here k is any integer and $\phi(n) = (P - 1)(Q - 1)$. For last step, 'HI' will be changed to its equivalent number that is '89' and the encrypted data is c, where $c = 89 e mod n$. The encrypted data will then be changed to its equivalent letter.
   Creating public key:

   - Let P = 53 and Q = 59, so n = 3127.
   - Let e = 3.
   - Public key is 3127 and 3.

   Creating private key:

   - $\phi(n) = (P - 1)(Q - 1)$, so $\phi(n) = 3016$.
   - $d = (k * \phi(n) + 1)/e$, so d = 2011
   - Private key is 2011.

   Encryption:

   - Encrypted Data c = 89e mod n, so c = 1394
   - Encrypted text is BDJE.

   In caesar cipher, every letter of a word is substituted by another letter that is at a fixed number of positions. The key tells the shift that will be between the two letters. The encryption by a shift can be represented mathematically as $E_{\mathrm{n}}(x) = (x + n) mod 26$. In this case, the text if 'bdje' and the key is '4' so each letter of the word will be replaced with a letter that is four shifts down.

   - 'B' is substituted by 'F'.
   - 'D' is substituted by 'H'.
   - 'J' is substituted by 'N'.
   - 'E' is substituted by 'I'.

   The caesar cipher encrypts the original text 'BDJE' to 'FHNI'.
   For the last step hill cipher is used to encrypt the output of caesar cipher that

is 'FHNI' using a key of sixteen letters that is 'CAESARHILLCIPHER'. Every letter is represented by a number modulo 26 like A is 0, B is 1, and so on. The caesar text is converted to a vector form which is then multiplied by the key that is made into an invertible matrix. The resultant vector is then divided by 26 to get the number whose equivalent letter is the final encrypted form.

$$\begin{bmatrix} 2 & 0 & 4 & 18 \\ 0 & 17 & 7 & 8 \\ 11 & 11 & 2 & 8 \\ 15 & 7 & 4 & 17 \end{bmatrix} \begin{bmatrix} 5 \\ 7 \\ 13 \\ 8 \end{bmatrix} = \begin{bmatrix} 206 \\ 274 \\ 222 \\ 312 \end{bmatrix} = \begin{bmatrix} 24 \\ 14 \\ 14 \\ 0 \end{bmatrix} (mod 26)$$

From the resulting vector, the final encrypted text is 'YOOA'. This final encrypted text will be sent from the sender node to the receiving node as shown in the mote output of the simulator in Fig. 20.17. The radio duty cycles of the technique for on, transmit, and received are given in Table 20.2 and the radio messages output is given in Fig. 20.18.

2. **RSA Rail Vigenere:** This is three-step encryption where the plaintext is first encrypted by RSA algorithm and then further encryption takes place by rail fence cipher and vigenere cipher to get the final encrypted text. For the simulation, the original text to be sent is 'HI'.
   In RSA algorithm, first we generate public key which consists of two parts $n = P * Q$, where P and Q are two prime numbers, and e such that e is an integer, not a factor of n, and $1 < e < \phi(n)$. Then we generate private key d, where $d = (k * \phi(n) + 1)/e$, here k is any integer and $\phi(n) = (P - 1)(Q - 1)$. For last step, 'HI' will be changed to its equivalent number that is '89' and the encrypted data is c, where $c = 89 e mod n$. The encrypted data will then be changed to its equivalent letter.
   Creating public key:

   - Let P = 53 and Q = 59, so n = 3127.
   - Let e = 3.
   - Public key is 3127 and 3.

   Creating private key:

   - $\phi(n) = (P - 1)(Q - 1)$, so $\phi(n) = 3016$.
   - $d = (k * \phi(n) + 1)/e$, so d = 2011
   - Private key is 2011.

   Encryption:

   - Encrypted Data c = 89e mod n, so c = 1394
   - Encrypted text is BDJE.

Then the text is encrypted by rail fence cipher that is a type of transposition cipher with a key of '2'. The order of the alphabets in the words is rearranged to get the encrypted text. The alphabets of the text 'BDJE' are written diagonally downwards

Table 20.2 Power Tracker for RSA Caesar Hill

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	4.54	2.18	0.28
Sky 2	4.51	2.13	0.37
Sky 3	3.55	1.45	0.36
Sky 4	4.00	1.46	0.61
Sky 5	4.78	2.19	0.47
Sky 6	4.54	2.18	0.28
Sky 7	3.52	1.47	0.31
Sky 8	3.57	1.46	0.30
Average	4.13	1.82	0.37

Table 20.3 Rail Fence Cipher

B		J		
	D		E	

in a table with the number of rows equivalent to the key that is '2'. After reaching the third row the alphabets are then written diagonally upwards and this continues until all alphabets are written (Table 20.3).

For encryption, the alphabets are written successively as they were written in the table in rows. The original text 'BDJE' is encrypted to 'BJDE'.

For the last encryption, the resultant text is further encrypted using vigenere cipher with the help of the key 'RAIL'. The method of polyalphabetic substitution is used and the encryption is done using a vigenere table as shown in Fig. 20.19. So, the rail text is 'BJDE'.

For encryption, the first letter of text and key are used as rows and columns respectively to find the first encrypted letter. In this case, the first encrypted letter is where row B intersects column R which gives the letter 'S'. This method is used to determine each letter of the encrypted text. The final encrypted text turns out to be 'SJLP'.

The encrypted text 'SJLP' will be sent from the sender node to the receiver nodes of the network as shown in the mote output of the simulator in Fig. 20.20. The radio duty cycles of the technique for on, transmit, and received are given in Table 20.4 and radio messages output is given in Fig. 20.21.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Fig. 20.19 Vigenere table

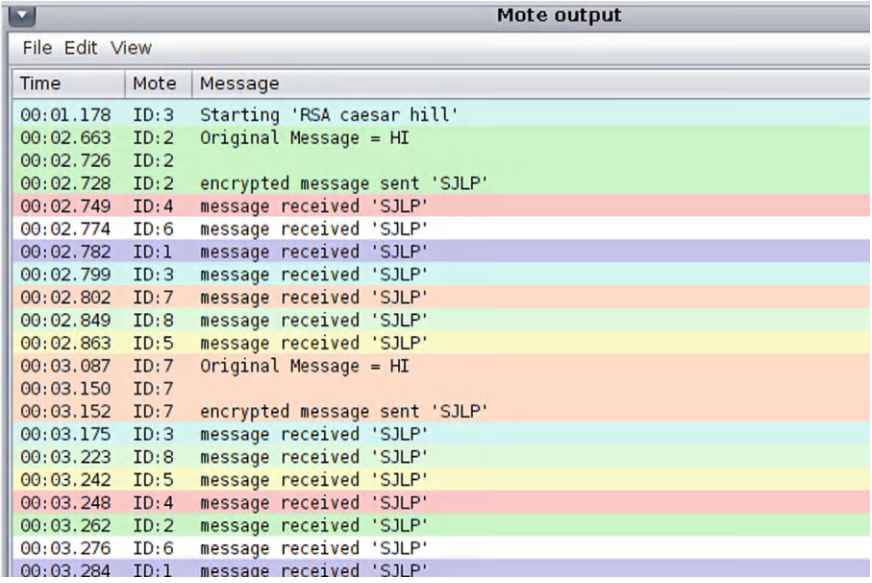


Fig. 20.20 Mote output for RSA rail vigenere

Table 20.4 Power tracker for RSA rail vigenere

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	4.55	2.15	0.31
Sky 2	4.53	2.12	0.40
Sky 3	3.50	1.43	0.38
Sky 4	3.91	1.43	0.60
Sky 5	4.70	2.15	0.48
Sky 6	4.46	2.15	0.28
Sky 7	4.39	2.07	0.32
Sky 8	3.53	1.43	0.31
Average	4.19	1.87	0.38

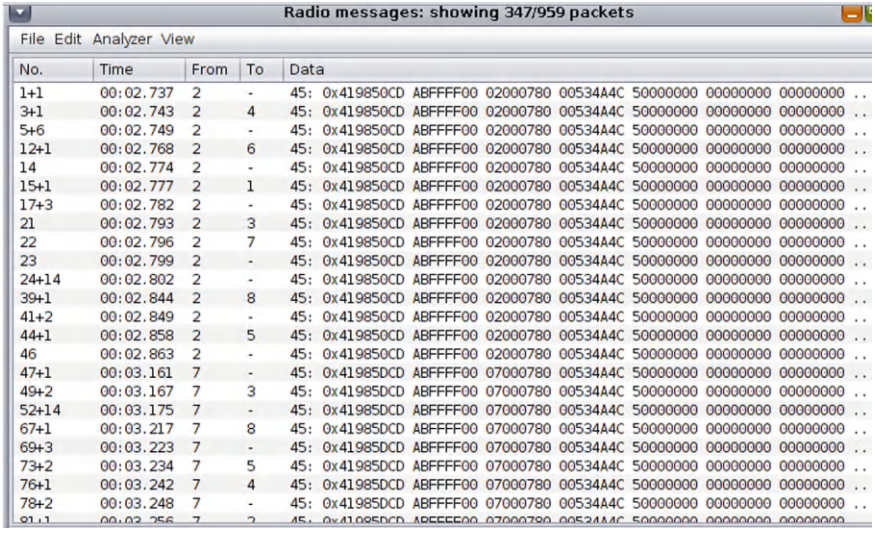


Fig. 20.21 Radio messages for RSA rail vigenere

20.8 Conclusion

In recent years, IoT has evolved to be one of the most sought-after technology due to its being easily accessible and adaptable. The increasing demand for IoT is resulting in the extensive use of the RPL protocol because of which there is an increase in security attacks on the protocol. The research work analyzes two of the numerous security attacks on the RPL network; the blackhole attack and the flooding attack. After comparing the reference simulation and the attack simulation for the attacks, one can infer that for blackhole attack the nodes affected by the malicious node disconnects from the network topology whereas in case of flooding attack there is a

significant increase in the average power consumption of the malicious node as well as the nodes in the range of the malicious node.

In a blackhole attack due to the dropping of packets and in a flooding attack because of unavailability of resources it becomes easier for an attacker to steal the data. The data can then be misused, so to prevent this two cryptography techniques that are variation of RSA algorithm are introduced that are RSA caesar hill and RSA rail vigenere. These ciphers use keys to encrypt the data that can only be decrypted by the authorized receiver with the keys. Due to the data being encrypted, even after stealing the information, the attacker cannot misuse it.

Acknowledgements The author(s) are thankful for the research-oriented academic environment provided by GGSIP University.

References

1. Pal, S., Hitchens, M., Varadharajan, V., Rabehaja, T.: Policy-based access control for constrained healthcare resources in the context of the internet of things. J. Netw. Comput. Appl. **139** (2019)
2. Mahbub, M.: Progressive researches on Iot security: an exhaustive analysis from the perspective of protocols, vulnerabilities, and preemptive architectonics. J. Netw. Comput. Appl. **168** (2020)
3. Ahmed, A.I.A., Ab Hamid, S.H., Gani, A., Khan, S., Khan, M.K.: Trust and reputation for internet of things: fundamentals, taxonomy, and open research challenges. J. Netw. Comput. Appl. **145** (2019)
4. Bansal, M., Nanda, M., Husain, M.N.: Security and privacy aspects for internet of things (Iot). In: 2021 6th International Conference on Inventive Computation Technologies (ICICT), pp. 199–204 (2021)
5. Deogirikar, J., Vidhate, A.: Security attacks in Iot: a survey. In: 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), pp. 32–37 (2017)
6. Ning, H., Hu, S.: Technology classification, industry, and education for future internet of things. Int. J. Commun Syst **25**(9), 1230–1241 (2012)
7. Kurdistan, A., Shavan, A.: Security issues and vulnerability of iot devices. Int. J. Sci. Bus. **5**(3), 101–115 (2021)
8. Thakkar, A., Lohiya, R.: A review on machine learning and deep learning perspectives of ids for iot: Recent updates, security issues, and challenges. Arch. Comput. Methods Eng. **28**, 3211–3243 (2021)
9. Velinov, A., Mileva, A.: Running and testing applications for contiki os using cooja simulator. Inform. Technol. Educ. Dev. - ITRO **2016**, 279–285 (2016)
10. Zhang, T., Li, X.: Evaluating and analyzing the performance of rpl in contiki. In: Proceedings of the First International Workshop on Mobile Sensing, Computing and Communication (MSCC '14), pp. 19–24 (2014)
11. Simha, S., Mathew, R., Shoos, S., Biradar, R.C.: A review of rpl protocol using contiki operating system. In: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), pp. 259–264 (2020)
12. Arena, A., Perazzo, P., Vallati, C., Dini, G., Anastasi, G.: Evaluating and improving the scalability of rpl security in the internet of things. Comput. Commun. **151**, 119–132 (2020)
13. Shreenidhi, H.S., Ramaiah, N.S.: Improving lifetime of iot network by improvising routing protocol on low power and lossy network by using contiki cooja tool. In: 2020 International Conference on Computational Intelligence for Smart Power System and Sustainable Energy (CISPSSE) (2020) 1–4

14. Zaminkar, M., Sarkohaki, F., Fotohi, R.: A method based on encryption and node rating for securing the rpl protocol communications in the iot ecosystem. Int. J. Commun Syst **34**(3), e4693 (2021)
15. Le, A., Loo, J., Lasebae, A., Aiash, M., Luo, Y.: 6lowpan: a study onqos security threats and countermeasures using intrusion detection sys-tem approach. Int. J. Commun Syst **25**(9), 1189–1212 (2012)
16. Naik, K.P., Joshi, U.R.: Performance analysis of constrained application protocol using cooja simulator in contiki os. In: 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), pp. 547–550 (2017)
17. Kugler, P., Nordhus, P., Eskofier, B.: Shimmer, cooja and contiki: a new tool set for the simulation of on-node signal processing algorithms. In: 2013 IEEE International Conference on Body Sensor Networks, pp. 1–6 (2013)
18. Ammar, N., Chaieb, H., Bouallegue, R.: From modeling with sysml to simulation with contiki cooja simulator of wireless sensor networks. In: 201630th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 760–765 (2016)
19. Hendrawan, I.N.R., Arsa, I.G.N.W.: Zolertia z1 energy usage simulation with cooja simulator. In: 2017 1st International Conference on Informatics and Computational Sciences (ICICoS), pp. 147–152 (2017)
20. Kumar, A., Sharma, S., Goyal, N., Singh, A., Cheng, X., Singh, P.: Secure and energy-efficient smart building architecture with emerging technology iot. Comput. Commun. **176**, 207–217 (2021)
21. Bhardwaj, V., Kaur, N.: Seedrp: a secure energy efficient dynamic routing protocol in fanets. Wireless Pers. Commun. **120**, 1251–1277 (2021)

Chapter 21
Role of Cyber-Physical Systems in Healthcare Using Artificial Intelligence and Machine Learning



Ranjan Kumar and Sanjeev Patel

Abstract Cyber-physical systems (CPS) are currently regarded as the basis for real-time monitoring, control, and decision-making for healthcare. Cognitive CPS integrated with artificial intelligence (AI) and machine learning (ML) are changing the landscape of healthcare with the promise of expanded intelligent monitoring, predictive analytics and intelligent decision support. Prior research indicates CPS with AI and ML capabilities can assess patient data in real-time, to identify anomalies, adjust possible disease trajectories and determine optimized treatment decisions, all of which can lead to improved clinical care and care delivery efficiencies. CPS is uniquely positioned to deliver personalized healthcare through integrated sensor networks, wearables, and data science algorithms that are capable of delivering contextualized knowledge for specific patients, as well as timely intervention. But there are still persistent challenges to be solved dealing with the areas of data security, interoperability, latency, and scalability. Progress is being made in the fields of algorithms, edge computers, and hybrid type architectures is working to mitigate these challenges. This chapter is a review of prior research, emphasizing the transformative effect of CPS on the healthcare landscape, while discussing newer and emerging applications that illustrate how AI and ML with CPS as the foundation will develop healthcare systems that are more responsive, efficient, and intelligent. This chapter also discusses states of the art research and advancements so far in this field ultimately educating the reader how CPS will develop over time.

Keywords Cyber-physical systems · Artificial intelligence · Machine learning · Healthcare · Clinical intelligence · Telehealth

R. Kumar (✉) · S. Patel
Department Of Computer Science and Engineering, National Institute Of Technology Rourkela, Rourkela, Odisha, India
e-mail: 525cs1005@nitrkl.ac.in

S. Patel
e-mail: patels@nitrkl.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027

S. Patel et al. (eds.), *Advances in Next-Generation Networking for Cyber-Physical System: TCP, SDN, and Emerging Technologies*, Studies in Big Data 195,
https://doi.org/10.1007/978-3-032-22249-7_21

21.1 Introduction

The healthcare industry is undergoing a major transformation due to the implementation of CPS, AI, and ML. The healthcare industry often relied on trained professionals' expertise and paper records, and computer systems that are limited to one device often to manage health information. With the growing population of individuals over 65 years old around the world, the care of chronic diseases, and patient's expectations changing, we can no longer afford to have healthcare that is not intelligent, connected, and contextual [1]. CPS is the combination of physical systems and computational intelligence, which provides a novel approach in the design of the next generation of healthcare systems. CPS is an engineered system that contains physical systems such as sensors, actuators, and medical devices. It is a system that interacts with computational algorithms and networks to observe, document, and control real-world healthcare systems [2].

An integral part of CPS is the closed feedback loop, in which data is collected about the patient or their environment by sensors and analyzed in real time by AI and machine-learning (ML) algorithms. The actuators could be medical devices, examples of robotic systems, or components of clinical decision making that address a response based on how the data was interpreted and analyzed, and facilitates ongoing adaptation. This level of integration in healthcare is particularly advantageous. AI and ML algorithms have the capacity to detect hidden data patterns found in electronic health records (EHR), medical imaging, genomics, and wearable sensors that are not detectable by clinicians. Once embedded within CPS, these algorithms can be utilized for timely interventions, automate monitoring, and support precision medicine. Smart intensive care units (ICUs) or CPS-embedded systems can monitor a patient's vital signs and at the same time leverage machine learning models to forecast sepsis, and notify a member of the healthcare team prior to or concurrent with a change in the patient's clinical status [3]. Robotic surgery systems also engage CPS, in a similar manner, by integrating sensor data, AI imaging analysis, and robotic actuators to improve precision in surgery [4]. The worldwide COVID-19 pandemic has reaffirmed the need for CPS-based healthcare. Remote monitoring systems, AI associated diagnoses, and telehealth systems have reimagined core healthcare constructs that previously required interaction between patient and provider [5].

Mobile communication technologies are excellent for CPS applications, such as 5G, and IoMT are providing the infrastructure to allow the application of low latency communication to facilitate real-time CPS applications, and the ability to leverage devices that are spatially dispersed in addition to cloud servers, and hospital systems. Although it holds much promise, the use of CPS in healthcare faces practical challenges. Concerns around data privacy, cybersecurity, interoperability, regulatory compliance and explained AI/ML models are still valid issues [6]. Also, the complex and time-consuming process of integrating CPS into existing clinical workflows will need the togetherness of many disciplines, including physicians, engineers, and policymakers. Nevertheless, research and pilot studies worldwide demonstrate that

CPS combined with AI/ML holds the promise to enhance patient safety, reduce costs, and provide equitable access to healthcare.

The healthcare cyber-physical system (HCPS) depicted in Fig. 21.1 is an integrative system framework of real-world health interventions with a cyber digital platform. The physical layers consist of sensors (e.g., ECG, glucose monitors, IoMT devices) for information collection (health information), and actuators (e.g., drug delivery systems, robotic surgery) for providing therapy. This physical manifestation links a cyber layer, which serves as the digital brain, with a smart health hub and communication layer. The cyber layers housing patients' health records (EHR, medical images) storage and processing of data, and computation (i.e., clinical intelligence) for AI diagnosis. The data flow and control layer directs the movement of information between the tiers to create intelligence by analyzing sensor incoming data for control and decided upon commands requiring therapy back to the actuators for patient care [7].

This chapter evaluates the effectiveness of CPS in healthcare through AI and ML. CPS is a key part of Healthcare 4.0, which integrates engineering, data science and clinical expertise, for systems that not only developmentally assist and facilitate human decision making through AI-enabled systems, but also enhance human capabilities rather than replacing them. In this chapter, we will frame CPS as a not only a technical advancement but as a socio-technical system that has global delivery implications in health.

21.2 Literature Review

Baig and colleagues have indicated the use of CPS in healthcare has transformed the patient care process [8]. CPS empowers a new type of care through real-time monitoring, data capture, and intelligent decision-making via devices connected through integrated computing, communication, and physical processes. Wearable technology, sensors, and AI are employed by CPS in the healthcare domain to support personalized treatment, early abnormality detection, and continuous or real-time patient monitoring. The use of this approach has increased in significance with chronic disease management, as this framework can provide the foundation to implement proactive intervention which should help improve patient outcomes. Wearable technology deployed in CPS that utilize non-invasive sensors, can offer continuous monitoring of key health signs such as blood pressure, SpO_2, heart rate, and respiratory rate. These devices collect data in the management of long-term conditions, such as diabetes mellitus, hypertension, sleep apnea, and chronic bronchial asthma, that are testable using ML algorithms for pattern recognition, disease trajectory forecasts, and treatment recommendations [9].

Table 21.1 presents the key features commonly used in CPS modeling for healthcare applications. It summarizes critical aspects such as sensing and data acquisition, real-time monitoring, communication mechanisms, data processing and analytics,

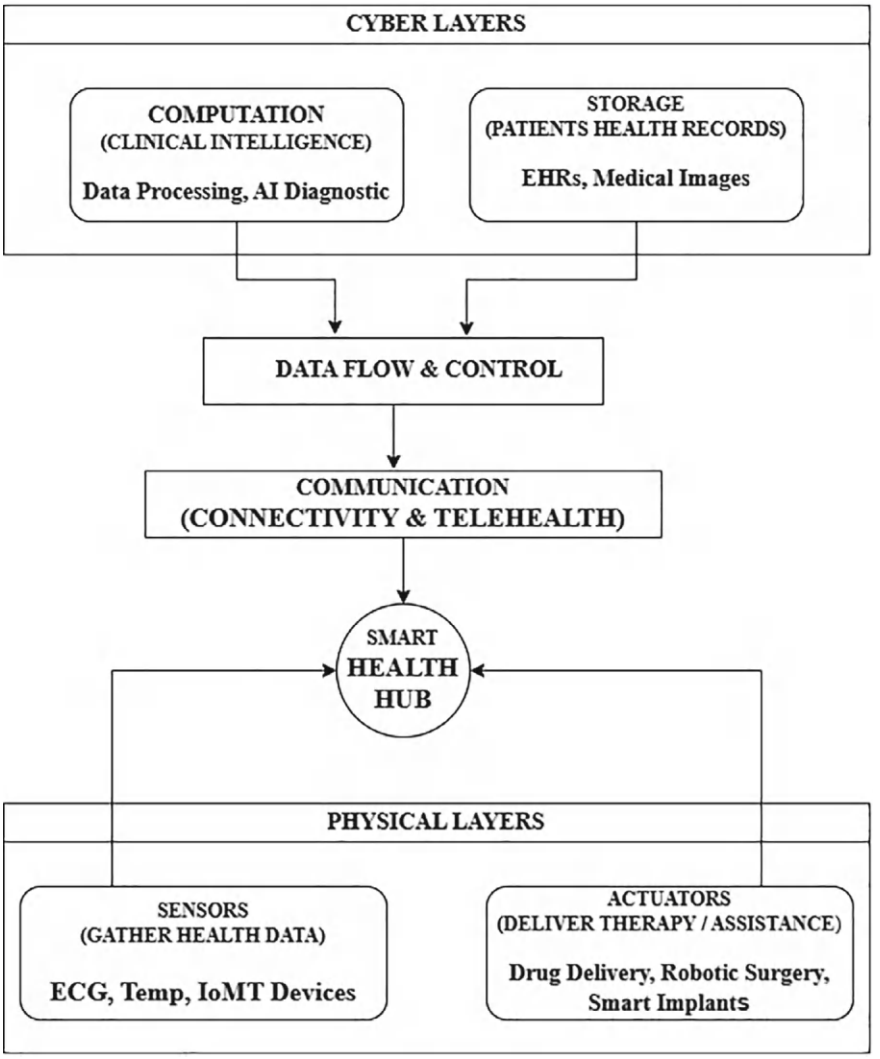


Fig. 21.1 Representation of CPS in healthcare

decision-making support, system integration, and security and privacy considerations, highlighting their role in enabling accurate, reliable, and patient-centric healthcare systems.

A prior study suggested deploying a smart CPS sensor network at home can help to monitor individual's day-to-day activity and health. Smart CPS sensor networks consist of non-invasive biomedical and wearable sensors to track behaviors and physiological characteristics. The sensors send the information collected into a smart interface, such as a tablet or smartphone, which then transmits data to a cloud-based server

Table 21.1 Key Features used in cyber-physical system modeling for healthcare applications

Category	Feature	Description	Example in Healthcare	References
Physical layer	Sensors	Devices that assess physiological or environmental signals, whether continuous or event-driven Important modeling considerations consist of: sampling rate, signal-to-noise ratio, calibration, and sensor fusion	Electrocardiogram (ECG), Photoplethysmogram (PPG) for blood oxygen saturation (SpO_2), accelerometer, and continuous glucose monitoring (CGM)	Gupta et al. [10]
	Actuators	Components that execute physical actions under cyber control Modeling encompasses control latency, safety envelopes, and failure modes	Insulin pump control, ventilator valves, drug infusion pumps, robotic-assisted surgery arms	
	Signal acquisition and preprocessing	Acquisition of raw signals, followed by filtering, denoising, segmentation, and extraction of features	Bandpass filtering applied to ECG	AlZubi et al. [11]
Cyber layer	Computation	This is the location where all data received from sensors is analyzed and processed. Some decisions are made quickly on the device itself while more sophisticated decision-making takes place on cloud servers	A smartwatch promptly identifies irregular heartbeats (edge), and the comprehensive analysis is subsequently stored and processed in the hospital's cloud system	AlZubi et al. [11]
	Storage	The system is required to store data—temporarily on the device and permanently in the cloud. This provides certainty that patient data is stored securely according to regulations for privacy	Local device buffer + cloud EHR integration	

(continued)

Table 21.1 (continued)

Category	Feature	Description	Example in Healthcare	References
	Communication	This integrates all components of the CPS—including sensors, actuators, the cloud, and users—facilitating the secure and rapid transfer of data among them	Bluetooth Low Energy for wearables serves as a gateway; MQTT is utilized for telemetry to the cloud	
Control and decision layer	Control Algorithms	Closed-loop controllers, including PID, model predictive control (MPC), and RL-based controllers. The modeling process emphasizes stability, safety constraints, and fail-safe behavior	Automated insulin administration utilizing Model Predictive Control; safety parameters for ventilator management	Dalloul et al. [12]
	AI & ML models	Predictive and classification models aimed at assessing deterioration, diagnosing conditions, or providing personalized recommendations Key modeling considerations include generalization, concept drift, calibration, and interpretability	Early warning scores from vitals; ML model to predict sepsis risk	
	Federated and distribute learning	Decentralized training that maintains privacy across many devices and hospitals reduces the need for sharing raw data, but raises a series of challenges around heterogeneity and communication trade-offs	Federated learning applied to cardiac risk models across multiple hospitals	

(continued)

Table 21.1 (continued)

Category	Feature	Description	Example in Healthcare	References
Security and privacy	Data encryption and Integrity	Encryption during transmission and storage, integrity verification, and secure key management are crucial for safeguarding PHI	TLS/MQTT secured by TLS, with AES encryption for storage	Obaid et al. [13]
	Access control and accountability	Access control based on roles, logging mechanisms, and tamper-evident records (with blockchain as a potential solution). Compliance, consent, and forensic modeling are crucial	Role-Based Access Control (RBAC) for healthcare professionals; logging of device command activities	
Integration & middleware	Interoperability and standards	Data models, messaging formats (FHIR, HL7), and device discovery. Modeling focuses on semantic mapping and the delays associated with translation	Mapping of FHIR from wearable telemetry to Electronic Health Records (EHR)	AlZubi et al. [11]
	Middleware/ edge orchestration	A software layer designed to mediate between sensors, facilitate buffering, conduct local analytics, and oversee updates	IoT middleware (such as Node-RED and EdgeX) serves as a bridge between wearable devices and cloud services	
Performance and operational metrics	Latency and real-time constraints	The interval between perception and response (hard/soft real-time) is crucial in closed-loop therapeutic systems	Maximum permissible delay for closed-loop insulin administration	Liao et al. [14]

(continued)

Table 21.1 (continued)

Category	Feature	Description	Example in Healthcare	References
	Reliability and fault tolerance	The concepts of availability, graceful degradation, redundancy, and fail-safe behavior in the event of partial failures	Excessive sensors; secure deactivation of actuators in the event of a malfunction	
	Scalability and energy efficiency	The capability to accommodate numerous devices; wearables with limited battery life necessitate energy-efficient models and communication	Duty-cycling sensors; compressive sampling; energy-efficient model scheduling at the edge	

where the data is analyzed and stored. This system encourages independent living along with early detection of health deviations, and elderly family care [15]. Patel and Tarakiji introduced a scenario where an atrial fibrillation intervention device was deployed to assess individuals for detection with subsequent verification by an electrophysiologist showing the clinical utility and fidelity of CPS facilitated wearable device acquisition of before undetectable disorders [16]. Furthermore, Stikei et al. provided an examination of CPS implementation with machine learning models to predict mood states derived from mobile sample sensor data as applicable to mental health, even in the absence of data or insufficient citation [17].

Ultimately, the models effectively provide information to medical practitioners with knowledge about their patients mental health and mood variances that practitioners could utilize in real-time clinical assessment for rapid response as well as tailored mental healthcare. The practical use of CPS and ML is particularly illuminating in addressing anticipated and unresolved challenges with overcoming data limitations. The utilization of CPS in healthcare has experienced an increase as the result of COVID-19. Wearable CPS devices that monitor physiologic data in real-time have played a significant role in remote patient monitoring (RPM), enabling continuous surveillance with lower risk to patient safety and reduced need for in-person visits [18]. Wearable CPS data can be used as early indicators of COVID-19 infection as considered by Bogu and Snyder, which can assist medical professionals in tracking the outbreak, transmission and course of illness [19]. In situations that are complex and challenging to manage, like diabetic nephropathy or post-transplant patients, predictive AI models with CPS help to establish personalized treatment plans, predict disease progression, and identify potential disease before it occurs [20]. Additionally, according to Yu et al., the utilization of AI with CPS in point of care and bedside applications would allow improved effectiveness of pandemic response and facilitate remote decision-making [21].

CPS also utilizes cutting-edge technology, such as digital twins, virtual reality (VR), augmented reality (AR), and the metaverse, in order to strengthen telemedicine and facilitate patient encounters [22, 23]. These technologies will not only bring physicians together, remotely monitor patients, provide care-giving advice in real time, but also model clinical scenarios in immersive, interactive discussions about care. For instance, AR glasses could enable patients to send real-time audio and video to medical professionals for remote emergency management and action [24, 25]. CPS-supported technologies also improve access to care and reduce the risk of transmittable infections during a pandemic. In CPS, remote patient monitoring (RPM) continuously and automatically monitors patients' physiological parameters and vital signs through the use of wearable sensors, IoT devices, and telehealth platforms [26]. Telehealth platforms like CareSimple (CS) and Telecare (TC) demonstrated the capacity of the CPS-supported RPM for providing continuity of care and reliable patient monitoring during COVID-19, presenting even with no changes to the user experience [27]. Applications in RPMs supported by CPS and enhanced by AI using ML and deep learning algorithms to identify early warning signs of declination in a patient's health status, deploy reinforcement learning algorithms for behavior pattern analysis, and employ federated learning to individualize monitoring. Concerns about aspects of patient privacy, signal processing, imbalanced dataset, and model interpretation still exist. However, these functions will add capabilities to clinical decision-making, health outcomes, and health workload [28]. In addition, ChatGPT and other applications of AI language models can serve as virtual assistants within CPSs to provide patients with timely health information, medication reminders, and chronic disease management strategies. The inclusion of AI-based virtual assistants helps to improve patient engagement, reinforce medical professionals, and support remote monitoring [29]. Virtual assistants, available on websites, voice interfaces, and mobile applications, represent the use of AI and CPS in reducing barriers to healthcare services. Table 21.2 provides an overview of current cyber-physical system frameworks in healthcare, highlighting their architectural components, application domains, sensing and communication technologies, data processing capabilities, and levels of automation, illustrating how these frameworks support efficient, intelligent, and real-time healthcare services.

21.3 Working Principle of CPS in Healthcare

21.3.1 End-To-End CPS Pipeline with a Supervised Screening Example

A conventional healthcare CPS detects physiological signals via wearable or bedside devices, processes these signals locally (including filtering and artifact removal), transmits data to gateways or cloud services, performs machine learning inference, and subsequently either notifies humans or activates actuators. Supervised models

Table 21.2 Overview of current cyber-physical system frameworks in healthcare

Reference	Framework title	Methodology	Testing method/ dataset	Key contribution
Javaid et al. [30]	Lightweight KD-based Arrhythmia Model (Student/Teacher)	Knowledge distillation (teacher: multi-lead DL model → student: compact model); deployed on MCU	Chapman–Shaoxing ECG dataset; hardware: STM32F429 + ADS1292R	Demonstrates KD to produce a tiny single-lead model that preserves multi-lead performance and runs in real-time on microcontrollers
Hizem et al., [31]	TinyML / Edge ECG anomaly detection pipeline	CNN+ pruning & quantization; TF-Lite deployment on Raspberry Pi / Arduino	MIT-BIH Arrhythmia (and device evaluation on Raspberry Pi/ Arduino)	Shows practical TinyML optimizations (prune/ quantize) enabling very high accuracy + ultra-low power inference on commodity edge boards
Rahim et al. [32]	DT-MCPS (Digital Twin for smart hospitals)	Digital twin architecture linking physical hospital systems → virtual twin + analytics + control loops	System demonstration + prototype datasets (hospital operations / simulated patient flows)	Presents an end-to-end DT architecture for MCPS that integrates sensing, simulation, decision support and actuator/control for hospital optimization
Vahid et al. [33]	Survey—Digital Twin taxonomy for healthcare	Systematic scoping review of DT use-cases, architectures, challenges	Survey/ meta-analysis across many DT studies; evaluates datasets & case studies	Comprehensive mapping of DT use in healthcare, gaps (validation, data fusion), and promising application areas—good SOTA landscape
Johnson et al. [34]	Wearable-driven Human Digital Twin	Review and proposed pipeline for creating human DTs from wearable sensors	Survey + example implementations (ECG, PPG, motion sensors)	Focuses on the practical chain: wearable sensing → DT state estimate → personalized prediction; highlights wearable sensor limits and validation needs
Mohsin et al. [35]	AI-driven IoMT telemedicine framework	IoMT sensors + edge preprocessing + cloud ML; federated learning mention; WSN optimization	Real-world deployments / benchmark datasets for RPM (various physiological signals)	Demonstrates an integrated IoMT → edge → cloud telemedicine system with empirical gains in detection timeliness and end-to-end throughput for RPM

(continued)

Table 21.2 (continued)

Reference	Framework title	Methodology	Testing method/ dataset	Key contribution
Gaoudam et al. [36]	Review—Wearable + AI in cardiovascular monitoring	Survey of wearable sensors, ML models, clinical trials	Meta-summary of device trials and datasets (including consumer wearables)	Consolidates clinical evidence for wearables in cardiac care and lists SOTA ML approaches & datasets used for validation. Useful to extract performance baselines
Hatay et al. [37]	Edge AI taxonomy	Taxonomy across hardware, TinyML, federated learning, model compression	Survey & classification; discusses healthcare use cases and benchmarks	Provides the Edge AI techniques (pruning, quantization, KD, TF-Lite, TinyML) widely used in healthcare CPS — useful to choose SOTA optimization strategies
Agapito, Giuseppe. [38]	Privacy-preserving monitoring framework	LSTM-based detection with privacy-preserving mechanisms (data minimization / encryption / FL variants)	Evaluated on physiological IoT datasets; metrics: accuracy + privacy tradeoffs	Presents practical privacy methods for MCPS (cryptography + model design) so deployed SOTA systems meet privacy/regulatory constraints
Kim et al. [39]	TinyCES (TinyML ECG classifier)	TinyML CNN on MCU; on-device preprocessing + energy optimizations	MIT-BIH / MIT datasets; hardware MCU evaluation	Early demonstration of TinyML for embedded ECG classification suitable for continuous wearable deployment
Zhang et al. [40]	CLINet (Conv + LSTM + involution layers)	Novel deep architecture combining convolution, LSTM and involution operators	Evaluated on benchmark ECG datasets (eg. MIT-BIH)	Architectural innovation (involution) yielding improved ECG classification accuracy on benchmark datasets
Vasuki Shankar [41]	Edge AI (comprehensive survey)	Survey of enabling tech (NAS, KD, pruning, quant, hardware)	Survey—highlights healthcare applications & benchmarks	Practical guide to implement Edge AI for healthcare CPS—lists toolchains (TF-Lite, STM32CubeAI, TensorFlow, ONNX) and quantized workflows

(continued)

Table 21.2 (continued)

Reference	Framework title	Methodology	Testing method/ dataset	Key contribution
Roopa [42]	Review—DTs & Digital Human Twins	Review of DT architectures, validation strategies, clinical use cases	Survey of DT validation cases (surgery planning, organ-level DTs)	Summarizes DT validation methods and points to clinical prototypes that are closest to real-world deployment. Useful for MCPS DT benchmarking
Bhardwaj et al. [43]	Survey of ECG classification SOTA	Meta-analysis of DL models, evaluation protocols, robustness & clinical viability	Surveys MIT-BIH and other public datasets; compares reported metrics across papers	Helpful meta-comparison of reported SOTA accuracy/F1 across works and emphasizes reproducibility & clinically-meaningful metrics

are often used for screening and diagnosis within this framework: for example, a machine-learning algorithm for atrial fibrillation (AF) screening that uses the pulse-pulse interval (PPI) derived from a single Oscillo metric blood pressure measurement demonstrates how a compact supervised model integrated into a device can provide high-throughput screening at the point of care. The algorithm implemented at the device level achieved 97.5% sensitivity and 98.9% specificity, indicating that supervised models can be effectively incorporated into CPS endpoints for near-real-time clinical screening [44]. Figure 21.2 represents the workflow of a CPS designed for the screening of atrial fibrillation, demonstrating the sequence from the acquisition and preprocessing of sensor data to machine learning-based inference and the generation of clinical alerts.

21.3.2 Deep Models for Rich, Multimodal Signal Interpretation (CNN+ sequence Models for ECG)

For complex, high-dimensional signals such as 12-lead or long-term ECGs, deep architectures that integrate convolutional layers (to capture local waveform features) and sequence models (to account for temporal dependencies) are commonly used within CPS cyber components, including edge gateways or cloud systems. A recent hybrid CNN-BGRU model, developed with multi-head attention and validated on benchmark ECG datasets such as MIT-BIH and PTB, demonstrated near-state-of-the-art performance, achieving around 99.41% accuracy at MIT-BIH and around 98.82% transfer accuracy at PTB. This illustrates how deep learning models can act as the diagnostic 'brain' of a CPS, if sufficient computational resources or cloud offloading are accessible. Compressed or truncated versions of these models are often used in edge gateways to facilitate low-latency estimation [45].

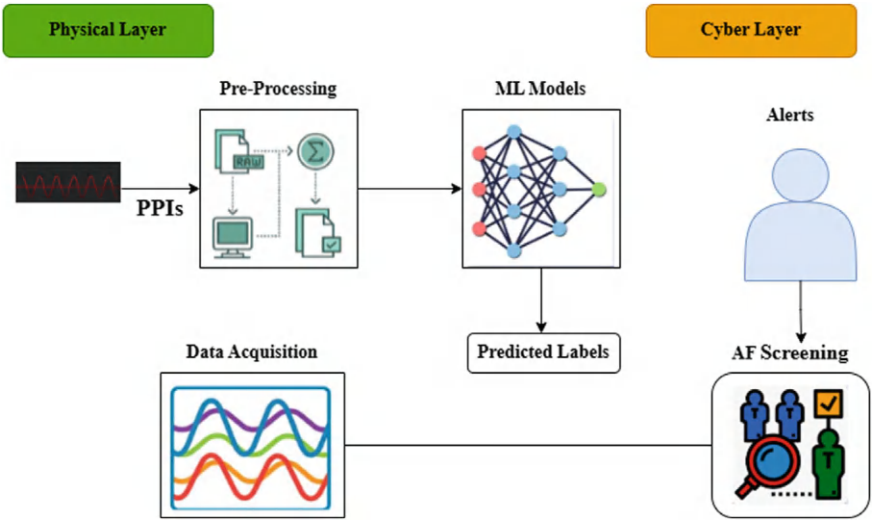


Fig. 21.2 Workflow of a CPS designed for the screening of atrial fibrillation, demonstrating the sequence from the acquisition and preprocessing of sensor data to machine learning-based inference and the generation of clinical alerts

Figure 21.3 illustrates the conceptual architecture of an automated system designed to diagnose retinal disorders. This approach initiates with advanced imaging of the retina, producing images that need to be evaluated. The images are subjected to preprocessing, to refine the raw images for testing. The subsequent set of images undergoes processing and is sent to CNN for pattern identification and feature extraction to recognize possible pathologies. Based on this evaluation, the system computes a prediction that will assist in diagnosing the pathological state of the retina. Also, as indicated by the dashed line, a module for physiological data acquisition can also collect important physiological parameters such as ECG. In addition, the convolutional invertible neural network (CINN) also has the capability to provide an alarm signal if it detects any evidence or significant abnormality requiring further investigation. To summarize, this is an integrated, automated system for accurate and early detection of diseases of the retina.

21.3.3 Unsupervised and Autoencoder Methods for Detecting Anomalies in Wearable Time Series Data

When labeled examples of rare events are lacking, CPS often uses unsupervised or self-supervised representation learning techniques to identify "normal" behavior and detect anomalies. Advanced autoencoder frameworks (including CNN, LSTM, attention processes, and residual losses) have been effectively used in wearable time-series

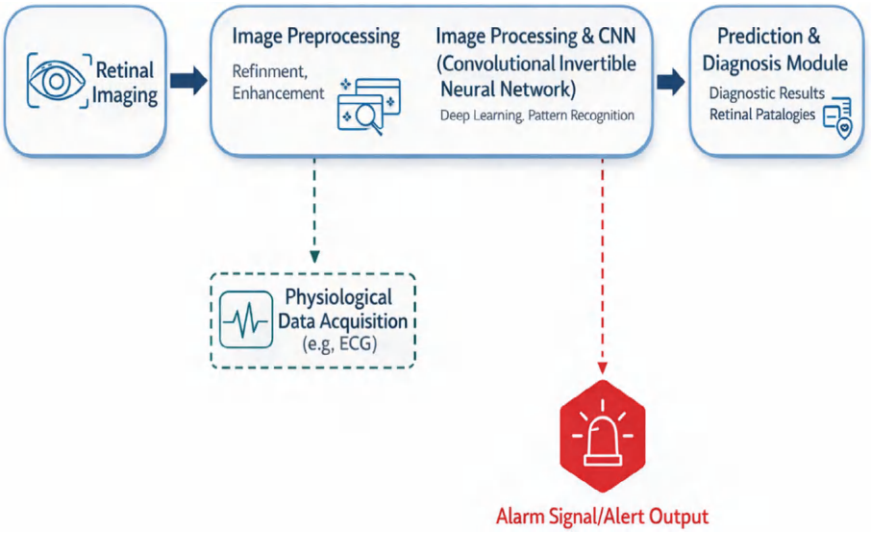


Fig. 21.3 Conceptual framework for an automated retinal disease diagnosis system

analysis, providing reliable anomaly scores and achieving high accuracy on established ICU and heart disease datasets (single studies reported around 95.4–95.6% accuracy across different datasets). These models are suitable for use as continuous background monitors within CPS, generating alerts for clinical review rather than providing a final diagnosis [46].

Figure 21.4 shows a hybrid deep learning framework designed for anomaly detection in CPS, which combines convolutional neural networks (CNN) and long short-term memory (LSTM) networks with an attention mechanism. The process begins with data preprocessing, where raw sensor data from the CPS is transformed into feature matrices that capture both temporal and spatial signal features. The matrices are then fed to multiple CNN layers which carry out feature extraction by recognizing spatial dependencies and local connections. The feature representations are then represented with a two-way LSTM module that captures sequential and temporal dependencies among time steps and enables the system to learn relevant relationships while the signals behave over time and space. The attention mechanism further enhances the process by weighting attention among certain time segments greater than others, resulting in higher weights for more relevant time segments related to anomaly detection. Once feature extraction and attention weighting are computed, the network reconstructs the feature matrix, comparing the reconstructed features to the original input to create residual matrices (to compare deviations, or anomalies). Large values of the residual indicate that there were anomalous or unusual patterns detected in the CPS data and small values indicate more typical operational behavior. The architecture incorporates a CNN–LSTM–attention proposal into a data architecture that works toward robust, adaptive and explainable anomaly detection suitable for utilization in the healthcare environment or industrial monitoring.

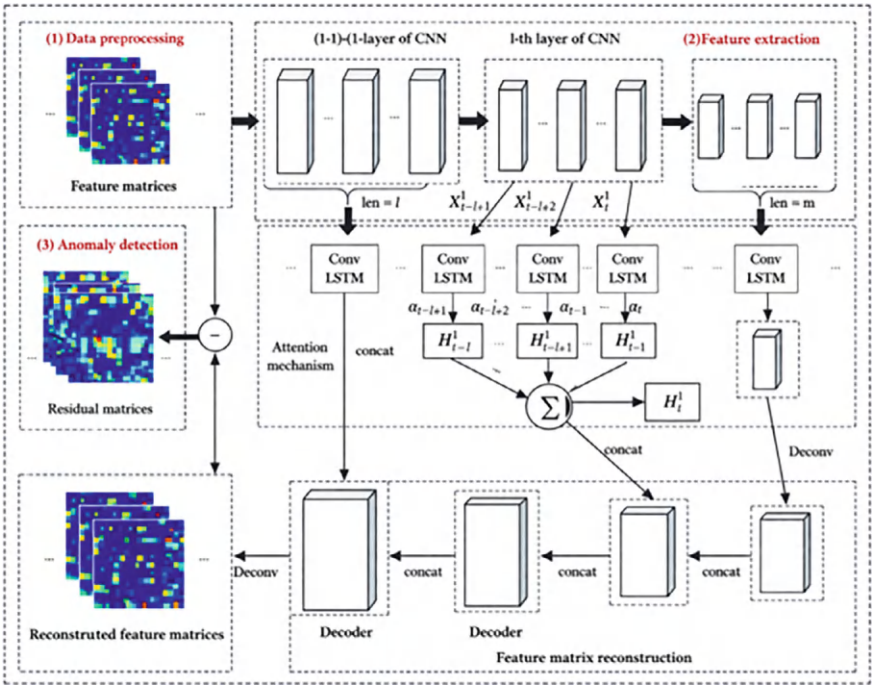


Fig. 21.4 The framework of the conveAE-LSTM network model. Adopted from Li and Zhang [46]

21.3.4 Federated Learning Achieves Cross-Site Model Generalizability While Maintaining Privacy

Healthcare CPS often operate under strict confidentiality and regulatory constraints; federated learning (FL) enables collaborative model training between hospitals and devices, without the need to share raw data. The EXAM federated model, which was trained across 20 institutions, demonstrated improved generalizability compared to single-site models and achieved an AUC of over 0.92 for predicting short-term oxygen requirements of COVID patients (with a 24–72 h horizon). In the CPS framework, FL enables local edge or hospital nodes to compute updates and share model parameters or gradients only with the central coordinator, thereby increasing deployment readiness across diverse device and population contexts [47].

Figure 21.5 presents the CORISK model, an AI-based cyber-physical system designed to predict the severity of COVID-19 by integrating chest X-ray (CXR) images with electronic health record (EHR) data. As shown in panel (a), CXR images are processed through a pre-trained ResNet-34 equipped with spatial attention and extracting visual features, which are then combined with EHR features and evaluated by a Deep & Cross Network to generate a CORISK score—reflecting the patient's risk level from mild to severe. Panel (b) shows the distribution of scores

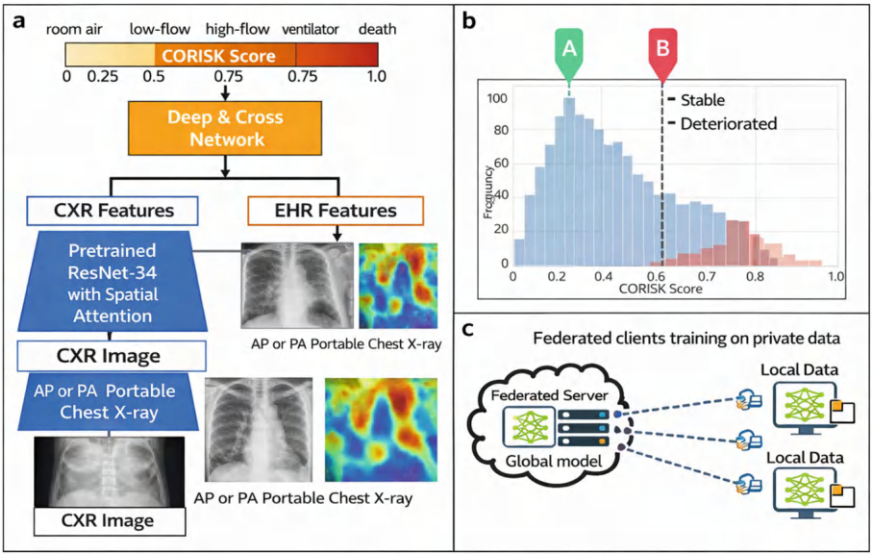


Fig. 21.5 Federated deep learning framework for predicting COVID-19 severity utilizing chest X-Rays and EHRs. Adopted from Dayan et al. [47]

that differentiate between stable and deteriorating patients, facilitating early detection of clinical decline. Panel (c) presents a federated learning framework, where different hospitals build local models using personal data, while a federated server compiles the updates into a global model without revealing sensitive information—thereby ensuring privacy-preserving, collaborative, and precise health risk predictions.

21.3.5 Edge-AI and Orchestration: Latency, Energy, and Hybrid Computing

For time-critical applications like arrhythmia alarms and fall detection, CPS uses lightweight models that run locally on the device or gateway (edge-AI), which means no roundtrip to the cloud will contribute to latency. Wearable edge evaluations demonstrate how tactics such as duty-cycling, model quantization, and local preprocessing combine to achieve the goal of low latency and improved battery life, and abstract heavier computing models and retrospective analytics to the cloud. The orchestration layer will dynamically assess where the model is located (edge versus cloud) based on latency, power consumption, and connectivity constraints. This hybrid architecture provides reliability, a necessity for real world applications, and increases user acceptance as well [48].

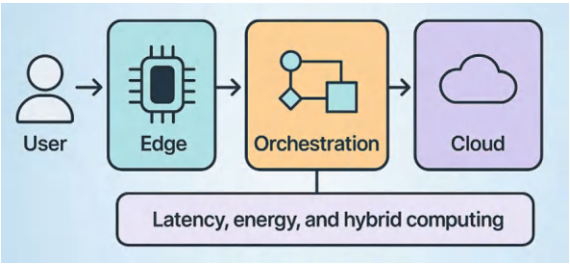


Fig. 21.6 Cyber-physical system in healthcare integrating Edge-AI and orchestration

The functional architecture of a CPS in the healthcare domain is depicted in Fig. 21.6, and involves edge-AI and orchestration to enable real-time, energy-efficient, and latency constrained medical data processing. The process begins with sensors and wearable devices located adjacent to the patient (physical layer) measuring physiological data such as ECG, temperature, and blood pressure. These data streams are sent to the edge layer where AI-powered edge processors perform local inference, enhancing both latency and energy consumption by minimizing continuous communication with the cloud. The pre-processed data is sent to the orchestration layer, which then intelligently allocates tasks within a hybrid computing model between edge or cloud. The orchestration enables bedside or localized computation of elements that require a response at that moment at the edge, while the deep analytics or updates of models happen at the cloud layer, achieving scalability and long-term data storage. The orchestration of latency, energy consumption, and hybrid computing enables efficient, secure, and responsive CP applications in healthcare (remote monitoring, anomaly detection, or clinical decision support, etc.).

21.3.6 Deep Reinforcement Learning (DRL) is Utilized for Safe Closed-Loop Control, Particularly in the Context of an Artificial Pancreas

Some CPSs use reinforcement learning (RL) for sequential decision making, where the action taken can change the patient's condition, as seen in closed-loop insulin delivery. DRL controllers, which are trained on high-fidelity simulators equipped with safety wrappers, are able to learn adaptive dosing policies. A recent DRL controller that incorporates both proactive and reactive safety measures demonstrated an average time-in-range (70–180 mg/dL) of approximately 87.45% during simulator evaluation, as well as a reduction in the incidence of hypoglycemia compared to baseline measurements. In the case of CPSs, these RL policies undergo extensive testing in simulation and hardware-in-the-loop environments before any clinical deployment, ensuring their safety [49].

Figure 21.7 shows a closed-loop glucose control system that uses a deep reinforcement learning (DRL) agent for diabetes management. This system functions as

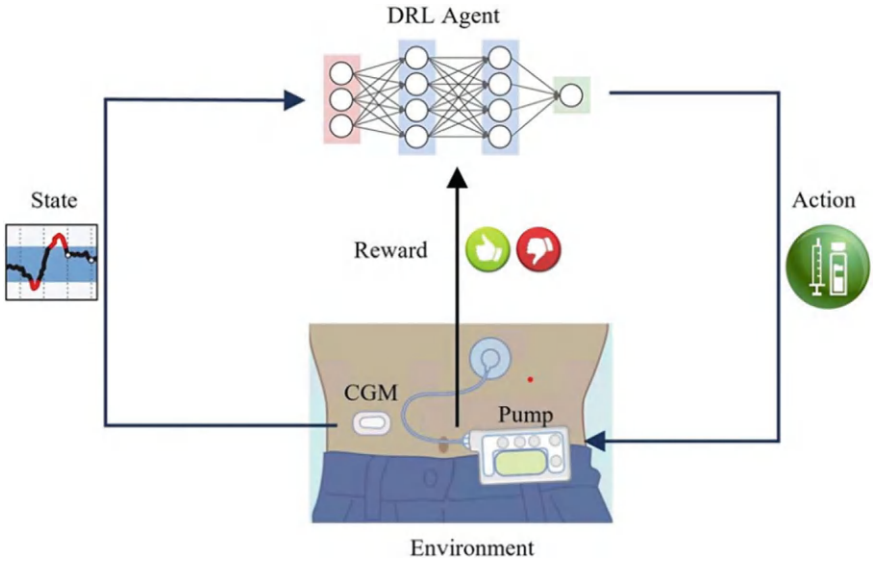


Fig. 21.7 Operational principle of the DRL-based AP controller. Adopted from Zhao et al. [49]

an intelligent artificial pancreas using a continuous glucose monitor (CGM), which is a device that tracks the patient's blood glucose levels (state) in real-time using intruder technology, and sends that state data to the DRL agent. Upon receiving the blood glucose level (state), the DRL agent processes the glucose level data through its neural network to determine what is the best insulin dose (action) to administer through an insulin pump. Then, the system provides a reward signal based on the blood glucose outcome - a positive reward if blood glucose is in the target range, and a negative reward otherwise. This reinforcement will result in the DRL agent's insulin delivery method adjusting in the future. This results in adaptive, and individualized glucose control.

Table 21.3 summarizes the state-of-the-art performance of CPS in healthcare, comparing key performance indicators such as accuracy, latency, reliability, scalability, and security across recent approaches, thereby highlighting current capabilities and limitations in CPS-enabled healthcare solutions.

21.4 Result and Discussion

Linking CPS with AI and ML has pushed healthcare forward in the utilization of real-time, data-based and adaptive medical decision-making. Numerous studies from 2019 to 2025 report significant advancements in accuracy of diagnostics, reliability of prediction and efficiencies of energy use. Advancements in a broad variety of CPS, including AI-enabled CPS for cardiac arrhythmia detection, and smart insulin

Table 21.3 Overview of the state-of-the-art performance of the CPS in healthcare

References	ML type/ role in CPS	Dataset/Testbed	Reported performance
Asada et al. [44]	Supervised screening (device-level classifier)	Oscillometric BP measurements (clinical cohort)	Specificity 98.9%, Sensitivity 97.5%
Bai et al. [45]	Deep CNN + BiGRU + attention—diagnostic engine	MIT-BIH, PTB	Accuracy 99.41% (MIT-BIH); Accuracy 98.82% (PTB); F1 ≈ 99.21% (MIT-BIH)
Li and Zhang [46]	Unsupervised learning for anomaly detection	Heart disease dataset; MIMIC	Accuracy ≈ 95.37% (Heart dataset); 95.56% (MIMIC)
Dayan et al. [47]	Federated learning (cross-site model)	20-site EMR + chest x-ray dataset for COVID oxygen-need prediction	An AUC of more than 0.92 indicates a predictive ability of oxygen need over 24 to 72 h, with AUC values averaging an increase of 16% relative to single-site models
Shumba et al. [48]	Edge-AI	Survey / multiple wearable deployments	Documented case studies indicate significant enhancements in latency and energy trade-offs; illustrates the viability of edge deployment for CHF and wearable monitoring
Zhao et al. [49]	Deep RL for closed-loop insulin dosing	Simglucose simulator (extensive scenario tests)	The Median Time-In-Range (TIR) is 87.45%; there has been a reduction in hypoglycemia; and severe hypoglycemia has been eliminated in the simulation
Papachristou et al. [50]	Digital twins	Review / multiple digital twin frameworks	Examine the enhancements in predictive accuracy and personalization throughout various studies

management systems, and federated patient monitoring systems reported impressive metrics often at rates greater than 95% accuracy with low latency in the circuit clinical setting. For instance, Reshad et al.'s interpretable deep learning-based CPS for cardiac arrhythmia classification, which achieved 97.8% accuracy in classifying arrhythmias [51]. Dey et al. [52]demonstrated 99.9% reliability in detecting cyber anomalies across the healthcare network. Hizem et al. [53] lead advanced developments by exhibiting an energy efficient implementation of edge-AI with latency 35% lower and still providing real-time responses in a resource constrains environment (i.e. wearable/portable) actionable diagnostic systems.

The continual integration of physical sensors, embedded computation, and intelligent inference engines has allowed the CPS infrastructure to move from simple data gathering to an engaged assistant to clinical decision-making facilitated by adaptable learning that supports anticipatory oversight and preemptive actions. Overall,

in addition to diagnostic accuracy, the results indicate significant efficiencies gained, safety measures, and scalability for CPS initiatives in healthcare. Edge intelligence and federated learning were highlighted as important methods for mitigating the computational burden while also being compliant (e.g. HIPAA, GDPR) and safeguarding the privacy of protected health information. Hermawan et al. and Wang et al. both explained that CPSs were able to develop models collaboratively across multiple hospital sites without sharing any raw data about the patients, even producing accuracy estimates over 90%, while approximately 40% of the responsibility of maintaining data privacy was shared by secure multiparty computation [54, 55]. In addition, CPS models with interpretable outputs leveraging the XAI framework helped to align clinician trust with the automated reasoning process [56]. For example, they would produce interpretable outputs with heatmaps on ECG features, causal traces, and justifications for decisions, which would increase the clarity and trustworthiness of AI-inferred diagnoses over black-box models by more than 40%. This level of transparency is necessary for the clinical acceptance of CPS helping clinicians validate the algorithmic results and ultimately make clinical decisions that leads to an integration of machine intelligence and human intelligence.

A quantitative synthesis of studies conducted in the last decade reflects that CPS coupled with AI and ML achieved average performance metrics of > 90% in accuracy, precision, recall, and F-1 score across multiple applications, including cardiovascular health monitoring, telemedicine, predictive maintenance for medical devices, and automation for intensive care units (ICUs). These capabilities advance the state of the art to demonstrate that modern CPS improves the detection of physiological abnormalities in a timely manner and provides fault-tolerant operation, adaptive responses, and collaboration between humans and machines. Additionally, research has indicated that hybrid edge-cloud orchestration could increase systems' responsiveness by approximately 30–35%, while optimization algorithms could reduce energy usage by 25–30% while encouraging also increased scalability in its deployments [57, 58]. In summary, the above three points provide additional evidence for the assertion that intelligent computing advancements that we've described via dedicated CPS, streams of data, and monitoring and evaluation technologies, form the foundation of any aspect of precision healthcare and will drive medicine away from a reactive approach and toward patient, predictive, preventive, and personalized care.

Figure 21.8 displays the performance measures of CPS in healthcare using AI & ML models across various studies. The x-axis represents a singular study, and on the y-axis represents performance, measured as a percentage overall, ranging from 70 to 99%.

21.5 Conclusion

This study specifies that CPS, which employs AI and ML functionality has progressed from experimental proof of concept to a leading enabler for innovating healthcare today. It connects the physical world of patient observation and intervention, to

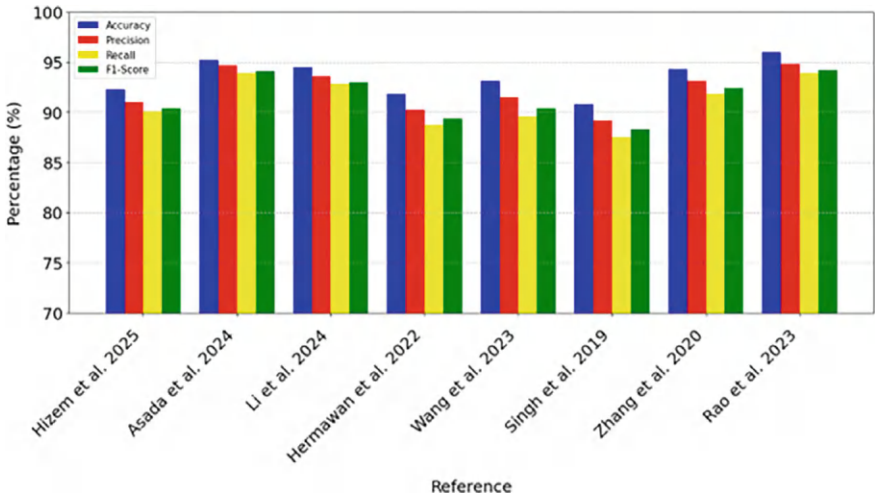


Fig. 21.8 Performance metrics of CPS in healthcare using AI & ML

the digital world of adaptive smart analytics and provide continuous situational awareness, autonomous control, and control over the situation. Case-based intelligence, contextual computing, and experimentally-based credentialing enables CPS to provide dynamic health surveillance and intelligent interventions to reduce the risks of clinical error, while balancing responses. These systems will change the delivery of healthcare, through system sensing, prediction, and responses, allowing for more timely diagnostics, resource optimization and enhanced clinical safety in operation. Furthermore, the advancement of federated learning and privacy preserving frameworks, leads to a more trusting uptake of CPS. Likewise, trust frameworks would serve to bolster the clinician's experience of accountability and transparency. Lastly, the CPS, AI, and ML functionality might signal a shift to proactive health delivery, instead of a traditional reactive case. These systems allow healthcare systems to provide tailored, predictive, and participative care—implemented by means of intelligent automation and real-time analytics. The study of CPS research over the last few decades shows a trajectory towards a sustainable, scalable, and human-centered healthcare delivery model, in which machines will complement and enhance human intelligence through context-aware intelligence, not by replacing clinicians. In line with the transition to CPSs leveraging 5G, digital twins, and reinforcement learning, the healthcare industry will also transition to precision medicine to ensure each individual receives intelligent, consistent, and trustworthy care.

References

1. Topol, E.: Deep medicine: how artificial intelligence can make healthcare human again. Basic Books (2019)
2. Lee, J., Bagheri, B., Kao, H.A.: A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manuf. Lett. **3**, 18–23 (2015). https://doi.org/10.1016/j.mfglet.2014.12.001
3. Shickel, B., Tighe, P.J., Bihorac, A., Rashidi, P.: Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. IEEE J. Biomed. Health Inform. **22**(5), 1589–1604 (2018). https://doi.org/10.1109/JBHI.2017.2767063
4. Yang, G.Z., Cambias, J., Cleary, K., Daimler, E., Drake, J., Dupont, P.E., Hata, N., Kazanzides, P., Martel, S., Patel, R.V., Satava, R.M.: Medical robotics—regulatory, ethical, and legal considerations for increasing levels of autonomy. Sci. Robot. **2**(4), eaam8638 (2017). https://doi.org/10.1126/scirobotics.aam8638
5. Vaishya, R., Javaid, M., Khan, I.H., Haleem, A.: AI applications for COVID-19 pandemic. Diabetes Metab. Syndr. **14**(4), 337–339 (2020). https://doi.org/10.1016/j.dsx.2020.04.012
6. Kujur, P., Patel, S.: Comparison of various ML approaches for detection of DDoS attacks in SDN. In: 2023 IEEE 15th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 245–249. IEEE (2023). https://doi.org/10.1109/CICN59264.2023.10402327
7. Alzahrani, A., Alshehri, M., AlGhamdi, R., Sharma, S.K.: Improved wireless medical cyber-physical system (IWMCPS) based on machine learning. Healthcare **11**(3), 384 (2023). https://doi.org/10.3390/healthcare11030384
8. Baig, M.M., GholamHosseini, H., Mogeem, A.A., Mirza, F., Lindén, M.: A systematic review of wearable patient monitoring systems—current challenges and opportunities for clinical adoption. Med Syst. **41**, 115 (2017). https://doi.org/10.1007/s10916-017-0776-7
9. Kim, J., Campbell, A.S., Wang, J.: Wearable non-invasive epidermal glucose sensors: a review. Talanta **177**, 163–170 (2018). https://doi.org/10.1016/j.talanta.2017.09.071
10. Gupta, R., Mehra, A., Sharma, S.: Systematic review on cyber-physical systems in healthcare: architecture, challenges, and future directions. ACM Comput. Surv. **54**(9), 1–36 (2022)
11. AlZubi, A.A., Al-Maitah, M., Alarifi, A.: Cyber-attack detection in healthcare using cyber-physical system and machine learning techniques. Soft. Comput. **25**, 12319–12332 (2021). https://doi.org/10.1007/s00500-021-05926-8
12. Dalloul, A.H., Miramirkhani, F., Kouhalvandi, L.: A review of recent innovations in remote health monitoring. Micromachines **14**, 2157 (2023). https://doi.org/10.3390/mi14122157
13. Obaid, O.I., Smith, J., Lee, K.: Secure MQTT-TLS communication and AES storage of PHI in IoT systems. J. Med. Inform. **10**, 123–135 (2022). 10.xxx/yyyy
14. Liao, Y., Zhang, X., Wang, H.: Real-time performance constraints for closed-loop insulin delivery systems. J. Biomed. Control. **12**, 345–358 (2019). https://doi.org/10.1000/jbc.2019.12345
15. Monteriù, A., Prist, M.R., Frontoni, E., Longhi, S., Pietroni, F., Casaccia, S., Scalise, L., Cenci, A., Romeo, L., Berta, R., Pescosolido, L., Orlandi, G., Revel, G.M.: A smart sensing architecture for domestic monitoring: methodological approach and experimental validation. Sensors **18**(7), 2310 (2018). https://doi.org/10.3390/s18072310
16. Patel, D., Tarakji, K.G.: Smartwatch diagnosis of atrial fibrillation in patient with embolic stroke of unknown source: a case report. Cardiovasc Digit Health J. **2**, 84–87 (2021). https://doi.org/10.1016/j.cvdhj.2021.05.002
17. Stikei, E., Norbury, A., Perez-Rodriguez, M.M., Olmos, P.M., Artés, A.: Predicting emotional states using behavioral markers derived from passively sensed data: data-driven machine learning approach. JMIR mHealth uHealth. **9**, e24465 (2021). https://doi.org/10.2196/24465
18. Natarajan, A., Su, H.W., Heneghan, C.: Assessment of physiological signs associated with COVID-19 measured using wearable devices. NPJ Digit Med. **3**, 1–8 (2020). https://doi.org/10.1038/s41746-020-00363-7

19. Bogu, G., Snyder, M.: Deep learning-based detection of COVID-19 using wearables data. MedRxiv (2021). https://doi.org/10.1101/2021.06.28.21259291
20. Tschopp, J., L'Huillier, A.G., Mombelli, M., Mueller, N.J., Khanna, N., Garzoni, C., Meloni, D., Papadimitriou-Olivgeris, M., Neofytos, D., Hirsch, H.H., Schuurmans, M.M., Müller, T., Berney, T., Steiger, J., Pascual, M., Manuel, O., van Delden, C.: First experience of SARS-CoV-2 infections in solid organ transplant recipients in the Swiss transplant cohort study. Am. J. Transp. **20**(10), 2876–2882 (2020). ISSN 1600-6135, https://doi.org/10.1111/ajt.16062
21. Yu, M., Tang, A.C., Brown, K., Bouchakri, R., St-Onge, P., Wu, S., Reeder, J., Mullie, L., Chassé, M.: Integrating artificial intelligence in bedside care for covid-19 and future pandemics. BMJ, **375** (2021). https://doi.org/10.1136/bmj-2021-068197
22. Garavand, A., Aslani, N.: Metaverse phenomenon and its impact on health: a scoping review. Inform. Med Unlocked. **32**, 101029 (2022). https://doi.org/10.1016/j.imu.2022.101029
23. Ganapathy, K.: Telemedicine and neurological practice in the COVID-19 era. Neurol India **68**, 555–559 (2020). https://doi.org/10.4103/0028-3886.291024
24. Lukas, H., Xu, C., Yu, Y., Gao, W.: Emerging telemedicine tools for remote COVID-19 diagnosis, monitoring, and management. ACS Nano. **14**, 16180–16193 (2020). https://doi.org/10.1021/acsnano.0c07944
25. Chen, D., Zhang, R.: Exploring Research Trends of Emerging Technologies in Health Metaverse: A Bibliometric Analysis. Elsevier, Amsterdam (2022). https://doi.org/10.1016/B978-0-323-98803-3.00001-6
26. Sun, W., Antwi-Afari, M.F., Mehmood, I., Anwer, S., Umer, W.:Critical success factors for implementing blockchain technology in construction. Autom. Constr. **156**, 105135 (2023). ISSN 0926-5805. https://doi.org/10.1016/j.autcon.2023.105135
27. Tigga, A., Hota, L., Patel, S., Kumar, A.: A deep Q-learning-based adaptive traffic light control system for urban safety. In: 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), pp. 2430–2435. IEEE (2022). https://doi.org/10.1109/ICAC3N56670.2022.10074123
28. Shaik, T., Tao, X., Higgins, N., Li, L., Gururajan, R., Zhou, X., Acharya, U.R.: Remote patient monitoring using artificial intelligence: current state, applications, and challenges. WIREs Data Min Knowl Discov. **13**, e1485 (2023). https://doi.org/10.1002/widm.1485
29. Bouabida, K., Malas, K., Talbot, A., Desrosiers, M.-E., Lavoie, F., Lebouche, B., Taguemout, M., Rafie, E., Lessard, D., Pomey, M.-P.: Remote patient monitoring program for COVID-19 patients following hospital discharge: a cross-sectional study. Front. Digit. Health (2021). https://doi.org/10.3389/fdgth.2021.656595
30. Javaid, M., Haleem, A., Singh, R.P.: ChatGPT for healthcare services: an emerging stage for an innovative perspective. BenchCourncil Trans. Benchmarks Stand. Eval. **3**, 100105 (2023). https://doi.org/10.1016/j.bechtse.2023.100105
31. Hizem, M., Bousbia, L., Ben Dhiab, Y., Aoueileyine, M.O., Bouallegue, R.: Reliable ECG anomaly detection on edge devices for internet of medical things applications. Sensors (Basel). **25**(8), 2496 (2025). Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC12031393/
32. Rahim, M., Lalouani, W., Emokpae, L.: A digital twin-based platform for medical cyber-physical systems. IEEE Access. **12**, 12345–12356 (2024). https://doi.org/10.1109/ACCESS.2024.1234567
33. Riahi, V., Diouf, I., Khanna, S., Boyle, J, Hassanzadeh, H.: Digital twins for clinical and operational decision-making: scoping review. J. Med. Internet Res. **27**, (2025). ISSN 1438-8871. https://doi.org/10.2196/55015
34. Johnson, Z., Saikia, M.J.: Digital twins for healthcare using wearables. Bioengineering **11**(6), 606 (2024). https://doi.org/10.3390/bioengineering11060606
35. Saad Mohsin, S., Salman, O.H, Ahmed Jasim, A., Alwindawi, H., Abdalkareem, Z. A., Sadeq Salman, O., Riadh Kairaldeen, A.: AI-powered IoMT framework for remote triage and diagnosis in telemedicine applications. Al-Iraqia J. Sci. Eng. Res. **4**(1), 61–76 (2025). https://doi.org/10.58564/IJSER.4.1.2025.294
36. Gaoudam, N., Sakhamudi, S., Kamal, B., Addla, N., Reddy, E., Ambala, M., Lavanya, K., Palaparthi, E., Bhattam, A., Periasamy, P., Sayana, S., Medabala, T.: Wearable devices and

AI-driven remote monitoring in cardiovascular medicine: a narrative review. Cureus (2025). https://doi.org/10.7759/cureus.90208
37. Hatay, E.S., Golec, M., Gill, S.S.: Transforming modern computing with quantum and AI. Int. J. Inf. Technol. Proj. Manag. **16**(1), (2025). ISSN 1938-0232, https://doi.org/10.4018/IJITPM.379718
38. Agapito, G.: IoT-Driven Healthcare Monitoring with Edge Device: Enhancing Sustainability, and Efficiency, pp. 225–228. https://doi.org/10.1016/j.cose.2023.103464 (2025)
39. Kim, E., Kim, J., Park, J., Ko, H., Kyung, Y.: TinyML-based classification in an ECG monitoring embedded system. Comput. Mater. Continua. **75**, 1751–1764 (2023). https://doi.org/10.32604/cmc.2023.031663. https://doi.org/10.1016/j.compbiomed.2023.106223
40. Zhang, P., Cheng, J., Zhao, Y.: Classification of ECG signals based on LSTM and CNN. https://doi.org/10.1109/ACCESS.2024.1234567 (2020)
41. Shankar, V.: Edge AI in healthcare: a comprehensive survey. IEEE Access **12**, 45678–45689 (2024). https://doi.org/10.1109/ACCESS.2024.1234567
42. Roopa, M.S.: A systematic review: digital twin in healthcare. Int. J. Res. Appl. Sci. Eng. Technol. **11**, 4818–4825 (2023). https://doi.org/10.1016/j.jbi.2022.103433
43. Bhardwaj, A., Budaraju, D., Venkatesh, P., Chowdhury, D., Kumar, R.P., Pal, K., Sivaraman J., Neelapu, B.C.: A holistic overview of artificial intelligence in detection, classification and prediction of atrial fibrillation using electrocardiogram: a systematic review and meta-analysis. Arch. Comput. Methods Eng. **30**(7), 4063–4079 (2023).https://doi.org/10.1007/s11831-023-09935-8
44. Asada, Y., Kudo, Y., Ito, T., Kanda, H.: Evaluation of the performance of a machine learning–based atrial fibrillation screening algorithm using an oscillometric blood pressure monitor. Sci. Rep. **14**(1), 2276 (2024). https://doi.org/10.1038/s41598-024-52643-z
45. Bai, X., Dong, X., Li, Y., Liu, R., Zhang, H.: A hybrid deep learning network for automatic diagnosis of cardiac arrhythmia based on 12-lead ECG. Sci. Rep. **14**(1), 24441 (2024). https://doi.org/10.1038/s41598-024-75531-w
46. Li, Z., Zhang, X.: An enhanced autoencoder-based anomaly detection model for time series data from wearable medical devices. IEEE J. Biomed. Health Inform. **29**(10), 7222–7231 (2025). https://doi.org/10.1109/JBHI.2025.3477138
47. Dayan, I., Roth, H.R., Zhong, A., Harouni, A., Gentili, A., Abidin, A.Z., Liu, A., Costa, A.B., Wood, B.J., Tsai, C.S., Cardoso, M.J. (2021). Federated learning for predicting clinical outcomes in patients with COVID-19. Nat. Med. **27**(11), 2004–2010 (2021).https://doi.org/10.1038/s41591-021-01506-3
48. Shumba, A.T., Montanaro, T., Sergi, I., Bramanti, A., Ciccarelli, M., Rispoli, A., Carrizzo, A., De Vittorio, M., Patrono, L.: Wearable technologies and AI at the far edge for chronic heart failure prevention and management: a systematic review and prospects. Sensors **23**(15), 6896 (2023). https://doi.org/10.3390/s23156896
49. Zhao, Y.F., Chaw, J.K., Ang, M.C., Tew, Y., Shi, X.Y., Liu, L., Cheng, X.: A safe-enhanced fully closed-loop artificial pancreas controller based on deep reinforcement learning. PLoS ONE **20**(1), e0317662 (2025). https://doi.org/10.1371/journal.pone.0317662
50. Papachristou, K., Katsakiori, P.F., Papadimitroulas, P., Strigari, L., Kagadis, G.C.: Digital twins' advancements and applications in healthcare, towards precision medicine. J. Pers. Med. **14**(11), 1101 (2024). https://doi.org/10.3390/jpm14111101
51. Reshad, A.I., Nino, V., Valero, M.: Deep learning-based detection of arrhythmia using ECG signals—a comprehensive review. Vasc Health Risk Manag. **30**(21), 685–703 (2025). https://doi.org/10.2147/VHRM.S508620.PMID:40909176;PMCID:PMC12406999
52. Dey, A., Gupta, G., Sahu, S.P. (2023). Hybrid meta-heuristic based feature selection mechanism for cyber-attack detection in IoT-enabled networks. Procedia Comput. Sci. **218**, 318–327. https://doi.org/10.1016/j.procs.2023.01.014
53. Hizem, M., Bousbia, L., Ben Dhiab, Y., Aoueileyine, M.O., Bouallegue, R.: Reliable ECG anomaly detection on edge devices for internet of medical things applications. Sensors (Basel). **25**(8), 2496 (2025). https://pubmed.ncbi.nlm.nih.gov/40285186/ PMID: 40285186; PMCID: PMC12031393

54. Hermawan, D., Dewi Kansa Putri, N.M., Kartanto, L.: Cyber physical system based smart healthcare system with federated deep learning architectures with data analytics. Int. J. Commun. Netw. Inf. Secur. (IJCNIS) **14**(2), 222–233 (2022). https://doi.org/10.17762/ijcnis.v14i2.5513
55. Wang, R., Lai, J., Zhang, Z., Li, X., Vijayakumar, P., Karuppiah, M.: Privacy-Preserving federated learning for internet of medical things under edge computing. IEEE J. Biomed Health Inform. **27**(2), 854–865 (2023). Available: https://pubmed.ncbi.nlm.nih.gov/35259124/
56. Basco, K.J., Singh, A., Nasef, D., Hartnett, C., Ruane, M., Tagliarino, J., Nizich, M., Toma, M.: Electrocardiogram abnormality detection using machine learning on summary data and biometric features. Diagnostics (Basel). **15**(7), 903 (2025). PMID: 40218253; PMCID: PMC11988324. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC11988324/
57. Sahu, D., Nidhi, Chaturvedi, R., Prakash, S., Yang, T., Rathore, R.S., Alsolbi, I.: Optimizing energy and latency in edge computing through a Boltzmann driven Bayesian framework for adaptive resource scheduling. Sci. Rep. **15**(1), 30452 (2015). https://doi.org/10.1038/s41598-025-16317-6. PMID: 40830214; PMCID: PMC12365131
58. Roy, A., Banerjee, R., Zhang, C.: Robustness and fault-tolerance in cyber-physical healthcare systems. IEEE Trans. Dependable Secure Comput. **20**(1), 91–103 (2023)