Zero Trust Security

A Hands-on Guide

Adam Tilmar Jakobsen

WILEY

Zero Trust Security

A Hands-on Guide

Adam Tilmar Jakobsen

WILEY

Library of Congress Cataloging-in-Publication Data has been applied for:

Hardback: 9781394361090

ePDF: 9781394361113

epub: 9781394361106

oBook: 9781394361120

Cover Design: Wiley

Cover Image: © Soham Pansuriya/Shutterstock

Table of Contents

List of Illustrations

List of Tables

Preface

Zero Trust is not a project, but a new way of thinking about information security. – John Kindervag, author of the founding Zero Trust paper, 'No More Chewy Centers'

The essential paper that started the Zero Trust idea was John Kindervag's 'No More Chewy Centers: Introducing The Zero Trust Model of Information Security'. It was an unfortunate name because how can you run a network if you do not trust anything or anybody? That was not what Kindervag meant by the word Zero Trust. The concept is based on how the military thinks about protecting secrets. Everything requires a 'need to know' basis. If you don't require the information to do your job, you shouldn't have access to it. That's all there is to Zero Trust. This idea of only granting access to what is required should apply to users, servers, network equipment and applications. We grant users access only to what they explicitly need for their work. That whole idea of Zero Trust. This way, we can reduce the attack surface of our environment by limiting access to services and data.

This book offers a practical guide to implementing Zero Trust principles in a simulated environment. The goal is to provide hands-on experience with the technologies and concepts that underpin Zero Trust, allowing readers to understand how to apply these principles in real-world scenarios. The book is structured to take you through the process of setting up a simulated environment using Docker, implementing network segmentation, monitoring and logging network activities, managing identity access, deploying endpoint detection and response solutions, integrating security information and event management systems, identifying and mitigating vulnerabilities, and protecting web applications using web application firewalls.

Acknowledgements

I would like to thank everyone in the cybersecurity community for all the awesome work they share on the internet. Without them, this book would not have been possible.

About the Author

My journey with computers began, as it did for many others, with playing video games. Initially, my goal was to become a game developer, which led me to get a master’s degree in computer science. Although that dream didn’t materialise, I instead joined the Danish Army as a cyber specialist for the army intelligence within the electronic warfare division. Because of the classified nature of my work, I cannot delve into specifics; however, my job generally involved expanding the utility of cyber capabilities within operations, focusing on SIGINT, OSINT and all-source intelligence while also supporting other departments such as HUMINT, PSYOPS and IMGINT. After three years with the army intelligence, I joined Bluewater Shipping, a major Danish shipping company. Initially serving as a solution architect for all internally developed software, my role shifted to information security following a significant cyberattack on the company. As the sole security engineer, I oversaw the entire operation pipeline, including defining and implementing detection rules, incident response and much more. Another three years passed, and the government offered me the opportunity to join as a security specialist. The knowledge I’ve gained throughout my journey is an amalgamation of research, reading white papers, taking courses, taking certifications and taking part in Capture the Flag challenges.

Introduction

I have written this book because I felt there was a lack of material about the more practical side of implementing Zero Trust, with hands-on exercises to help you learn. For this reason, I have included as many practical elements throughout the book. This resulted in the book becoming somewhat of a coursebook. The goal is to help people who are new to the field of cybersecurity. To where they can implement Zero Trust.

This book draws inspiration from The Phoenix Project,[1] as well as Project Zero Trust.[2] What both these books have in common is that they are about a fictional company that goes through a transformation of its information technology (IT) practices during a crisis. However, I found these books lacked a practical component that went into detail on some of the specific technologies that are used to achieve their goal. My goal with this book was to fill that gap by providing hands-on experience with these technologies. Giving the reader a demonstration of how different security controls work in action. Helping you understand how they work and what makes them important. The primary goal of this book is to bridge the gap between theoretical knowledge and practical skills. What this book will not do is make you an expert in Zero Trust, nor would it be possible for any book to do that. This is just the starting point of your journey towards becoming an expert in Zero Trust.

What is it that makes Zero Trust so interesting and confusing at the same time? Zero Trust is a change in how we view security. Traditional models of security rely on a strong perimeter to keep threats outside the network. The problem with this approach is that once an attacker breaches the perimeter, they can freely move within the network with relative ease. This is where Zero Trust comes into the picture. Its goal is to mitigate this by enforcing strict access controls that are continuously verified. And constantly monitoring the network for any threats. Zero Trust emphasises the idea of minimising trust. This involves identifying instances where user and system access is beyond what is necessary for their job.

The way we explore Zero Trust in this book is through a fictional organisation called Juice Factory. This fictional organisation's IT infrastructure will be simulated using Docker containers. I chose Docker as it offers a lightweight and efficient way to create, deploy and manage multiple applications. This makes it an ideal tool for this project. Because it is a simulated environment, it has limitations, including control over the network, lack of network equipment for us to work with, and the absence of identity access management (IAM), and of real users. Containerisation also doesn't fully replicate enterprise network infrastructure components such as physical switches, routers and hardware firewalls.

Despite these limitations, Docker provides a practical and accessible platform to illustrate the key concepts behind Zero Trust. You can find the Docker environment for this book at the GitHub repository.[3] Here you can download the Docker environment we will use throughout this book.

Each chapter will guide you through Zero Trust principles and the implementation of security controls in our Docker environment, giving you step-by-step instructions on how to do so. As we progress through the book, you will learn:

- Set up a simulated environment using Docker
- Implement network segmentation
- Monitor and log network activities continuously
- Challenges of IAM and Using Jump Box to secure access to segmented networks
- Deploy and configure Endpoint Detection and Response (EDR) solutions
- Integrate Security Information and Event Management (SIEM) systems
- Identify and mitigate vulnerabilities using vulnerability scanners
- DevSecOps and Protect web applications using web application firewall (WAF)

Each chapter of the book will build upon the previous one. Upon completion, we will have a completely new environment, incorporating various security controls to defend against threat actors. While this book provides a detailed and practical guide for implementing Zero Trust, it is important to remember that no single resource can cover every scenario or challenge that you will face in the real world. The goal is to provide you with a foundational understanding of things, including some hands-on experience. This means that additional learning and adaptation are necessary to address the specific needs and complexities of your organisational environment. I encourage you to experiment, ask questions, and seek further knowledge as you embark on your journey towards cybersecurity expertise. By the end of this book, you should have a foundational knowledge of what Zero Trust thinking is and the controls we use in Zero Trust and cybersecurity.

Notes

1 The Phoenix Project by Gene Kim, Kevin Behr, and George Spafford, IT Revolution Press, 2013.

2 Project Zero Trust, George Finney and John Kindervag, Wiley, 2022.

3 `https://github.com/Paradoxxs/zero-trust-enviroment`

Chapter 1
Use Case: Juice Factory

> What do you mean, zero trust? To be a successful organisation, we need trust. Yes, but not beyond what is required for them to do their job.

Throughout this book, we will explore the ideas of Zero Trust Architecture (ZTA) using a fictional company that allows us to look at the different concepts of zero trust in a practical and fun way. The corporation we will use is called Juice Factory, which represents a business that comprises both information technology (IT) used for communication and collaboration between employees and customers. And then there is also the operational technology (OT) that controls the production system making the Juice. At the last seminar for board members, they heard about a concept called Zero Trust, which can be used to make any organisation secure. Combined with the increasing risk of a cyber incident. The board of directors has hired you as their first Chief Information Security Officer (CISO). As the CISO, it's your job to successfully adopt the strategies of ZTA to the organisation's IT and OT environment. Making them more resilient against cyber threats and reducing the probability of a significant cyber affecting the business.

Generated with AI using Hugging Face Hub

1.1 Company Profile

The first thing any CISO or security professional should do is to understand the organisation. It's impossible to defend something you do not understand, like why the business makes the decision that they do. With that said, let's get started getting to know the business we will work with. Here are some more details about the Juice Factory, to help clarify what they are all about.

- Name: Juice Factory
- Industry: Manufactory
- Headquarters: Europe
- Employees: 500
- Revenue: $100 million annually
- Sales Distribution:
  - 10% from Business to Consumer (B2C) online sales
  - 90% from Business to Business (B2B) sales

1.1.1 Headquarters

The headquarters of Juice Factory is located in a major European city. With approximately 450 employees located here. It comprises two buildings: one dedicated to the administrative offices of the organisation and the other to the engineering and manufacturing of juice products. All of the data centres for Juice Factory are located on-premises within the headquarters building. This ensures streamlined of IT and OT operations.

1.1.2 Satellite Offices

Juice Factory also operates several smaller satellite offices focused primarily on B2B sales in different countries. Sales personnel mostly staff these offices. With a typical size of around 4–6 employees per office. These people manage local customer relationships and sales activities in their respective countries.

1.1.3 Operational Technology

Juice Factory's OT systems are integral to its manufacturing processes. It automates the flow of production, using machinery control that is connected to a programmable logic controller (PLC) that is integrated with a human–machine interface (HMI), which allows the engineer to control the flow of the production facility with just a press of a button. The engineering team is responsible for monitoring and controlling the OT system. Because the production facility is critical for the business, it is not possible to reboot the system outside the planned maintenance cycles.

1.1.4 Information Technology

For the IT environment, the Juice Factory has its own IT infrastructure. The data centres are all at headquarters. The satellite offices only consist of the minimum network equipment to function and endpoints. Every satellite office is required to have a constant connection to the headquarters network to access the corporate resources. We achieve this using site-to-site VPNs.

1.1.5 Business Consideration

For the design of the security architecture for an organisation, it is important to consider the business processes. When implementing new security controls at Juice Factory, consider the following business factors.

- The continuous production of juice.
- Remote and Mobile Workers: Employees often work remotely or are on the move, requiring secure access to company resources

from various locations and devices.

- Many Business Partners: The company collaborates with numerous partners, necessitating secure communication and data exchange channels.
- Intellectual Property: The Juice Factory holds valuable production and manufacturing trade secrets that require protection from unauthorised access and cyber threats.
- Diverse Endpoint Devices: The company's IT environment includes a mix of endpoint devices running on Windows, Mac and Linux, requiring a versatile security solution.
- Geographic Diversity: With operations across different geographic locations, the Juice Factory must comply with local regulations and meet diverse compliance requirements.

Whatever organisation you are working with, the same processes should be done. Talk with the different department leaders, understand how they work and what unique challenges they are facing. This helps you identify what problems you can mitigate and, most importantly, not introduce new problems for the departments.

1.2 Getting to Know the Business and Finding the Crown Jewels

Security can no longer afford to hide in our cubicle. The goal of security is to allow business processes to be performed in a secure manner. Not only that, we also have to be sure that the security control we put in place does not limit the business. Otherwise you will quickly be faced with shadow IT, and you will no longer have control over where the business data exists. For example, when our sales team needed a file-sharing solution for large technical documents, security initially prohibited cloud storage use. This led to sales representatives creating personal Dropbox accounts for sharing product specifications, inadvertently exposing proprietary data. A good way to mitigate shadow IT is by implementing approved alternatives before restricting tools.

What is required by us is to get to know the business. For that to happen, we have to create a relationship with the rest of the business leadership team. The best way to do this is through communication and collaboration between departments. How do we go about achieving this? The goal should be to identify how security can add value to the organisation. Speak to the different key departments for the organisation. These are often sales and legal. Legal allows you to understand what regulation the organisation must adhere to. And for sales, identify if there are any security concerns from the customers or if the organisation has lost any deals with potential customers because of lacking security. Another example is if sales or legal answers compliance-related questions about information security. You should step up and take this off their shoulder. Not only are you best suited to answer the question, it also creates a positive relationship with the department. The goal is to demonstrate to the other leaders that security is not just a cost centre, and instead is a business enabler. This requires you to speak the language of business, which they know and leave behind all the IT jargon behind, for our fellow security engineers.

The end goal is to understand which business processes makes the business profitable. Including the supporting systems that enables these processes. With that, we should have identified the crown jewels of the organisation. This allows us to prioritise what to focus on first and which system needs greater protection. Let’s look at the Juice Factory and try to understand both the crown jewels and the challenges the organisation is facing. Since this is a fictional business. It is not possible for us to go out and ask the different department leaders for input about the different functions of the business and the problems they are facing. We just have to pretend that we have already done so, and these are their responses.

- Legal: Our Payment Card Industry Data Security Standard (PCI DSS) compliance efforts currently consume approximately 200 working hours yearly. We’re seeing increasing scrutiny from auditors, particularly around our card data environment segmentation. We’ve identified gaps in our GDPR compliance, specifically around data subject access requests and cross-border data transfers. Failure to address these issues may result in potential fines of up to 20 million euro or 4% of annual revenue.

- Production: Our OT environment runs on legacy systems that control specialised juice processing equipment. We cannot reboot the production environment because of the loss of production time to the business. The only time we can perform maintenance and security update are part of scheduled maintenance, with only a single day each year is all that’s allowed for maintenance. We’ve identified three critical vulnerabilities in our SCADA systems, but patching requires a full system shutdown. We store our proprietary recipes digitally, accessible to 15 engineers; however, we lack an access log to audit who accesses the documents. A competitor recently attempted to hire one of our senior engineers, raising concerns about IP protection.

This covers the security concerns of the organisation. For this book, we will not go into how to handle compliance. That could be a book on its own. Instead, let’s focus on the IT and OT environment. What we know so far is that we cannot reboot the OT environment, as the production loss is too high. That limits us to only activities that do not directly touch the production systems itself. This limits us to only modify the network configuration. We should mark the production system as a crown jewels because of its criticality to the organisation. And it should be one of the first systems we will try to improve the protection for. For the IT environment, we need to ensure continuous operation, as any downtime stops sales for contacting potential leads and managing customer relationships.

Now that we have learned a little about the Juice Factory, which we will try to protect throughout this book. In Chapter 2, we will be heading straight into Zero Trust, to get a better understanding of how the strategies of Zero Trust can identify and design the controls that we need to implement.

Chapter 2
Zero Trust

Zero Trust is not a project, but a new way of thinking about information security. – John Kindervag, author of the founding Zero Trust paper, "No More Chewy Centers"

The essential paper that started the Zero Trust idea was John Kindervag's 'No More Chewy Centers: Introducing The Zero Trust Model of Information Security'. It was an unfortunate name, because how can you run a network if you do not trust anything or anybody? That was not what Kindervag met with the word Zero Trust. The concept is based on how the military thinks about protecting secrets. Everything requires a 'need to know' basis. If you don't require the information to do your job, you shouldn't have access to it. That's all there is to Zero Trust 'if you do not need it to do your job, you shouldn't have access to it'. This idea of only granting access to what is required should apply to users, servers, network equipment and applications. We grant users access only to what they explicitly need for their work. That's whole idea of Zero Trust. This way, we can reduce the attack surface of our environment by limiting access to services and data.

When it comes to implementing Zero Trust, there is no product that can make you 'Zero Trust', no matter how much marketing tells you otherwise. Zero Trust is a philosophy, a strategy and a new way of thinking about security. The goal is to evaluate your environment's security state continually. There will never be a presentation saying, 'We have achieved Zero Trust'. There is no need to go out and buy a suite of new tools; most organisations probably already have technology within their environment that can get you a long way down the road of Zero Trust. The only difference from before is that you just have to configure them with the mindset of Zero Trust. Before we can dive into examples, we have to get an understanding of the basic design principles. In this chapter, we dive into the four core design principles of Zero Trust, giving you a method for prioritising and evaluating what controls need to be implemented first and how.

2.1 Why Perimeter Security Is Insufficient

To understand Zero Trust, we have to understand the security architecture that came before, which would be perimeter security. Perimeter security treats outside network connections as untrusted and all inside connections as trusted. The aim is to build a strong fortification, resembling a wall around a castle. The problem is, once you are on the inside, they trust similarly to the Trojan horse. Others like to compare the perimeter design to a hard candy shell on the outside but a scrumptious chocolate centre in the middle. The problem with perimeter design is by default we trust all internal systems. With this model, once an adversary is on the inside of the network, they become automatically trusted. Security controls don't address internal-to-internal attacks. A question I like to ask is 'Do you have the same level of protection for outbound connection to the internet, as you have for inbound connection?' This is a good question to help understand if the organisation is thinking in terms of Zero Trust. Another question you should ask is 'does your organisation have devices on the network whose configuration or software you cannot guarantee?'

I frequently hear of organisations that let internal vulnerabilities persist because they believe these vulnerabilities are safe from exploitation within their internal network. What about getting access to the different systems? Do employees and contractors truly have the least privilege access? Did we restrict contractor access to the contract period? What about your employees? When was the last time you verified that people with privileged access still have a need for this kind of access?

The idea of the perimeter had died a long time ago. We have in our pocket a small computer that is constantly jumping between different networks that we do not have control over. Laptops are also being removed from the office and connected to different networks like the airport Wi-Fi. Then there was the cloud, and the perimeter completely broke down. We now have a critical system for the organisation running on someone else's hardware, transmitting data over a network we have no control over. The one thing everyone knows is that the environment will, at some point, get compromised. The problem is our architectures do not reflect this. Every part of the environment requires the same level of rigorous protection. We can no longer rely on perimeter architecture for protection. We have to find new design principles, which we have used to help us design an environment that we can protect.

2.2 Zero Trust: Principles

With an understanding of what came before, we can now head into understanding Zero Trust. Good for us is that Kindervag also includes four design principles in his Zero Trust paper. This will help to get a better understanding of what Zero Trust is and how it can apply to our environment.

1. Understanding Business Outcomes: The goal is to understand how the business functions before making any organisational changes. It is critical to have a foundation of understanding of the organisation's business objectives. By asking, 'What is the business trying to achieve?' This ensures that the security measures align with the business objectives. The goal should be to help better protect the organisation, without limiting its ability to make a profit. Even better if your security program can be a business enabler and drive additional revenue. One requirement of understanding the business is that you have identified the crown jewels of the organisation. What are the business processes that are driving the profits of the organisation? These systems and processes need to be protected better than anything else in the organisation, or the business will not survive.
2. Designing with the Mindset of 'from the Inside Out': Starting with the crown jewels that are required for the business to function, this means looking at the full picture of the organisation, including data, applications, assets and services (DAAS). This approach builds your security architecture and controls around the crown jewels. The following are the architectural steps when designing security controls.

- Identify Protect Surfaces: I like to think of protecting surfaces in terms of business processes. Take a business process and determine what DAAS used to facilitate them and group them together into a single protection surface. A common question is, what if a system belongs to multiple business processes? Then make it into its own protected surface.
- Map Out Interactions: Understand how the different protected surfaces interact with one another.
- Design Protective Measures: Implement security controls starting from these protective surfaces and moving outwards, ensuring comprehensive coverage.

3. Determine Who or What Needs Access: Determine who needs to have access to a resource to get their job done. It is very common to give users and systems more access than it requires to perform their job. This can lead to access to sensitive data, which the user has no business reason to access. These are the steps of implementation:
   - Role-based Access Control (RBAC): Define the roles needed within your organisation and have them assigned to users based on job requirements.
   - Continuous Review: Regularly review access rights to ensure they remain aligned with the current job functions of the employees.
   - Automated Provisioning: Use automated tools to manage and enforce access controls efficiently.
4. Continuous Monitoring and Logging: The process of continuously watching and logging the activities of the organisation’s environment, with the goal of identifying and responding to any malicious activities happening. The act of timely responding is critical. It does not matter how good your monitoring systems are, if no one is looking at them.
   - Deploy Monitoring Tools: Use tools that provide visibility into network traffic and user activities.

- Log Analysis: Continuously analyse logs for signs of unauthorised access or other anomalies.
- Incident Response: Establish procedures for responding to detected threats based on log data.

2.3 NIST SP 800-207: Zero Trust Security Architecture

We cannot talk about Zero trust without also talking about the National Institute of Standards and Technology (NIST) and its publication on Zero Trust.[1] The following is how NIST SP 800-207 defines Zero Trust.

> Zero trust (ZT) provides a collection of concepts and ideas designed to minimise uncertainty in enforcing accurate, least privilege per-request access decisions in information systems and services in the face of a network viewed as compromised. Zero trust architecture (ZTA) is an enterprise's cybersecurity plan that utilises Zero Trust concepts and encompasses component relationships, workflow planning, and access policies. Therefore, a Zero Trust enterprise is the network infrastructure (physical and virtual) and operational policies that are in place for an enterprise as a product of a Zero Trust architecture plan.

This definition set out by NIST is hard to understand. So, let's try to rewrite the definition to make it easier to understand what is going on. The rewritten version could be something like this:

> Zero Trust is a cybersecurity approach that assumes by default to trust no one inside or outside your network. This means the system verifies the identity and access of every person and object attempting to access a resource, granting only the minimum access necessary for their task.

NIST SP 800-207 has also created seven tenets to follow when implementing Zero Trust.

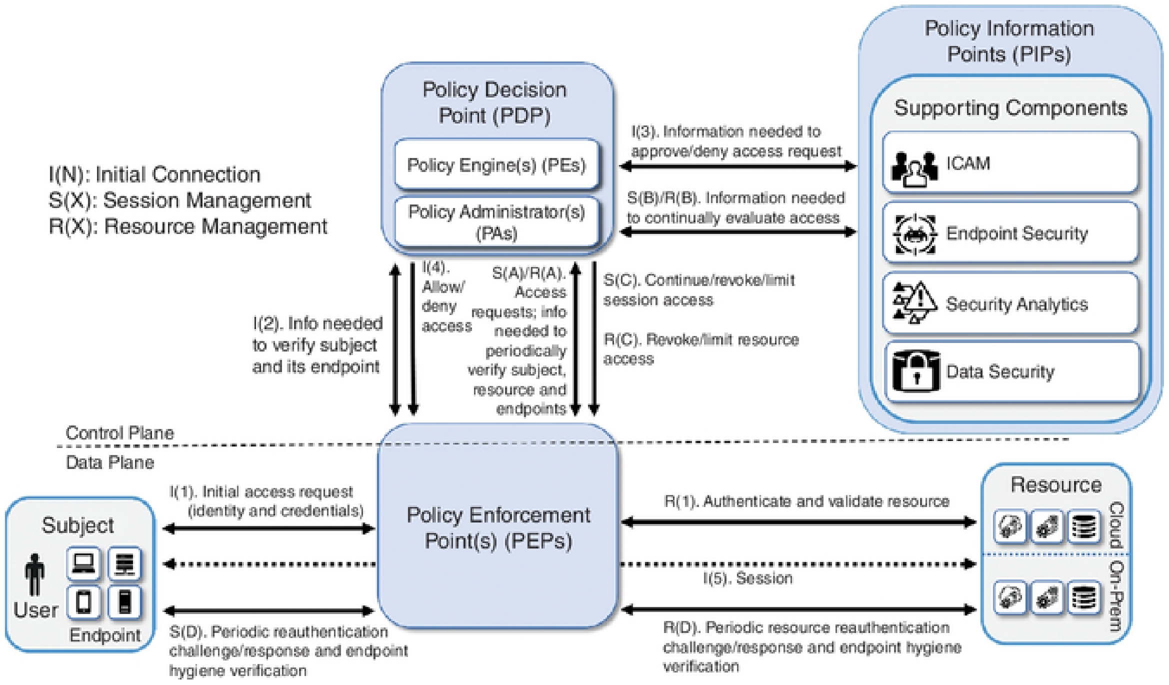
1. All data sources and computing services are considered resources.

2. All communication is secured regardless of network location.
3. Access to individual enterprise resources is granted on a per-session basis.
4. Access to resources is determined by dynamic policy – including the observable state of the client, application/service and the requested asset. It may include other behavioural and environmental attributes.
5. The enterprise monitors and measures the integrity and security posture of all owned and associated assets.
6. All resource authentication and authorisation are dynamic and strictly enforced before access is allowed.
7. The enterprise collects as much information as possible about the current state of assets, network infrastructure and communications and uses it to improve its security posture.

There are also six assumptions from a network perspective that every Zero Trust project will need to address:

1. The entire enterprise private network does not qualify as an implicit trust zone.
2. Devices on the network may not be owned or configurable by the enterprise.
3. No resource is inherently trusted.
4. Not all enterprise resources are on enterprise-owned infrastructure.
5. Remote enterprise subjects and assets cannot fully trust their local network connection.
6. Assets and workflows moving between enterprise and non-enterprise infrastructure should have a consistent security policy and posture.

The NIST Zero Trust model says one should shift from the traditional perimeter-based security strategies to a more granular approach, where they verify each access request. This change is necessary to address the evolving landscape of cyber threats, where attackers are constantly bypassing traditional defences and exploiting vulnerabilities within the network itself to gain full control of our systems. When it comes to implementing the idea behind NIST SP 800-207, you are quickly going to be hit by a brink wall. That is because there are no references for any specific technologies, everything is at a very high level. In Figure 2.1, you can see the NIST Zero Trust reference architecture. Looking at it, it is not surprising that people still have a difficult time understanding how to implement Zero Trust into their own environment.



Policy Information Points (PIPs)
Supporting Components
ICAM
Endpoint Security
Security Analytics
Data Security
Policy Decision Point (PDP)
Policy Engine(s) (PEs)
Policy Administrator(s) (PAs)
I(N): Initial Connection
S(X): Session Management
R(X): Resource Management
I(3). Information needed to approve/deny access request
S(B)/R(B). Information needed to continually evaluate access
I(2). Info needed to verify subject and its endpoint
I(4). Allow/ deny access
S(A)/R(A). Access requests; info needed to periodically verify subject, resource and endpoints
S(C). Continue/revoke/limit session access
R(C). Revoke/limit resource access
Control Plane
Data Plane
I(1). Initial access request (identity and credentials)
Subject
User
Endpoint
Policy Enforcement Point(s) (PEPs)
S(D). Periodic reauthentication challenge/response and endpoint hygiene verification
R(1). Authenticate and validate resource
I(5). Session
R(D). Periodic resource reauthentication challenge/response and endpoint hygiene verification
Resource
Cloud
On-Prem

Figure 2.1 NIST SP 800-207 Zero Trust reference architecture.

The problem is it does not reference any specific technologies, e.g. the policy decision point(s) or pdp could just as well have been referencing Microsoft Active Directory. I would much prefer that they have given an example of how to implement Zero Trust using specific technologies that most information technology (IT) engineers have some experience with. That would make it easier for people to translate the idea of Zero Trust to their own network. That is one of the reasons why I am writing this book, hopefully to help people transition from knowing about Zero Trust to implementing it.

2.4 Implementing Zero Trust

When it comes to implementing Zero Trust, organisations must adopt a mindset that treats every connection, device and user as compromised. This approach not only requires robust identity and access management (IAM) governance but also mandates continuous monitoring and assessment of network traffic and user behaviour. We aim to grant the least privilege for the shortest necessary time, thus reducing the risk of unauthorised access or lateral network movement, while watching the systems and user to ensure they are following expected behaviour. John Kindervag came up with the Five-Step Zero Trust Design Method to make Zero Trust implementation simpler.

- Define the protected surface.
- Map the transaction flows between protected surfaces.
- Architect the Zero Trust environment.
- Create Zero Trust policies.
- Monitor and maintain.

The idea is to make implementing Zero Trust a little more practical by taking the big problem of implementing Zero Trust in the entire environment and breaking it into smaller problems. Allow us to focus on bringing that single element closer to Zero Trust; afterwards, we can then address the next attack surface. Continue this process until you have examined all the organisation's protected surfaces. Once done, you can then start all over again. It is not possible to do everything at once. There are too many elements to consider. You might think, well, that logically, but what protected surface should I then start with? To get feel for the Zero Trust, Kindervag recommends starting with a learning project, which is protected surfaces that are not critical for the business. This helps prepare the team for success in a low-risk environment. From there, move granularly up towards the crown jewels of the organisation. The process should look like a wave as shown in Figure 2.2.

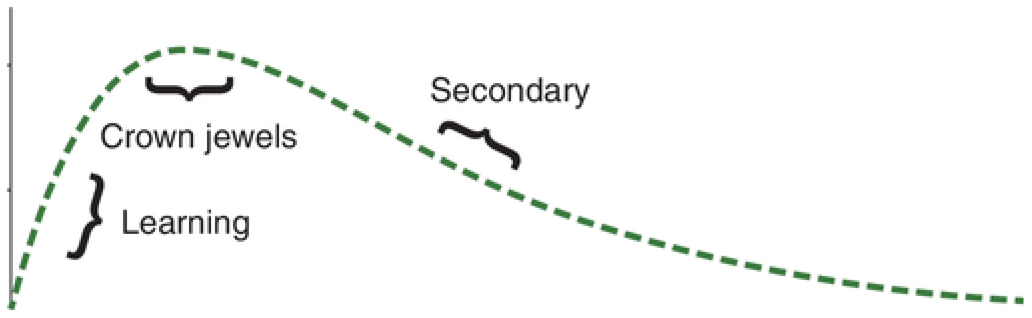


Figure 2.2 John Kindervag – Zero Trust learning curve.

John Kindervag calls this the Zero Trust Implementation Curve. The *x*-axis shows the sensitivity or criticality of the protected surface. The *y*-axis shows the progression of the Zero Trust journey, continuing as long as the organisation exists. Start by defining the transaction flows for the protected surface, and next create security controls to minimise the attack surface to what is absolutely necessary for the business to continue. The peak of the graph is the crown jewels, the business' most important protected surfaces; from there are the second most important protected surfaces, and as we move further down the wave, we have the least important protected surfaces based on criticality. The idea is to split the project into phases. Where in the beginning, making mistakes will have as small impact on the organisation as possible.

2.5 Why Zero Trust Projects Fail

The idea of converting our old perimeter network into Zero Trust can appear both unachievable and expensive. We think that to achieve Zero Trust, we have to throw everything out, and start anew, buying the most expensive technology. It is easy to think that just buying the newest technology will accomplish the objective. This type of thinking fails because of the belief that just buying technology and flip a switch and the system will manage itself. What goes wrong is that we do not allocate enough resources in terms of people and processes to manage these new systems.

Many organisations have the 'two-guys-and-a-dog' approach to IT. They operate the routers, security stack, printers and much more. Now we want them to manage the organisation's Zero Trust strategy as well.

Zero Trust is requires the whole organisation to be a part of the transformation. Deciding which employees get access to what resources is not a decision we want to sit with the two guys sitting in the basement team. That's a decision, something that should be addressed at the senior levels of your organisation. Even in a small- to medium-sized company, defining access policy is a business decision, not an IT decision. It is something that should happen in collaboration with IT, business partners and HR.

Try to instead of thinking of Zero Trust, as a journey on the never-ending path of improvement, something to pursue every day. There are a million things we could do on that Zero Trust journey. Focus on the things we can do right now, with technology and personnel that we already have. Allowing us to close the digital holes in our network. Start by identifying the project surface that will have the highest impact on your environment first, while still keeping the cost low. These can be things such as logical and micro-segmentation, asset management and IAM. The best time to begin is now, so start improving your identity management program with the least privilege, if you haven't already add two-factor authentication onto the road map, and start limiting what action systems can do to the bare minimum of what is required.

Remember, you can apply the Zero Trust principle to much of the technology you already own. All that is required is looking at things differently; take Windows as an example. It already has all the tools built in to follow the Zero Trust principles. A common trust we have in our system is allowing the user to execute any problem they want on their endpoint. To follow the principle of Zero Trust, we should only allow executables necessary for the user's job. Using AppLocker can achieve this. It is an application that allows listing policies that enforce that only predefined executables can run on the system. Without imposing a monetary cost on the organisation, we have already taken a step towards zero trust.

2.6 Applying Zero Trust to Juice Factory

Let’s make this a little more practical by designing a Zero Trust project for Juice Factory. The first step is to understand the business; we should ask ourselves what is Juice Factory trying to achieve? Let’s look at the data of the organisation. We know that the business sells juice that is made within the factory, and that 90% of the profits come from B2B and the rest from B2C. With this information, we know that the existence of the business relies on the ability to produce juice. If the production environment went down, then 100% of the profits would disappear. That makes production our most critical business process for the organisation. Remember, organisations can have multiple processes that are critical. The next crown jewel would be the business process that allows us to sell and ship the juice to our B2B customers. Finally, we have the process of selling to B2C customers, which in this case is the company’s web store. I would classify these three business processes as business critical for the organisation. This would include the DAAS that facilitates these processes that are the crown jewels we need to be protected.

1. Define Business Outcomes: Juice Factory is a manufacturing business that produces juice products for its customers. It means the continuous operation of the production facility is critical for the business to succeed, making the factory one of the crown jewels of the organisation.
2. Design from the Inside Out: Starting with the crown jewels of the Juice Factory. We know that the production facility is what we should start with. Design with that in mind and try to understand how it interacts with the rest of the organisation and possible external systems. In our scenario, the factory system only communicates with the engineer’s endpoint device. From here, we can identify what controls need to be in place to reduce the probability of a cyber event could occur.
3. Determine Who Needs Access: Follow the principle of need to know and determine who needs access based on their job role. Only factory engineers at the Juice factory need access to the system.

4. Inspect and Log All Traffic: Deploy monitoring tools to log access to the facility and set up alerts for unauthorised access activities, including monitoring all traffic coming and going to the system.

This was just a quick example of how the design principle of Zero Trust can apply to a single business process. The real world involves many more moving parts to consider. That is why I recommend taking one business process and starting with some easy and noncritical systems first to get a feeling for using the design principles.

Now, we understand the business, and what it is trying to achieve. The next thing that we need to do is to identify the protected surfaces of the organisation. Defining protected surfaces is a fine art; you should not go so granular that you might as well be looking at the network configuration, or so simple that it provides no value; the way I link to break it down is by the processes that support the organisation, e.g. I break down the operational technology (OT) environment into a single protected surface instead of one for human–machine interface (HMI) and another for programmable logic controller (PLC). For Juice factory, we have the OT environment for producing the juice, and then there is the web application where we sell to customers. People also need to authenticate themselves using IAM. We also have all the basic applications required to run a company, such as email, customer relationship management, wireless and access to the internet. Then, there are also some basic security elements, such as physical security and backup. This is of course not an exhaustive list of all the protected surfaces, just enough to get a general idea of the concept. Now we have defined some different protected surfaces we can work on. The way I like to work with protected surface is using a table like Excel or similar software. In Table 2.1, you can see an example of juice factory protected surfaces. In the table, we define the protected surface along with its description and the systems that are part of it.

TABLE 2.1

Protect surfaces of Juice Factory.

Protect surface	Description	Systems
OT	Product environment	PLC, HMI
Web store	B2C sales platform for juice	JS, Express, SQLite
IAM	Access management	Active directory
Physical security	Building security	Camera, security gate
Backup	Secondary data storage	Backup tape
Wireless	Corporate wireless network	Cisco Meraki
Email	Email service	Office 365
Internet	The public internet	
Customer relation management (CRM)	Customer management system	

Know that we have defined the protected surfaces. The next step is to understand how the different protected surfaces interact with one another. Kindervag proposed visualising this information using a transaction map. It is basically a matrix, with protected surface as rows and columns. With each cell coloured based upon their need for data exchange, using the colour light grey to show there is a data exchange between the protected surface. We do not care about the type or the direction connection in the transaction map; the goal is to keep it as simple as possible. Table 2.2 shows the Juice Factory transaction map.

TABLE 2.2

Initiate transaction map of Juice Factory.

	CRM	IAM	OT	Physical security	Web store	Backup	Wireless	Email	Internet
CRM	■				■	■			
IAM	■	■	■	■		■	■	■	
OT			■					■	
Physical security		■		■					
Web store					■	■			■
Backup					■	■		■	
Wireless		■					■		■
Email		■			■	■	■	■	■
Internet					■		■	■	■

The good thing about these two tables is that we can create both of them in a tool we all know, Excel. With these two tables, we have defined both the protected surfaces and their interactions with one another. From here, it is up to the system administrators and network engineers to implement the architecture set out by the transaction map into the organisation's IT environment using the necessary controls. And you can use the documentation to verify that the system follows the agreed-upon architectural design. For completeness shake, let's iterate over the process one more time, just to make it clear.

1. Identify Protected Surfaces: I like to think of protected surface in terms of business processes. Determine what DAAS is most critical to your business operations.
2. Map Interactions: Understand what elements need to interact for the protected surface to function.

3. Design Protective Measures: Implement security controls starting from these protected surfaces outward, ensuring comprehensive coverage.

This process continues until the organisation’s end, with the goal of continually eroding trust within the system.

2.7 Processes and Procedures

In implementing Zero Trust Architecture, one of the greatest challenges security professionals face is the limited direct control over the environments they protect. While technical controls are essential, they alone cannot ensure security. This is where well-defined processes and procedures become critical – they serve as the operational framework that transforms Zero Trust from a theoretical concept into practical security measures. These processes and procedures act as the connective tissue of Zero Trust Architecture, ensuring consistent implementation across all aspects of the organisation’s security posture. For example, when an employee requests access to a resource, a properly defined process ensures that the request follows the Zero Trust principle of ‘verify explicitly’ by requiring approvals, documentation of business needs and implementation of security required. The lack of structured processes will lead to inconsistent application of security controls, resulting in a flawed security implementation and allowing threat actors to bypass the security control. Furthermore, processes and procedures provide the governance framework necessary for maintaining Zero Trust principles over time. These processes and procedures dictate security decision-making, documentation and review, thus ensuring consistent application of the ‘never trust, always verify’ philosophy throughout the organisation. These aren’t merely administrative guidelines – they are critical operational elements that translate Zero Trust principles into daily security practices, from access management and system configuration to incident response and security monitoring.

The problem is that people are quick to go back to their old ways. It’s not enough to set guidelines for how things should be done. Often, we expect people to read and understand the procedures, and then act accordingly. There needs to be a way to verify the procedures are actually being followed. This is often done by either auditing the systems, e.g. verifying that servers are setup to the agreement upon specification. There are two things you are on the lookout for when auditing. That the processes are being followed, but equally important that they are working as intended; there is no point in following a procedure that is not fulfilling its purpose. That would be a waste of time for both you and the people required to follow the process.

Automation can play a significant role in streamlining processes. I favour automating anything possible because it eliminates human error and guarantees consistent process adherence. This is a standard practice in DevOps, where the developer uses automation to do repetitive tasks, like testing and deployment of software. Having the system do the steps itself not only removes the possibility of human error but also makes the processes much faster and more effective. Implementing processes and procedures is not without its challenges. These are not only technical challenges, but organisational challenges. There is often resistance to any changes both from the engineers and getting sign-off from senior leadership. There are also technical complexities and resource constraints. We need a plan to overcome these obstacles. This plan should involve a step-by-step approach, clear communication about the benefits and consistent communication with everyone involved. Despite these challenges, the benefits of well-implemented Zero Trust processes are substantial. Having well-defined processes and procedures is key to maintaining a strong security posture.

2.8 Summary

Zero Trust Architecture represents a fundamental transformation in cybersecurity strategy, requiring both technical implementation and organisational change. Through the Four Zero Trust Design Principles outlined in this chapter, organisations can begin their journey by taking specific, measurable steps: aligning security controls with business objectives, implementing granular asset protection, establishing strict access controls and deploying continuous monitoring systems. The path to Zero Trust implementation begins with practical action. Organisations should start by conducting a thorough assessment of their current security posture and identifying their critical assets. This baseline assessment provides the foundation for developing a phased implementation plan that prioritises protecting crown jewels while maintaining business operations. Security teams should focus on quick wins that show value – such as implementing multi-factor authentication or establishing micro-segmentation for critical systems – while building towards comprehensive Zero Trust coverage.

Initially, when Zero Trust became mainstream, people hailed it as the future of network security. However, despite the widespread recognition of its importance. Many organisations still struggle to implement it effectively. The challenge lies in the practical examples; the original concept, outlined in Kindervag's foundational paper, lacked detailed guidance on execution. This book aims to fill that gap by providing the how to section for implementing Zero Trust in a real-world context.

The transition to Zero Trust demands ongoing commitment and collaboration across the organisation. Security leaders should establish clear metrics for success, maintain regular communication with stakeholders and continuously assess and adjust their implementation strategy. Start by identifying one critical business process, apply the Zero Trust principles we've discussed and use that success as a blueprint for expanding across your organisation. Remember: Zero Trust is not a destination but a continuous journey of security improvement, requiring persistent evaluation and adaptation to meet new threats as they emerge. Hopefully, through the hands-on practice throughout this book, you will gain the skills and insights necessary to implement Zero Trust effectively within your organisation.

Note

1 https://doi.org/10.6028/NIST.SP.800-207

Chapter 3
Docker

Containerisation has revolutionised how we develop, deploy and manage applications. At the heart of this revolution is Docker, an open-source platform that has become the industry standard for containerisation technology. Throughout this book, we'll use Docker to create isolated environments that simulate real-world infrastructure, allowing us to explore and implement Zero Trust principles in a controlled setting. Docker enables us to package applications and their dependencies into standardised units called containers. Each container runs as an isolated process on the host operating system, sharing the kernel but maintaining strict boundaries for resources, filesystems and network access. This isolation provides several key advantages for our security testing environment: we can quickly create and destroy compromised systems, test security controls without affecting production environments and ensure consistent behaviour across different testing scenarios. Unlike traditional virtual machines that require a full operating system for each instance, containers are lightweight and start up in seconds. They share the host system's kernel while maintaining isolation through Linux namespaces and control groups (cgroups). This architecture makes Docker particularly well-suited for our purpose of creating multiple interconnected services that mirror real-world infrastructure.

The host systems limit these containers. Only one container can use a specific port. That means if you have multiple web servers running, they cannot all be listening on port 80 at the same time. There is also the CPU architecture of the host system you have to think about. If you are running an Advanced RISC Machines (ARM) processor, I cannot be sure that the environment will work on your machine. Here are some of the key benefits of Docker:

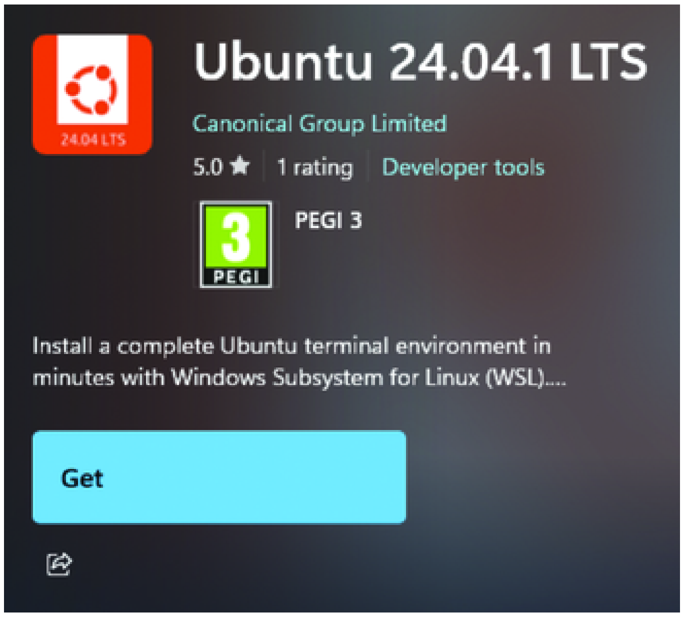
- Isolation: Containers encapsulate an application and its dependencies, ensuring consistent behaviour regardless of the host environment.

- Portability: Containers can run on any system that supports Docker, making it easy to move applications between the different stages of development.
- Efficiency: Containers share the host OS kernel, making them more lightweight and efficient than traditional virtual machines.

Docker works on Windows using a feature built into Windows called Windows Subsystem for Linux (WSL). It is a compatibility layer for running Linux natively on Windows. WSL allows users to use a Linux distribution alongside their Windows OS without the overhead of a virtual machine. All that is required to install WSL is executing a single command.

```
wsl --install
```

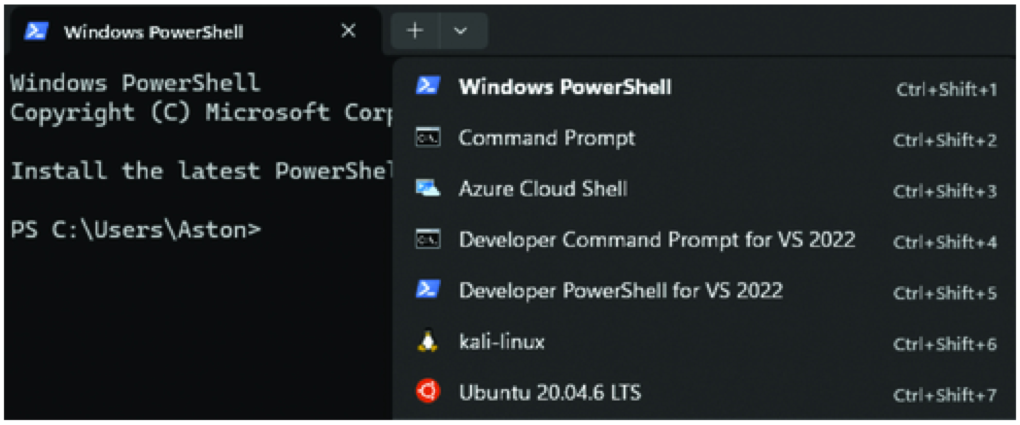
With WSL installed, you can install a Linux distribution. You can do this by going to the Microsoft store and searching for one of the many Linux distributions, e.g. ‘Ubuntu’, as shown in Figure 3.1.



Install a complete Ubuntu terminal environment in minutes with Windows Subsystem for Linux (WSL)....

Figure 3.1 Microsoft store.

Once you install the Linux distribution, you can interact with the Linux system using the terminal program, as shown in Figure 3.2.



```
Windows PowerShell
Copyright (C) Microsoft Corp

Install the latest PowerShe

PS C:\Users\Aston>
```

Windows PowerShell
Windows PowerShell Ctrl+Shift+1
Command Prompt Ctrl+Shift+2
Azure Cloud Shell Ctrl+Shift+3
Developer Command Prompt for VS 2022 Ctrl+Shift+4
Developer PowerShell for VS 2022 Ctrl+Shift+5
kali-linux Ctrl+Shift+6
Ubuntu 20.04.6 LTS Ctrl+Shift+7

Figure 3.2 Windows terminal.

3.1 Installing Docker on Windows

You do not have to go through the process of installing WSL to install Docker. It should install and enable WSL as part of the Docker installation process.

Let's go through the step-by-step process of installing Docker on Windows.

1. Download Docker Desktop:
   - Go to the Docker Desktop download page.[1]
   - Click 'Download Docker Desktop for Windows'.
2. Install Docker Desktop:
   - Run the Docker Desktop installer.
   - Follow the installation instructions, accepting the licence agreement and selecting the default options.
   - Docker Desktop will autoconfigure itself to use WSL 2 as the backend.
3. Start Docker Desktop:
   - Once the installation is complete, launch Docker Desktop from the Start menu.

- Docker Desktop will start and may prompt you to enable additional settings for integration with WSL 2. Follow the prompts to complete the setup. You may need to reboot your computer to complete the setup.

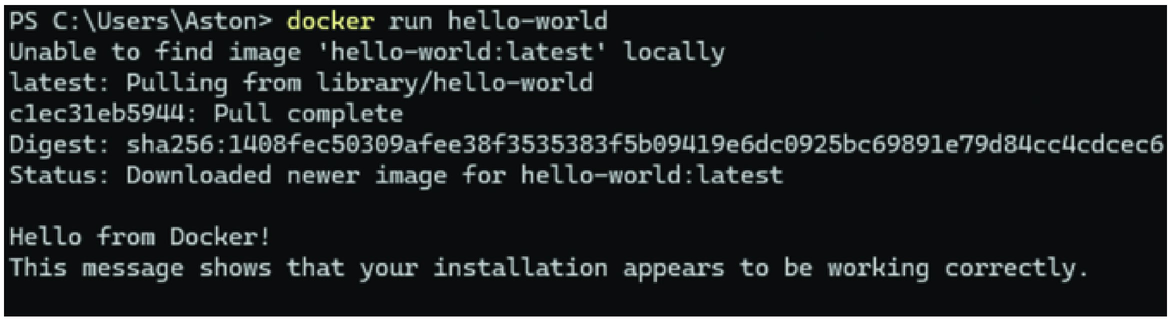
That should be it. You are now ready to run your first Docker container. To do so, follow these steps:

1. Start up Docker Desktop. It should be present either at the desktop or start menu.
2. Open the terminal.

With the terminal window open, use the following command to run the hello-world container:

```
Docker run hello-world
```

This command downloads the hello-world image (if not already downloaded) and starts a container using the hello-world image. Upon container startup, a message confirming Docker's proper installation and function will appear, similar to Figure 3.3.



```
PS C:\Users\Aston> docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:1408fec50309afee38f3535383f5b09419e6dc0925bc69891e79d84cc4cdcec6
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Figure 3.3 Docker hello-world.

3.2 Handling Containers with Docker

You have now installed and run your first Docker container. In this section, you will learn how to use Docker commands to open a shell inside containers, stop containers, publish ports and more.

We are going to start off with something simple, like running an Ubuntu container, which is often the base for more advanced containers. To make it a little more interesting, we are going to get an interactive shell inside the container. Start by running the following command to start up an Ubuntu container:

```
Docker run -it ubuntu bash
```

This command will fetch the latest version of Ubuntu container image. The argument '-it' is used to run the container in interactive mode, together with the bash command giving you shell access to the container. Without the '-it', the container would execute bash and then just quit immediately. The output of the command should look something like this:

```
PS C:\Users\Aston> Docker run -it ubuntu:latest bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
ff65ddf9395b: Pull complete
Digest: sha256:99c351...
Status: Downloaded newer image for ubuntu:latest
root@851cda543a96:/#
```

You can now run any command or software you want inside the container. To exit the container, type the command 'exit' and you will quit the container. If you were to run the command again, it would spin up a new container using the same image. Any changes you made in the first container would not be reflected in the second container. When it comes to viewing what containers are currently running. You can use the command 'Docker ps'; it will output a list of all the running containers on the system. The following is an example of an output you should get with the Ubuntu container running:

```
CONTAINER ID   IMAGE            COMMAND   CREATED
    STATUS          PORTS      NAMES
61b37d55af41   ubuntu:latest    "bash"    59 seconds ago
    Up 59 seconds              crazy_shirley
```

Notice how the system generated the container's name, which is not descriptive. Using the argument '–name name_here' you have the option to give your container a name that is more descriptive of its purpose. For stopping or killing a running container, the 'Docker kill' command can be used. This doesn't mean it has been removed from Docker. You can use the command 'docker ps -a' to view all containers on the system, even those that aren't running.

```
PS C:\Users\Aston> Docker ps -a
CONTAINER ID   IMAGE           COMMAND   CREATED
    STATUS                       PORTS     NAMES
61b37d55af41   ubuntu:latest   "bash"    9 minutes ago
    Exited (137) 4 seconds ago
crazy_shirley
851cda543a96   ubuntu:latest   "bash"    14 minutes ago
    Exited (127) 9 minutes ago             funny_tharp
```

To remove a container from the system, the command 'Docker rm container_name' should do the trick. To remove multiple containers from the system at once, you can run the command 'Docker container prune' to remove every non-running container. The large string of numbers you see when running the prune command is the container id, which is given to them at random at the moment they start up.

You have currently only been running blank Ubuntu containers; as you progress, you will run more complex containers, with software running inside, that often requires one or more ports to be exposed for you to access the application from the host system. That requires you to expose a port for the container to use. A network port is a numbered socket that is used by the computer to sort out what network traffic should go to which application. Many of the default protocols use predefined ports, such as port 80 for HTTP and 443 for HTTPS communication. To expose a port to a container. The container maps the port when it starts. The '-p host_port:container_port' argument is used. For example, the argument '-p 5000:80' maps the host system port 5000 to container port 80. The following is an example of Docker run command:

```
Docker run -p 8080:80 httpd:latest
```

This command will download and run the latest version of httpd from Docker Hub.[2] The '-p' argument tells Docker to expose and map port 8080 on the host to port 80 inside the container. To visit the web server available inside the container, go to http://localhost:8080.

With some containers, you want to expose part of the host filesystem to the container. It allows you to store data in more recent ways. This is important for any database container that you want to run. There is not much usage of database if the data stored inside does not persist across restarts. To ensure persistence after the container stops, we must mount a host system folder into the container in these instances. We do this using the '–mount' argument. It tells Docker you want to mount a volume to the container. Here is an example of how to do it.

```
Docker run -t -i --mount type=bind,src=/data,dst=/data
    busybox sh
```

The argument after the '–mount' tells Docker how and what you would like to mount the volume. Type specifies how the mounting should be done. Here, there are two types, 'bind' and 'volume'. The 'bind' settings allow you to specify where on the host system you would like the data to be stored. With 'volume' Docker will store the data in a location specified by Docker. 'Src' specifies where on the host filesystem to mount into the container, while 'dst' is the location inside the container, where the folder should be mounted to. Docker also offers volumes, which exist only within the Docker environment. The process of mounting a Docker volume to a container is very similar. The following is an example of how to mount a Docker volume to the folder '/app' inside the container:

```
  Docker run -d --mount source=vol,target=/app
nginx:latest
```

This should be everything you need to know to get started with Docker. There are still many things to learn about Docker, but this should be enough to get you started using Docker.

3.3 Docker Compose

While Docker is great for managing single containers, more complex applications often comprise multiple interconnected services. Docker Compose[3] is a tool that simplifies the orchestration of such multi-container Docker applications. With Docker Compose, you can define all the services you need in a single file and manage them all together. Docker Compose comes pre-installed with Docker Desktop for Windows. If, for any reason, it is not available, you can install it separately by following the official installation instructions.[4]

3.3.1 Understanding the Docker-compose.yml File

Instead of using command lines to define the container that needs to run. Docker Compose instead uses a Yet Another Markup Language (YAML) file that allows you to define multiple containers. This YAML file is a human-readable data format that's easy to write and understand. Here's a breakdown of a basic 'Docker-compose.yml' file to get an understanding of how it works.

```
services:   #Define that we are talking about services
  web:   #Name of the first service
    image: nginx:latest   #Specify the Docker image to
use
    ports:
    - "8080:80" #Map port 8080 on the host to port 80
of the container
    volumes:
    - ./html:/usr/share/nginx/html   #Mount the local
directory to the container
  database:   #Name of the second service
    image: mysql:latest
    environment:
      MYSQL_ROOT_PASSWORD=example   #Set environment
variables
      MYSQL_DATABASE=exampledb
    volumes:
    - db_data:/var/lib/mysql   #Define a named volume
for persistent storage
volumes:   #Defines named volumes for persistent
storage
  db_data:
```

Luckily Docker has good documentation on the different parameters within Docker-compose.yml.[5] To start a multi-container application with Docker Compose, navigate to the directory that contains your 'Docker-compose.yml' file and execute the following command:

```
Docker-compose up
```

This command will start all the services defined in your 'Docker-compose.yml' file. To stop the services, use:

```
Docker-compose down
```

3.4 Summary

With this quick introduction to Docker, you should now have enough information to understand and edit the Docker Compose configuration file as we proceed through the rest of the book. Remember that if you face any problems, Docker has great documentation[6] on its usage.

Notes

1 `https://www.Docker.com/products/Docker-desktop`

2 `https://hub.Docker.com`

3 `https://docs.Docker.com/compose/`

4 `https://docs.Docker.com/compose/install/`

5 `https://docs.Docker.com/compose/compose-file/`

6 `https://docs.Docker.com/`

Chapter 4
Initial Design of the Juice Factory Network

> The first 90 days goes with trying to understanding your team and the different leaders to help you understand how security can help improve the business. That is not all. You also have to understand the environment we will work in.

Let's get started with understanding the environment we will work in. In this chapter, we will discuss the initial design of the Juice Factory environment. The environment we will explore throughout this book is a very simplified version of an organisation's. This environment helps us understand the foundational concepts of cybersecurity.

4.1 Services and Docker Images

The first thing we should look at is the Docker configuration that we will work with. It will be the basis for this environment. It defines the services and applications available in the initial environment design of Juice Factory.

```
services:
  juice:
    image: bkimminich/juice-shop
    container_name: webapp
    ports:
      - 8080:3000
  bottle-filling_plc:
      image: paradoxxs/virtuaplant:plc
      container_name: bottle-filling_plc
      ports:
        - 5020:5020
        - 3001:3000
  bottle-filling_hmi:
      image: paradoxxs/virtuaplant:hmi
      environment:
        - PLC_SERVER_IP=bottle-filling_plc
        - PLC_SERVER_PORT=5020
      ports:
        - 3002:3000
      depends_on:
        - bottle-filling_plc
  vuln_samba:
    image: vulhub/samba:4.6.3
    container_name: samba
    tty: true
    volumes:
    - ./samba/smb.conf:/usr/local/samba/etc/smb.conf
    - ./samba/data:/storage
    ports:
    - "4450:445"
  postgres:
    image: vulhub/postgres:10.7
    container_name: postgres
    ports:
    - "5432:5432"
    environment:
    - POSTGRES_PASSWORD=postgres
networks:
 default:
   name: flat
```

This can be quite a lot of information if this is the first time you are looking at Docker Compose configuration file, which may feel overwhelming. Let's go into more detail about each container that is defined in the configuration file.

1. Juice Shop (Web Application)
    - Image: bkimminich/juice-shop
    - Description: Juice Shop is the web shop of Juice Factory that handles their B2C sales. The application is based on Juice Shop, which is an intentionally vulnerable web application used for security training. The goal of the application is to simulate an e-commerce site that is vulnerable to the most common attacks.
2. Bottle Filling PLC
    - Image: paradoxxs/virtuaplant:plc
    - Description: This simulates a PLC used in the production line for controlling the juice bottle filling process. It is essential for managing the operations in the Juice Factory's manufacturing process.
3. Bottle Filling human–machine interface (HMI)
    - Image: paradoxxs/virtuaplant:hmi
    - Description: The HMI allows operators to monitor and control the PLC. It provides a user-friendly way to interact with the bottle filling process.
4. Vulnerable Samba Server
    - Image: vulhub/samba:4.6.3
    - Ports: 4450:445
    - Description: This container runs a version of Samba, a file-sharing service that has known vulnerabilities.
    - PostgreSQL Database
        - Image: vulhub/postgres:10.7
        - Ports: 5432:5432

    - Description: A vulnerable version of PostgreSQL.

The Docker Compose configuration file specifies a single network with the name flat, which connects all services together into a single network.

```
networks:
  default:
    name: flat
```

Just reading the configuration file is not enough. Let's spin up the environment. The first step is to open a terminal and navigate to the directory where the Docker Compose file is located. Once you are in the directory, you can start the environment by running the following command.

```
docker-compose up -d
```

It should now pull the images and start the containers. Once the containers are up and running, you can check their status by running the following command:

```
docker ps
```

If all the containers are up and running, head over to the Juice Shop web application by opening the browser and navigate at http://localhost:8080. It should look something like Figure 4.1. This is the front-end of the Juice Factory's e-commerce site, where customers can browse and purchase different juice products.

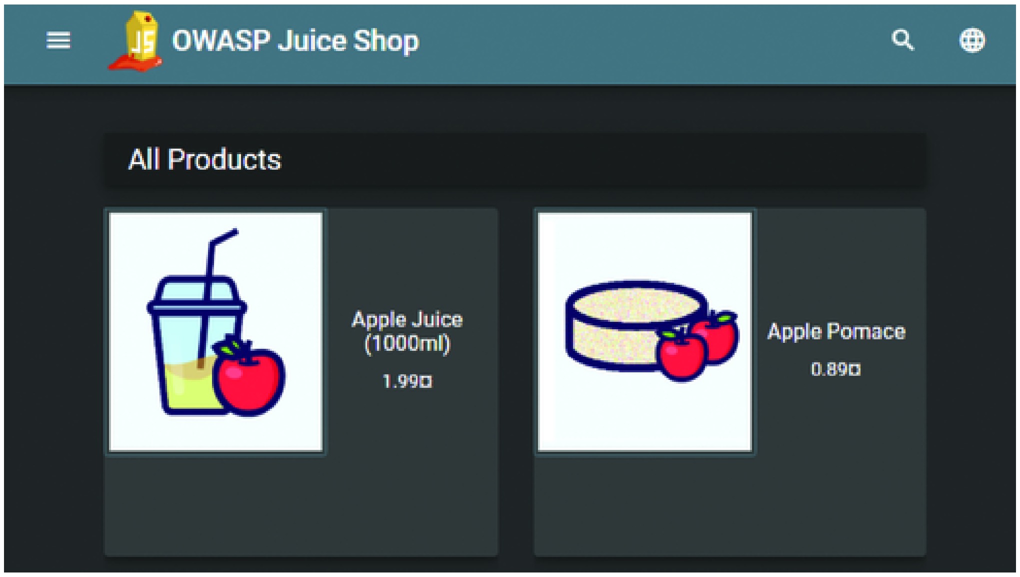


Figure 4.1 **Juice Shop.**

That is not all. We also have the production system of the Juice Factory, which includes a PLC and HMI. You can access the PLC and HMI by navigating at http://localhost:3001 to access the PLC, and the HMI is accessible at http://localhost:3002. The PLC is a Python script that simulates a PLC that is used in the production line for controlling the juice bottle filling process. Figure 4.2 displays the production system.

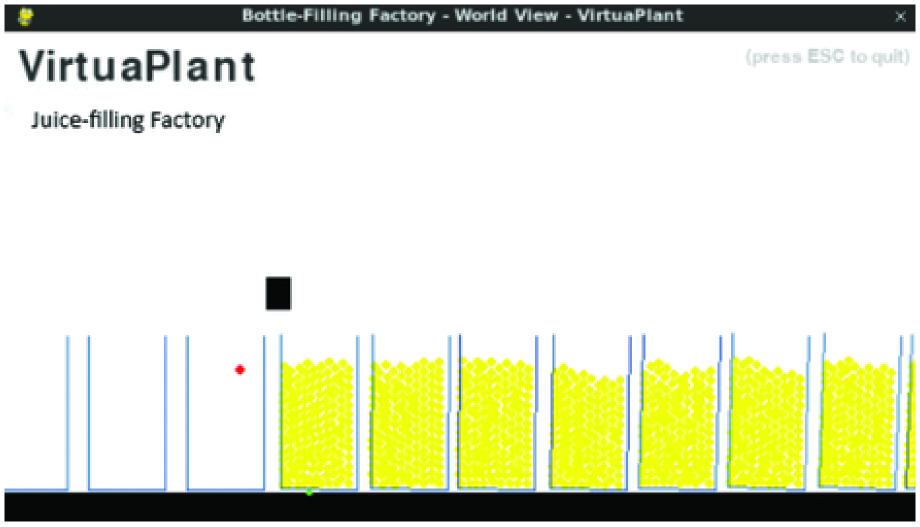
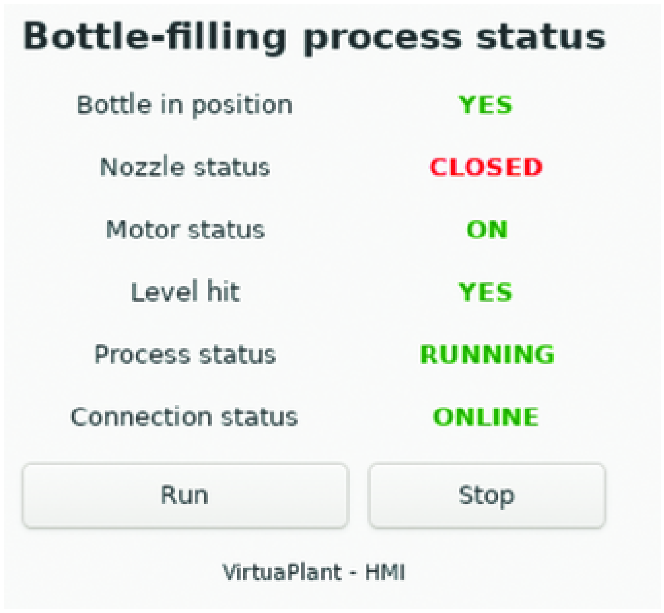


Figure 4.2 **Juice Factory – production system.**

The HMI allows you to monitor and control the PLC, as shown in Figure 4.3.



Bottle-filling process status

Bottle in position	YES
Nozzle status	CLOSED
Motor status	ON
Level hit	YES
Process status	RUNNING
Connection status	ONLINE

Figure 4.3 Juice Factory – HMI controls.

The Juice Factory's all services are shared on a single network, creating a flat network. One advantage of this setup is the ease of adding new services to the network, which can immediately communicate with existing services. This means a single, shared network connects all services and components together. This design simplifies communication between services, as everyone can communicate with each other. There is just one problem: if a threat actor compromises one system, they can easily move to another system. Many legacy business networks have a similar network design. A simple way to mitigate this problem is with segmentation, which we'll address in Chapter 5. In Figure 4.4, you can see the network diagram of Juice Factory.

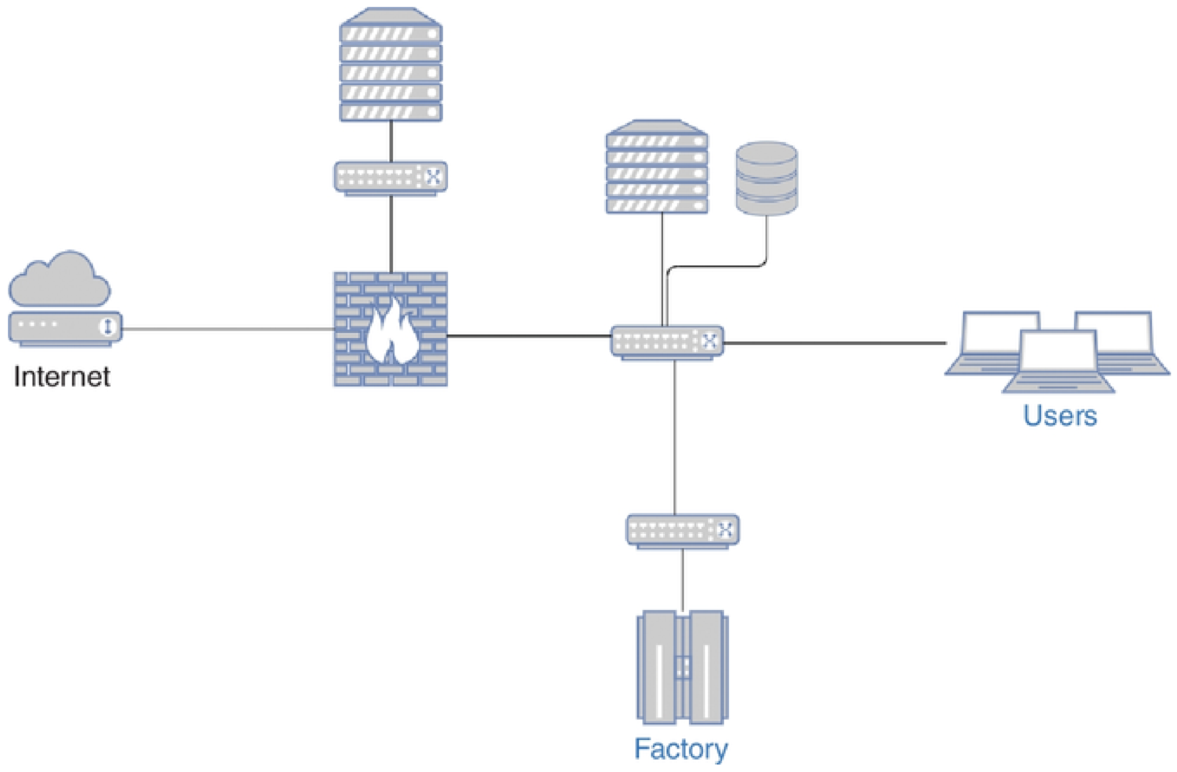


Figure 4.4 Network diagram.

The primary security concern with this architecture is the lack of network segmentation. When an attacker compromises any single system within the network, they gain immediate access to communicate with all other systems. For instance, if an attacker exploits a vulnerability in the public-facing Juice Shop web application, they could directly interact with the manufacturing systems, including the PLC and HMI interfaces. This is called 'lateral movement', where attackers can pivot from one system to another without encountering network-level security controls. As we progress through the chapters of the book, we will introduce network segmentation and other security measures to help protect the organisation and align with the Zero Trust principles.

4.2 Summary

This was a quick overview of the initial environment of Juice Factory that we will work from. It was created with the purpose of being vulnerable, so we have some projects that we can implement to help make the environment more secure.

Chapter 5
Network Segmentation and Network Security

> Three months into your role as CISO at Juice Factory, you've invested significant time understanding the organisation's departments, operations, and current IT infrastructure. As you climb the stairs to your office one morning, the CEO catches up with you. "Hey," she says, falling into step beside you. "We've been talking about Zero Trust for months now. Any plans for what your first initiatives will be?" It's a fair question. Your initial assessment has revealed multiple security concerns within the environment, from the flat network architecture to inadequate access controls. The challenge isn't identifying what needs to be done. It's determining where to start for maximum impact with minimal disruption to business operations.

Now that we have got an understanding of the current state of the environment. You might look at the environment, and you think what to do first? There are currently multiple security concerns within the environment. What initiative should we begin with? For the first project, it should be something that impacts the security state of the organisation. While also having a minimum impact on the production of the organisation. This is where the design principles of Zero Trust provide valuable guidance. By systematically analysing Juice Factory's environment through the Zero Trust framework, we can identify the most effective first steps:

- Identify Protect Surfaces: Operations production and corporation for handling sales, and then there is the web store for B2C sales.
- Map Interactions: There is no interaction between the operational technology (OT) environment and the rest of the network. The web application does not interact with the other components of the network.
- Design Protective Measures: Segmenting the OT environment from the rest of the network will provide the most significant security improvement for the least amount of time, effort and money.

Our assessment of protect surfaces reveals three distinct operational areas: the production environment, corporate systems for sales operations and the B2C web store. Each of these areas requires different levels of protection based on their business impact and risk profile. When mapping interactions between these areas, an important pattern emerges: the OT environment operates independently of the corporate network, and the web application has no business requirement to interact with other internal systems. This isolation in business processes suggests a natural starting point for our security improvements. Based on this analysis, network segmentation emerges as our optimal first initiative. By separating these naturally isolated systems into distinct network segments, we can achieve significant security improvements with minimal operational impact and reasonable resource investment. This approach aligns perfectly with Zero Trust principles while providing immediate risk reduction. Throughout this chapter, we'll explore how network segmentations can be implemented and then change our Docker environment to separate it into smaller networks. Now we have an initiative for our first project for the environment. Network segmentation is a foundational security control that divides a network into isolated segments, each with its own security policies and access controls. In Zero Trust Architecture, effective segmentation is crucial as it helps enforce the principle of least privilege and contains potential security breaches. By implementing network segmentation, organisations can:

1. Isolate critical systems and data from potential threats
2. Enforce granular access controls between segments
3. Reduce the attack surface available to threat actors
4. Maintain compliance with regulatory requirements
5. Improve network performance and manageability

Having a good network segmentation design allows you to define what devices are high and low risk, and the same goes for user types. This informs what security controls need to be implemented in each segment. In this chapter, we will discuss the concept of network segmentation and its improvements in security within our organisation. What we will do is take our initial flat network design and transform it into a segmented network by creating network zones for the different business functions.

Specifically, we'll separate our network into three different segments: Demilitarised Zone (DMZ), Production and Corporate. The goal of the segmentation security controls is to reduce the probability of an event in one part of the network affecting another part by keeping the most important assets and data separate from the more vulnerable systems.

5.1 Segmenting the Network

Network segmentation is taking an IP range and splitting it into smaller pieces. In Chapter 4, we see that our docker environment used the 172.19.0.0/16 subnet for the entire network. With segmentation, we take the IP range and break it into smaller subnets. We then use network policy to prevent communication between these zones. Segmenting your network can significantly enhance the security of the organisation using minimal time, effort and cost. An effective network segmentation plan separates high- and low-risk devices and users. There are many approaches for how to split devices into different segments. This can be based on the physical location of the device, business unit, the criticality of the business or micro-segmentation. Which approach to pick depends upon the resources available and the complexity of the network. The more fine-grained the segmentation, the deeper an understanding is required for how the different systems communicate with each other. Not only that, there is also a need for better documentation of the network itself to help with troubleshooting network problems. When implementing network segmentation in production environments, organisations typically follow these steps:

1. Asset Classification and Mapping
    - Identify and classify all network assets
    - Document data flows and dependencies
    - Define security requirements for each asset class
2. Segment Design
    - Create logical separation based on:
        - Business function

        - Data sensitivity
        - Compliance requirements
        - Operational needs
3. Policy Development
    - Define inter-segment communication rules
    - Establish access control policies
    - Document monitoring requirements
4. Technical Implementation
    - Configure network devices
    - Implement access controls
    - Deploy monitoring solutions
5. Validation and Testing
    - Verify segment isolation
    - Test business continuity
    - Confirm policy enforcement

A good example of segmentation is with Internet of Things (IoT) devices, which are often less secure because of minimal testing and weak default security settings. This makes IoT devices inherently more vulnerable. Placing these devices on a logically or physically separate network will lower the risk of threat actors exploiting them and moving laterally to more critical parts of the network. Micro-segmentation is the recommended approach for network segmentation, where we only allow each device to communicate with the systems that are explicitly required to complete its task. This granularity should be down to the specific ports and protocols used. A more common approach is to put devices into trust groups and allow them to communicate with each other freely. The problem with micro-segmentation is that there is a larger initial cost to that approach, because of the upfront design of the network architecture. This upfront cost will result in a more secure network and a much more in-depth understanding of what normal activity looks like on the network.

5.2 Key Technologies for Network Segmentation

Before we go on, we must have a basic understanding of the different network equipment that exists and their purpose.

- Firewalls: Firewalls are used to enforce network policies for the traffic that is passing through it. Traditional firewalls filter traffic based on IP addresses, ports or Open Systems Interconnection (OSI) layers 3 and 4.
- Next-generation Firewalls (NGFWs): NGFWs go beyond the capabilities of a traditional firewall by offering advanced features such as application awareness and control, integrated intrusion prevention, threat intelligence and advanced malware protection. This technology allows for more precise control over network access, which improves network security.
- Routers and Switches: These devices control the flow of traffic within and between network segments. Administrators commonly configure Virtual Local Area Networks (VLANs) on switches to create isolated network segments on the same physical network. VLANs help in logically separating different traffic, such as separating guest network traffic from internal corporate traffic. A router is the conduit between these two networks. In your own home, you probably have a router that connects your home network to the internet, allowing you to browse the internet.
- Access Control Lists (ACLs): Routers and firewalls use ACLs to control traffic flow between network segments.
- Network Access Control (NAC): NAC solutions enforce security policies on devices attempting to access the network. NAC can ensure that only compliant and authenticated devices can connect, helping to maintain the security and integrity of network segments.

5.3 Implementing Network Segmentation

Implementing network segmentation can vary widely depending on the setup of the environment. For an organisation that has embraced the infrastructure as code (IaC) approach changes, the segmentation of the network comes down to rewriting the environment configuration file. However, an organisation with a more traditional setup of switches, firewalls and routers might need to manually create network policies by hand. Overall, there are only two approaches to segmenting a network: physical segmentation, where each segment uses separate wires and network equipment, and logical segmentation, where policies within a switch or firewall define the segments and ensure correct package routing.

5.3.1 Physical Segmentation

The simplest way to separate two networks is to use different network devices and wires for each network. This way, the two networks will never interact with each other. An example would be to have two different modem: one for IoT devices and another for user endpoints. This way, a device on one network can never interact with a device on other network. That would require there being a separate internet connection for each of the networks. The benefit of a physical segmented network is that it is harder for an attacker to overcome and pivot from one network to another compared to a logical segmentation network. Pivoting is the act of moving from one compromised host on the internal network to another, which is common for threat actors for them to achieve their objective. Physical segmentation's drawbacks include increased administrative overhead, higher hardware and infrastructure costs and the need for separate internet connections for each physically separated network. The only organisations that I have seen who do physical segmentation of networks are government organisations. Where it is required to physically segment lower classified networks from higher classified networks.

5.3.2 Logical Segmentation

Logical segmentation is a far more common approach and less expensive to implement, as it does not require the need to buy separate hardware for each network. We achieve logical segmentation with VLANs, making devices appear to be on the same network while logically separating them from other devices. Where each VLAN has its own virtual switch that exists within the physical switch. The way it configured on the switch is through the usage of network policies that define what port on the switch can communicate with which VLANs. This is called port-based VLANs. You can also do IP-based VLANs. Here, you split up the IP range into smaller segments. With the devices on these network segments can communicate with each other, any outbound traffic is routed through a shared gateway. Setting up logical VLANs is an easy 'win' for an organisation for making it more difficult to pivot around the network for a threat actor. An easy way to create segment is through port or private VLANs. To understand private VLANs, we first have to understand the different types of ports that make private VLANs possible. The flexibility of private VLANs comes from the different port types that exist.

- Promiscuous Port: Can send and receive traffic to and from any device on the VLAN.
- Isolated Port: May only communicate with promiscuous ports.
- Community Port: May send traffic to promiscuous ports or other ports of the same community. Cannot send packages to isolated ports.

With just these three port types, it is possible to take any network and create a segment plan that can handle almost all scenarios. Use isolated ports for systems requiring separation from the rest of the network. And VLAN community for systems that require communication with other systems. Figure 5.1 shows a diagram of the three types of VLAN port to better help understand how they work.

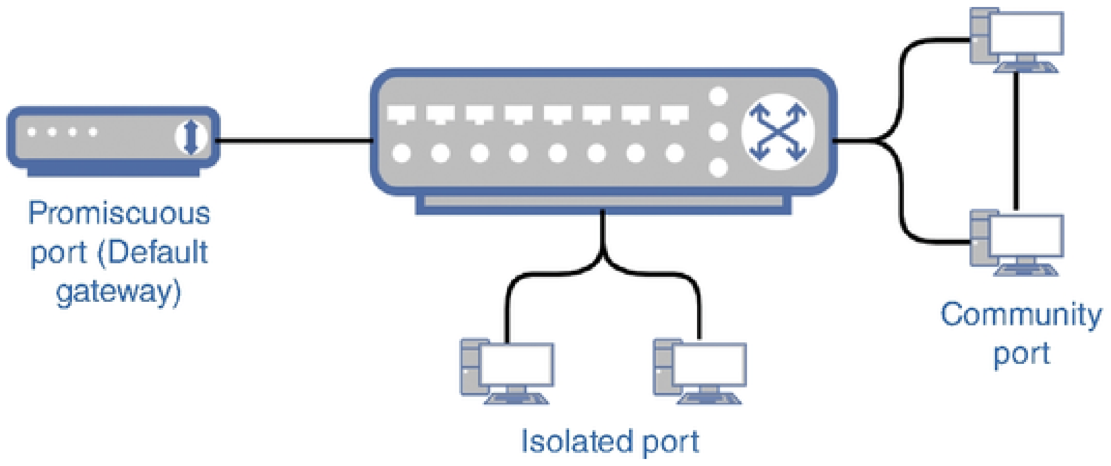


Figure 5.1 **VLAN ports.**

The diagram shows the three types of ports that make up a private VLAN. With the isolated port only being able to send packages to the promiscuous port. The devices connected to the community ports can send network packages to each other and the promiscuous port. The diagram shows only one community; however, multiple communities are allowed. When setting up a private VLAN, the first step is to define a promiscuous port or uplink port. This port is normally the one connected to the outbound router, firewall, etc. Once you have defined that, the next step is to define the isolated ports. A package that is received on an isolated port is only forwarded to the promiscuous port, no matter the IP or medium access control (MAC) address of the destination. The packages are then forwarded as usual from the promiscuous port to the destination. This is, of course, not without potential issues. A common issue that is experienced is with peer-to-peer applications, such as with some communication platforms or Windows 10 peer-to-peer patching mode. Most applications have a simple workaround: configure a gateway to receive packages. You can also use a private VLAN in a DMZ to restrict communication between systems without the effort and cost of creating separate DMZ segments for each service. Network administrators can be resistant to VLANs, claiming that it requires a lot of work to set up. Requiring setting up VLAN for each client on the network is not true. You can create a single VLAN for all clients, configure it to be in private mode and then place all your clients on that single VLAN.

5.3.3 DMZ Design

Not all segments are equal; a common network segment to have is a DMZ, which has the purpose of creating a protective layer around the services that need to interact with the internet. The goal is to isolate these systems from the rest of the internal network. This is because they are accessible from the internet; this means that they will be more frequently attacked than any part of the network. An increase in attack activity will lead to a higher risk of that part of the network being compromised. For that reason, the DMZ has to be designed in such a manner to mitigate the possibility of a compromised DMZ leading to the compromise of the entire network. Which means, we have to be good at not only analysing the connection coming from the internet to the DMZ but also the connection from the DMZ to the internal network. A good belief to hold is that any system that can be accessible by the internet is compromised. We should design systems according to this principle. It is not uncommon to have all internet-facing systems on the same DMZ segment, allowing them to communicate with each other, even if they do not have the need. That does not follow the principle of Zero Trust design. Here, private VLANs offer an elegant solution to the problem of containment. We do this by configuring a private VLAN on the DMZ switch and configuring the outbound firewall interface as a promiscuous port. Then have all internet-facing services as isolated ports. This way, the different services can only communicate with the firewall and no one else on the VLAN.

5.4 Segmenting the Juice Factory Network

Now that we have a basic understanding of network segmentation. The first question you might have is how many zones and what services should be in each zone? There are multiple ways of mapping this out. Because of the limited control we have with Docker, we will be dividing the network based on business processes.

1. The first segment contains everything the outside world would like to connect to, such as the web application. We will call this network DMZ. In a real organisation, it is normal to have multiple DMZ segments for each internet-facing service, to mitigate the risk of one DMZ system compromising another. Since we only have our customer store, we only need a single DMZ network.

2. Then there is the internal network, which is where the employees need to interact with different services to perform their duties. This segment will be the corporate environment for sales, HR, etc.
3. The last is the OT environment, which handles the production of juice.

Your organisation might have more business segments you can slice the network into, or you can do a tier approach based on the criticality of the organisation. Let’s look at the network design of our initial flat network and divide it into three distinct segments: DMZ, Production and Corporate. Figure 5.2 shows the revised network diagram.

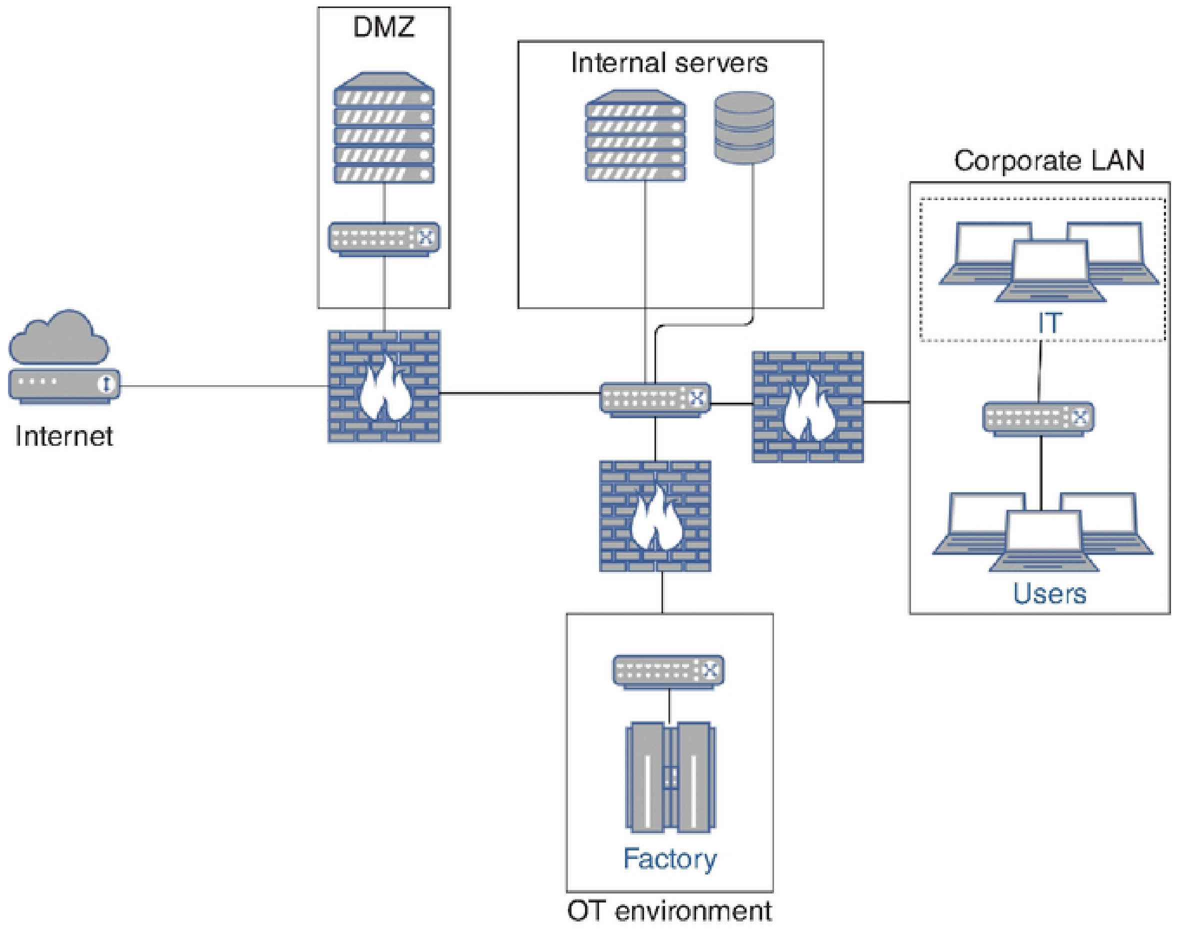


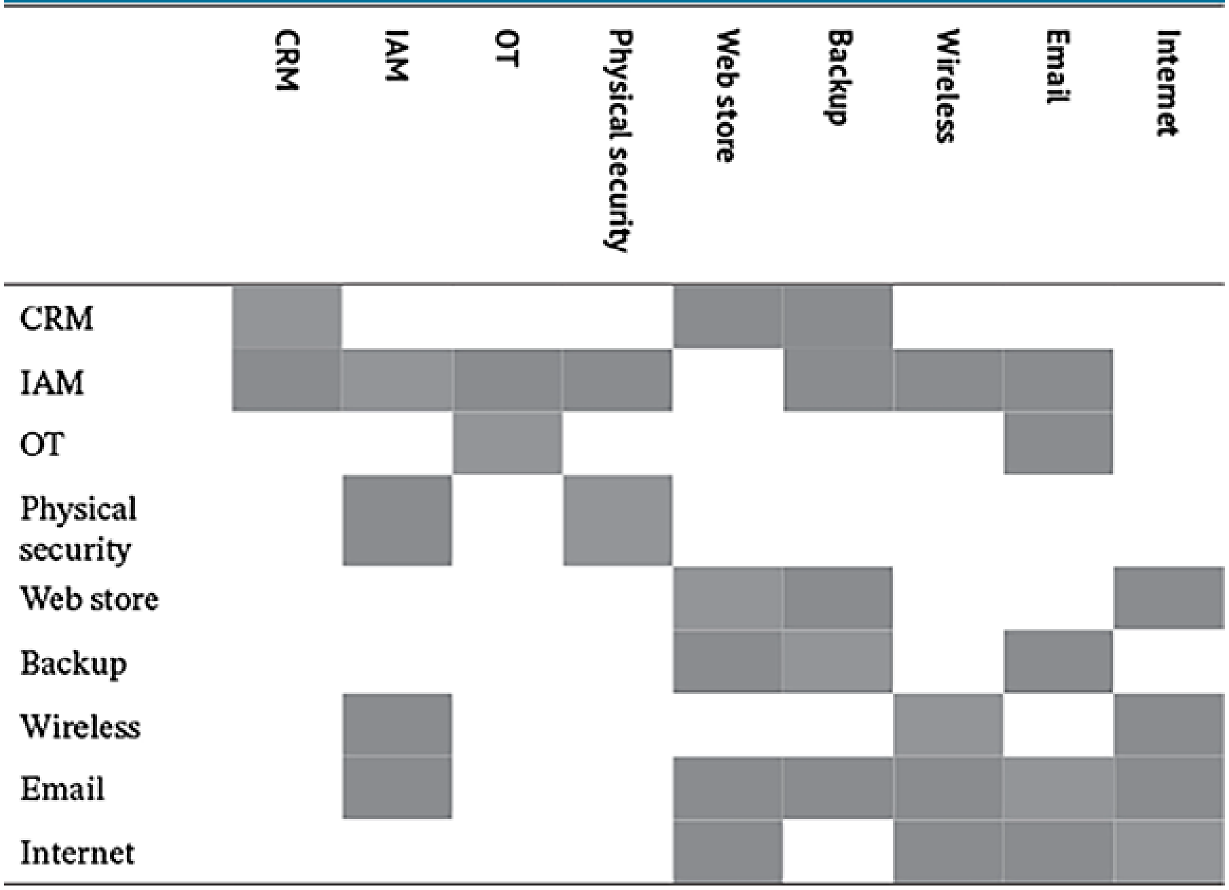
Figure 5.2 Network diagram.

Understanding a network diagram can be difficult, especially for people without networking experience. They also do not tell the full story. What we are looking for is getting a better overview of what systems will interact with one another. To solve that problem, we will create a transaction map. It is a matrix of all the protected surfaces we have defined in our environment. Because we do not have a fully simulated network, I have decided a few extra protected surfaces that would exist in most modern organisations. This includes things such as email, wireless and customer relation management (CRM).

When looking at the transaction map in Table 5.1, notice how at a quick glance it is possible to identify the protected surfaces and how each of them interacts with one and other. Throughout this book, we will be updating both the network diagram and the transaction map to reflect the current situation of the network, as additional systems and security controls are added to the network.

TABLE 5.1

Juice Factory transaction map.



	CRM	IAM	OT	Physical security	Web store	Backup	Wireless	Email	Internet
CRM	■				■	■			
IAM	■	■	■	■		■	■	■	
OT			■					■	
Physical security		■		■					
Web store					■	■			■
Backup					■	■		■	
Wireless		■					■		■
Email		■			■	■	■	■	■
Internet					■		■	■	■

5.4.1 Implementing Network Segmentation in Docker

We previously segmented the network into three pieces: DMZ, production and the internet. We now need to take that design and implement it within Docker. To achieve the desired network segmentation, we will define the three networks in our Docker composed configuration file. The configuration file after we segmented the network should look something like this:

```
services:
  juice:
    image: bkimminich/juice-shop
    ports:
      - 8080:3000
    networks:
      - dmz
  bottle-filling_plc:
    image: paradoxxs/virtuaplant:plc
    ports:
      - 5020:5020
      - 3001:3000
    networks:
      - Production
  bottle-filling_hmi:
    image: paradoxxs/virtuaplant:hmi
    environment:
      - PLC_SERVER_IP=bottle-filling_plc
      - PLC_SERVER_PORT=5020
    ports:
      - 3002:3000
    depends_on:
      - bottle-filling_plc
    networks:
      - Production
  vuln_samba:
    image: vulhub/samba:4.6.3
    tty: true
    volumes:
      - ./samba/smb.conf:/usr/local/samba/etc/smb.conf
      - ./samba/data:/storage
    ports:
      - "4450:445"
    networks:
      - corporation
  postgres:
    image: vulhub/postgres:10.7
    ports:
      - "5432:5432"
    environment:
      - POSTGRES_PASSWORD=postgres
    networks:
      - corporation
```

```
networks:
  dmz:
    name: dmz
  Production:
    name: Production
  corporation:
    name: corporation
volumes:
  db_data:
```

This setup will create distinct networks, one for each defined network in the compose file. This will effectively isolate each of the networks from each other, similar to how VLANs work in a real-world environment.

5.4.2 Inspecting Docker Networks

Since we do not have a physical network, it is not possible for us to login into a switch and inspect the network from there. Luckily for us, it is possible for us to inspect the Docker like we did in Chapter 4. I have added a few additional commands to help with managing and inspecting the internal networks of our Docker environment. Here are some of the most useful commands:

- List networks:

  ```
  docker network ls
  ```

- Inspect a specific network:

  ```
  docker network inspect <network_name>
  ```

- Create a network:

  ```
  docker network create --driver bridge <network_name>
  ```

- Remove the specified network:

```
docker network rm <network_name>
```

- Inspect the DMZ Network for additional details:

```
docker network inspect dmz
```

The following is an example output of the DMZ network; yours will most likely differ from this one.

```
"Name": "dmz",
"Id": "e1f5f6a28c6a...",
"Created": "2024-06-20T14:32:24.0267178Z",
"Scope": "local",
"Driver": "bridge",
"EnableIPv6": false,
"IPAM": {
  "Driver": "default",
  "Options": {},
  "Config": [
    {
      "Subnet": "172.18.0.0/16",
      "Gateway": "172.18.0.1"
    }
  ]
},
"Containers": {
  "e9bb1a3c3b4b...": {
    "Name": "juice",
    "EndpointID": "e4b1d6b3e1b2...",
    "MacAddress": "02:42:ac:12:00:02",
    "IPv4Address": "172.18.0.2/16",
    "IPv6Address": ""
  }
},
"Options": {},
"Labels": {}
```

It is not enough to just inspect the network configuration. We also have to verify that the segmentation is actually working. To do that, we have to jump onto one of the containers and ping a container that exists on another network segment. If everything works as intended, you should get a ping fail error message back.

1. Jump onto one of the containers defined in the configuration file.

   ```
   docker exec -it <container-name> /bin/bash
   ```

2. Once you have shell access, install the iputils package that enables to ping. Use the following command to achieve this:

   ```
   apt update && apt install iputils-ping
   ```

3. You should now try to ping a container that exists on another network. You can do this using either the container name or its IP address. For bonus points, you can also ping a container that is on the same network, to verify that everything works as intended.

   ```
   ping <container-name>
   ```

5.5 IPv6

The forgotten IP space that many are not thinking about. When visiting an organisation and asking about their IPv6 usage, we often hear that they only use IPv4. The next question is always 'Oh, so you have disabled IPv6 on all devices?' They will look at you confused, as they have only setup IPv4. What many do not known about IPv6 is that it is enabled by default and often automatically used. All Windows operating systems since Windows Vista have used IPv6 by default. That means all modern operating systems both support but also generate IPv6 traffic unless it has been explicitly disabled.

Before we go into details about what this means for your network, let's go over how IPv6 works and how it differentiates between IPv4 and IPv6. IPv6 was created to address the problem of having to few IP addresses. IPv4 allows for 4.2 billion addresses, which sounds like a lot. The designers of the 1970s underestimated the demand, as we surpassed 4.2

billion devices long ago. Just think about all the devices you have that are connected to the internet. With IPv6 it uses a 128-bit address space, compared to the 32 bit of IPv4. This translates to 340 undecillion addresses, which should be more than enough. When it comes to distinguishing between the two, it is quite easy, not only because of the length of the address but also because IPv4 uses a 'dot' format, for example, 192.168.1.0. Where IPv6 uses ':' to break up the IP address like 'fd7a:115c:a1e0::d001:9335', note also that IPv6 uses hexadecimals, instead of only decimals. You might also have noted the '::' part. IPv6 uses this to summarise repeating strings of zeros. For example, 'fd7a:115c:a1e0::d001:9335' is the summarised version of 'fd7a:115c:a1e0:0000:0000:0000:d001:9335'. This summarisation only occurs once to avoid confusion. That is not all. IPv6 protocol also has a much cleaner and simpler header compared to IPv4, and routing is also much simpler and has many additional features, which I will talk about later.

Since this section is about segmentation, let's also look at IPv6 subnets. It's common practice for an organisation to be assigned a /48 network. This allows the IP space to be split into '10 995 116 277 776' subnets, which they can use to allocate to their internal resources. The smallest subnet of IPv6 is /64 networks. Starting from a /48 network that is split up into /64 networks allows for 65 536 subnets. Each subnet having 18+ quintillion addresses. Then there was also another 65 536 possible subnets using the local unique addresses. Not all IPv6 addresses are the same. They come in three different types.

- Link-local addresses – Used on the local subnet, begins with 'fe80'. All IPv6 systems have this address. Unique local addresses (ULAs), similar to private IPv4 addresses and starting with 'fd00', and used on private networks. This means they are not publicly routable, but are globally unique.
- Global unicast addresses are the addresses that are publicly routable, which means they are the address that we are used to connect to the internet.

What is the use of ULA? Because these addresses cannot reach the internet, we can use them as a layer of defence. Take this example of an organisation hosting a file share for internal resources only by having the file share service only listen to the ULA address and not the global unicast addresses. The file share service can never reach the internet directly. The reason that ULAs are globally unique is that when two private wide area networks (WANs) need to be connected, it avoids the requirement of reassigning devices with new IP as the two networks come together. All IPv6 systems possess a link-local address, which by default enables communication only within that specific subnet. To add to the whole thing, a single system can have multiple unique local and global unicast addresses, plus privacy extension addresses, which we will talk about later. There is also loopback or 'home', which is another address type that every system possesses. This is at 127.0.0.1 for IPv4 and ::1 for IPv6 (::1 represents a series of 31 zeros followed by a one), which is used by the system to communicate with itself.

Another feature of IPv6 is Stateless Address Auto Configuration (SLAAC). It allows a system to independently determine its own IPv6 address, eliminating the need for additional infrastructure or servers. This means that a Dynamic Host Configuration Protocol (DHCP) is no longer necessary. Instead, what the system does is listen for an upcoming IPv6 global prefix router advertisement and use the advertised prefix as the network portion of its global address, with the host portion of the address based on the system MAC address. The system combines these two to form the IPv6 address. These router advertisements can come from any IPv6-enabled routers. The link-local address goes through the same process, just without the router advertisement. If there are no routers advertising for the IPv6 global prefixes, the system will only assign itself a link-local address.

This is somewhat a debatable benefit, as many organisations would like more control of their IP address assignment process. They originally designed this method to create global unicast addresses, think public IPv6 addresses, by taking the system's MAC address while using a public network prefix. The problem with the protocol is that it creates new privacy issues. It allows sites on the internet to track the systems via their IPv6 address, regardless of the network they were on. Because the MAC addresses are part of the address, as they are globally unique. This enables sites to build databases of MAC addresses associated with IP addresses and monitor them efficiently. Due to this privacy issue, a new protocol was developed, which defines unique local and global unicast addresses. This protocol is called privacy extension. These addresses are no longer based on the system MAC address. The operating systems use these privacy extension addresses as their unique local and global unicast addresses and will continue to embed the MAC address into the link-local addresses, which are only used on the local subnet. That means every system will create two addresses for each unique local and global unicast address. The permanent addresses that contain the MAC address of the system and the temporary privacy address that is preferred for all communication.

The next question might be how are these privacy-enhanced IPv6 addresses then generated? Because of the sheer number of possible addresses, the host portions of the IPv6 privacy-enhanced addresses are simply generated by random on every system. The next question you might have is what about collision, if the addresses are just generated at random? That is where the large number of possibilities comes into play. Remember that each subnet has 18+ quintillion addresses. That means the sun will most likely become a supernova before any two hosts on any network worldwide receive the same random number. They added duplicate address detection as an extra protection level. This will generate a new random address, if the current address is in use on the network. Remember that systems only use privacy-enhanced IPv6 addresses when they use SLAAC to generate an IP address; they don't use them when a DHCPv6 server or other method assigns the IPv6 address. These privacy addresses also come with a preferred lifetime and a valid lifetime. When the preferred lifetime is reached, the system will generate a new privacy address used for all new connections. The valid lifetime is for how long the system will continue to accept new packages from existing connection on that address. On Windows machine, it is possible to know the lifetime of IPv6 addresses by using the command:

```
netsh interface ipv6 show address
```

Using the command, it is also possible to view the default lifetime of a privacy address.

```
netsh interface ipv6 show privacy
```

On most systems, the default IPv6 preferred lifetime is one day, and the valid default lifetime is seven days.

Let's imagine you want a little more control of what IPv6 addresses get assigned to systems. There are a few methods available for assigning IPv6 addresses. First, it statically assigns the IP address to the system. A more automatic method is to have the system use SLAAC to assign the host part of the address stateless and control the network prefix using a router advertisement daemon like radvd.[1] The last option is the usage of a DHCPv6; it mirrors the IPv4's DHCP functionality. Because DHCPv6 cannot be used to assign the default gateway, it also requires the usage of a router advertisement daemon.

The last thing I would like to look at before moving on to the security consideration of IPv6. With the sheer number of IP addresses for IPv6. The question is, how do we perform host discovery? Unlike IPv4, IPv6 lacks support for broadcast addresses. Instead, IPv6 uses multi-cast and anycast address to achieve a similar result, allowing the system to discover the hosts that exist on the same network. These broadcast addresses allow a single system to communicate with all hosts on the network and have their responses back with their IP address. IPv6 uses the ff00::/8 network prefix for multi-cast addresses. The two most important broadcast addresses are:

- ff02::1 – All local nodes
- ff02::2 – All local routers

Now that we have a basic understanding of how IPv6 works, let's move on to how IPv6 can affect the security of our environment.

5.5.1 Securing IPv6

The National Institute of Standards and Technology (NIST) has created a special publication 800-119[2] about how to deploy IPv6, with the name 'Guidelines for the Secure Deployment of IPv6'. Here is a snippet of the risks, including in deploying IPv6.

- Unauthorised deployment of IPv6 on existing IPv4 networks

- Dual operations of IPv4 and IPv6
- Lack of support from the existing environment

As you learned from Section 5.5, every environment with Windows Vista or newer is using IPv6 right now. And the problem is most are not aware of it. To add to the whole mess, not all firewalls can handle IPv6 traffic, defaulting to open state, allowing all traffic through. Let's go into a little more details about the different security issues of IPv6.

The first thing I would like to highlight is Small Office/Home Office (SOHO) modem. Most of them have IPv6 firewall support and are often very limited. Which comes with the IPv6 secure settings turned off to ensure functionality. One test of a home modem showed that enabling the IPv6 security setting only secured TCP and UDP traffic for IPv6, still allowing ICMPv6 traffic through. This gives the possibility of a threat actor to tunnel traffic through ICMP, bypassing all security. For enterprise solutions, they often come with better security features. The question is, have they been enabled and configured?

What about discovering if anyone is using IPv6? It is not really feasible to scan the entire IPv6 to see if any system would be responding. What we need is a way of scanning for any usage of IPv6. Fortunately, most operating systems include native tools we can use for discovery. These tools will list the IPv6 Neighbour Discovery Protocol (NDP) table and the IPv6 route table. We can use this information to determine if anyone is using IPv6. On Windows and Linux, use the following commands to ping IPv6 addresses and view the IPv6 NDP and IPv6 route tables (Table 5.2).

TABLE 5.2

Comparison of IPv6-related commands across different operating systems.

OS	IPv6 route table	IPv6 ping	IPv6 NDP table
Windows	netsh interface IPv6 shows route	ping	netsh interface IPv6 show Neighbour
Linux	netstat -A inet6 -rn	ping6	ip -6 neighbour show

Another option is to monitor the network traffic. We will go into more details about network monitoring in Chapter 6. One thing you have to be aware of besides making sure that the monitoring system can monitor IPv6 traffic is the placement of the monitoring system. It can only monitor the traffic that is going through the interface that is connected to the monitoring system. In a segmented network, you would have to have a monitoring system at each junction point, to help discover any local subnet usage of IPv6.

Another security concern of IPv6 is that it comes with tunnelling features, which is a technique using the IPv4 network for IPv6 traffic. They encapsulate the IPv6 traffic inside IPv4 packets, allowing IPv6 networks to work over the existing IPv4 infrastructure. This feature is necessary as many parts of the internet still rely on IPv4. The following is an example of some of the tunnelling protocols.

- 6 to 4 – Tunnels IPv6 packets using IPv4 networks, with the need for a tunnel.
- Teredo – Designed for tunnelling IPv6 traffic across NAT Network Address Translation devices in IPv4 networks. Often used by home network.
- ISATAP Intra-Site Automatic Tunnel Addressing Protocol – Similar to 6 to 4 using DNS.

These IPv6 tunnels make both prevention and detection difficult. To add problem, these tunnels can bypass the security controls of intrusion detection systems (IDSes) and intrusion prevention systems (IPSes). For us, what is important is how can we discover if tunnelling is happening within our network? One way of discovering IPv6 tunnelling is by looking at the protocol field of the IP header, which is the third layer of the OSI model.[3] Here you have to look for the value 41 inside the protocol field, which is the protocol for IPv6. Most network analysis tools, such as Suricata (which we will discuss in Chapter 6), use the detection rule like 'ip_proto:41' to detect possible IPv6 tunnelling.

For Teredo tunnelling, we have to use a slightly different strategy. By default, it uses UDP port 3544, which is one way to detect Teredo tunnelling. This will, of course, generate a lot of false positive, as other protocols can use the same port. To add to the complexity of detecting

Teredo, it is possible for it to use other ports. Using only the port number is insufficient for detection; additional attributes enhance detection and lower the false positive rate. Tools such as Wireshark and Tshark (discussed in Chapter 6) come with build-in filters for detecting Teredo tunnelling, no matter what port is being used. What about some preventive controls that can limit the possibility of threat actors using IPv6? Some Cisco routers that support IPv6 come with the ability to block protocol 41, using the command 'deny 41 any any log'. Remember that prevention is nice, but detection is a must. Just because you have configured your router to block protocol 41. This means you should still have detection rules alerting on this kind of activities.

Another attack that can happen on IPv6 is rogue IPv6 Router Advertisements. Which is a malicious system that will broadcast router advertisements to the local subnet, exposing the subnet to the public IPv6 Internet. The attack follows the following pattern:

1. It starts with a compromised system.
2. The actor creates a tunnel from the compromised system to the IPv6 Internet.
3. The compromised system sends IPv6 router advertisements to the local subnet, identifying the compromised system as an IPv6 router.
4. The system on the local subnet will not create a global unicast address, using the network prefix assigned by the rogue IPv6 router.
5. Local subnet is now exposed to the public IPv6 Internet.

When it comes to defending against these types of attacks, one of the best approaches is through detection. Alerting when router advertisements are coming, multiple sources or unauthorised devices should trigger an alarm. Router advertisement also come with a preference option. This tells the endpoint which router advertisement it should prefer. Sending the legit router advertisements with high priority preference will make it harder for an attack to overwrite the legitimate IPv6 configuration.

This is not a complete list of all the security concerns of IPv6. The goal of this section was to give you enough information to understand IPv6 and some of the security considerations that you should be aware of. The idea is to now have you not forget about IPv6 and take an active stand that both IPv4 and IPv6 landscape are properly architected and protected.

5.6 Web Proxy

We are not only limited to segmenting the internal network. To further enhance the secure the network, we also need to create a layer of protection between our environment and the internet. The way we do this is through the usage of a web proxy. Web proxy helps us protect organisations assets as they try to access the internet. To be more specific, the proxy will analyse the browser traffic, including both HTTP and HTTPS. The way web proxy works is by having endpoints connect to the web proxy. The web proxy will then make a separate connection out to the desired destination. This results in two separate connections. This means that the endpoint connecting stops at the web proxy, never having touched the destination server. Giving the proxy full visibility into both the request and response coming and going to the web connection. Allowing the proxy to make security decisions on behalf of the endpoint. The way it secures the environment is by looking into the details of every web connection that is being made. This includes things like status code, MIME type, user agent and much more. This also allows the web proxy to catch malware using web connections to communicate with C2 servers, but whose formatting doesn't match the web proxy's endpoint expectations.

The web proxy will also perform a lookup of the sites that are being requested. This will include information about the site category like news, education, social media, etc. For all new sites will fall into the unknown category. Here, commercial proxy has the advantages as they come with a more current and accurate category classifications and reputations of websites. This makes it easier to determine whether to allow or block sites. Control site access by selecting a category in the web proxy and setting it to allow or deny traffic. A full list of web categories is available in Azure Firewall.[4] Most systems use a similar type of categorisation. This is not the only attribute they use. There is also a reputation score on the site. This reputation score is often based on if the site has been used to perform malicious attacks before. The population of the site is often determined by the site placement on the top 1 million of the most visited sites. It is possible to view 1 million visited domains over at majestic.[5]

Content filtering blocks sites using its reputation, category and other attributes to determine if it should allow or deny access to the site. How hard is it to bypass this kind of filtering? An adversary can easily overcome content filtering. The simplest approach to this is to purchase a pre-used domain name that is up for auction. Almost all domain registrars allow for the purchase of previously owned domains. The good thing about these domains is that they come with their previous category and reputation. An alternative way is to create a website with content specific to the desired category. Eventually, a content filtering service provider will scan the website and assign it to a category.

A better approach to protecting the organisation is the use of allow listing. The difference with this approach is that we only allow access to sites that have been authorised. This works by approving websites that have a business need; all other sites are blocked. A quick way to get started on implementing an allow list is to log all sites that are accessed over a period and have them as a starting point for the allow list. Moving forwards from there, any new news sites need to be approved. This approach, of course, also risks, including possible malware, but at least it gets things rolling forwards. The biggest problem with allow listing is the time needed to maintain this kind of list. How quickly can and should we expect the approver to authorise new sites? Having to wait for someone to approve sites before you can continue the work can quickly turn from a security tool into a business disabler, which is the number one reason why allow listing projects often fails. Not because of lack of technology, but because of the human resource requires authorising new website requests.

Web proxy is, of course, not without downside; because of the broken connection, it’s difficult to analyse the traffic. To help with this, some proxies add X-Forwarded-For, or XFF, to the HTTP header. This field will contain the endpoint IP address. Remember to remove the XFF header before any request leaves the environment. Otherwise, you risk revealing internal environmental information that could be exploited.

Another challenge of web proxy is the rise of encryption, which breaks the proxy ability to analyse the connection. All requests or response that are intercepted by the proxy, which is encrypted, are impossible for the web proxy to inspect. The encryption standard that is used for web traffic is called secure socket layer (SSL). It uses a handshake approach that verifies the server's certificate is valid. The problem with proxy is that it has the request and response between the site and the endpoint and must therefore act as a middle-man between the two. This will break the chain of trust between the server and the client, warning the user that something is wrong. To solve this problem, a SSL certificate can be added to the endpoint that it should trust. When the proxy then creates a connection, it will present the trusted certificate to the user. This way the proxy can analyse the connection between the two, without warning the user that something is wrong. The problem with this approach is that it requires the endpoint to use SSL/TLS 1.2 or below for its encryption protocol, as TLS 1.3 and beyond use a new encryption protocol, which stops this kind of interception. And at some point, most web servers will support TLS 1.3.

Now that we know the purpose of a web proxy. Let's take a closer look at a specific web proxy. For this example, we will look at web proxy by the name of Squid,[6] it is an open-source web proxy, that can run on most hardware and virtualisation platform. It supports both transparent and explicit mode deployment.

Explicit web proxy requires endpoints to be configured to use the proxy. Whereas transparent proxy sits inline on the network and does not require the endpoints to be aware of its existence. To get the full functionality of an explicit proxy, you must block all other attempts to access websites that don't go through the web proxy. This ensures that all web connection goes through the proxy. Forcing connections through the web proxy allows you to detect connections from unauthorised, misconfigured or malicious endpoints. This approach alone is enough to stop most malware, as they are not proxy aware and tried to access the web without using the proxy, resulting in a failed connection. Proxy awareness means that the program is designed to use the proxy settings from either the operating system or allow the user to manually specify proxy settings within the application. With all the configuration and setup that is required for deploying a proxy in explicit mode, comparing that to deploying a web proxy in transparent mode that just sits inline makes it relatively easier to deploy web proxy in transparency mode and less of a headache to maintain.

Another configuration to consider is to have the web proxy be in authenticated or unauthenticated mode. The usage of authentication means that the client using the proxy must prove who they are before they are authorised to use the proxy. In unauthenticated mode, any device can use proxy, defaulting to trust every endpoint. Unauthenticated is often the default rollout for many web proxy deployments.

The combination of authenticated and explicated can create a strong layer of protection, as all the out-going connection not only have to go through the proxy but also have to be authenticated. In order to be successful, a malicious connection must steal the system's credentials, use them on the proxy and then make an outbound internet connection. Squid also comes with other capabilities of performing antivirus checks using ClamAV, web caching and optional SSL interception. This uses an Internet Content Adaptation Protocol (ICAP) server. You can find a list of supported ICAP servers here.[7]

Those are the basic features of Squid proxy, and it is free. That also means that commercial web proxies are required to be more feature-rich and mature compared to Squid. The ideal is to use an explicit web proxy. The problem is not all assets have the ability of using web proxies. Devices such as IoT devices might not have this capability. In these cases, you must set up the proxy as a transparent web proxy and have the IoT be on a separate network segment where all traffic goes through the proxy. For devices, such as smartphones, that constantly enter and leave the network, it can be more challenging. These mobile devices require a means to access the proxy, no matter where they are. We can achieve this using a cloud-hosted proxy service or a DMZ internet-facing web proxy.

5.7 Summary

We have now implemented network segmentation in the Juice Factory environment by dividing the Juice Factory network into DMZ, Production and Corporate segments. This way, we have limited the impact of a cyber incident in one part of the network affecting one of the others. This is, of course, no guarantee that we will be 100% secure. It is just a single step for us in the right direction of Zero Trust. We changed the configuration file of our Docker Compose file to accomplish this. In a real-world scenario, we would typically configure our firewalls, routers or switches to achieve this. We also talked about general network security you have to be aware of. This includes IPv6 and the usage of web proxy. Remember that network segmentation is one of the fundamental security practices that can enhance the security posture of an organisation by isolating different parts of the network from each other.

Notes

1 https://radvd.litech.org/

2 https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-119.pdf

3 https://en.wikipedia.org/wiki/OSI_model

4 https://learn.microsoft.com/en-us/azure/firewall/web-categories

5 https://majestic.com/reports/majestic-million

6 https://www.squid-cache.org/

7 https://www.squid-cache.org/Misc/icap.dyn

Chapter 6
Network Monitoring

> You are presenting to the C-suite how we redesigned the network and split it into three zones: DMZ, Production, and Corporate. Just as you are about to change slide, presenting the next project for Zero trust. The CFO asks "Does that mean we are now "Zero trust"?", you click to the next slide and tell to them "Zero trust is a mindset of continuous improvement. And there are still many things that need to be done." You continue the presentation and a new slide pops up with the header "You can't protect what you can't see" a slide on how monitoring is one cornerstone of Zero Trust. On how you would like to implement network monitoring which requires new hardware, software and people to operate the monitoring tools, looking for unusual activity. The board agrees, and you set out to implement network monitoring.

We have now segmented the network into three different zones: DMZ, Production and Corporate. This segmentation helps limit the possibility of threat actors moving laterally between assets. The problem we are currently facing is that we do not know what is going on within our network. You might remember from Chapter 2 that monitoring is one of the key design principles of Zero Trust Architecture. It is for that reason that we will, in this chapter, be exploring the importance of network monitoring and the tools and techniques used to set up network monitoring. When it comes to implementing network monitoring in our simulated environment, the goal is to set up Suricata[1] that is going to be monitoring what is going on within our network.

What is it that makes network monitoring so important? It provides visibility into the activities that are happening on the network. Without this visibility, it is impossible to identify what normal looks like on our network. And knowing what normal activity looks like is required for creating detection rules that alert when any suspicious network activity is happening. Without this kind of knowledge, it would be like finding a needle in a haystack, and the problem is network data can quickly become a very large haystack. Alerts are often just the starting point for any investigating incidents. Without alerts, the discovery of incidents would only occur after threat actors had encrypted the entire environment. The critical aspect of network monitoring is the following:

a. Visibility and Insight: Continuous monitoring provides real-time visibility into network traffic, enabling security teams to identify anomalies and potential threats.
b. Threat Detection: By analysing network traffic, monitoring tools can detect indicators of compromise (IOCs) and alert the security team to possible malicious activities.
c. Incident Response: Monitoring helps in the rapid detection of incidents, allowing for a swift response to mitigate potential damage.
d. Compliance: Many regulatory frameworks require continuous network monitoring to ensure data security and integrity.

6.1 What Traffic to Monitor?

We have come to an agreement that having visibility is important. Now you might wonder, well, should we just be monitoring all the network traffic that is happening within our network? The short answer is yes, if you have the resources to collect, store and analyse the data that is being generated. I assume you do not have the resources for handling the data stream being generated by your entire network. In those cases, I would recommend focusing on traffic you know attackers are going to use. The question is what network has the highest probability of being used by threat actors? For that reason, I select monitoring the egress traffic of the organisation. This is the network traffic that is leaving the organisation heading towards the internet. The reason for monitoring this traffic above anything else is:

a. Detecting Malware and Their Communications: All malware and Command and Control (C2) communications require external connections to operate. By monitoring egress traffic, we can identify and block malicious outbound connections.
b. Data Exfiltration: Monitoring outbound traffic helps detect attempts to exfiltrate sensitive data from the organisation.

As you can see, it is almost impossible for threat actors not to generate egress traffic.

6.2 Techniques for Monitoring

Now we have to figure out what part of the network we are going to be monitoring. The next question is how are we going to access and store the network packages? Let's first look at how we can get access to the network packages. For monitoring the network, the most commonplace to do monitoring is at junction points. Which are places on the network that traffic has to pass through. This is often at switches and firewalls. For capturing the traffic at these junction points, there are two primary techniques used, either with switch port analyser (SPAN) port or network Test Access Points (TAPs). The selection of the correct junction point to monitor is crucial for both methods, as this determines which traffic gets captured.

To give an example of how the placement can affect the data that is collected. It is what side of the firewall you should monitor on. Where you have to place the monitoring on the internal side of the firewall. That would result in not having the ability to see the traffic that is coming from the internet and is being dropped by the firewall. Conversely, placing the monitor outside the firewall means all outbound traffic will show the firewall's IP address because of Network Address Translation (NAT). Therefore, you cannot distinguish the originating local machine. That is just one example of why being aware of the placement for the monitoring system is crucial. The best approach when setting up your monitoring is to verify that the expected traffic actually gets captured. After finding a suitable placement, the next step is to choose how you will send network packets to the monitoring device. So, let's go into a little more details about the differences between SPAN port and network TAPs.

6.2.1 Switch Port Analyser

SPAN, also known as port mirroring, is a feature on network switches that allows the switch to duplicate the traffic coming from one or more switch ports to a designated mirror port. Figure 6.1 shows the network flow of a port mirror. This method is quite easy to set up, and the switch you already have in your environment should already have this feature built-in. The problem arises when there is a large amount of traffic flowing through the switch, because SPAN ports have a lower priority than other ports. The switch will begin to drop packets destined for the mirror port, thus preventing the monitoring tool from receiving them. This problem worsens when the SPAN must aggregate multiple ports to a single SPAN port, causing oversubscription.

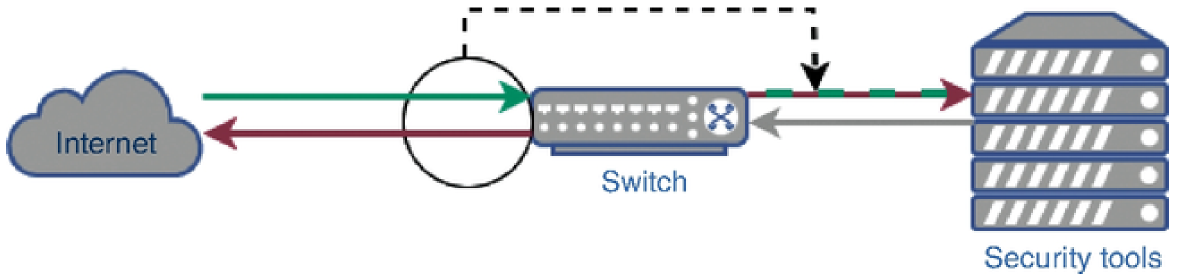


Figure 6.1 Network SPAN.

Here are some of the pros and cons of choosing to use port mirroring to collect network packages.

- Pros:
  - Easy to configure on most managed switches.
  - Can monitor multiple ports simultaneously.
- Cons:
  - Potential for packet loss under high traffic conditions.
  - Limited by the switch's capacity to handle mirrored traffic.

You can easily configure an SPAN port on a Cisco switch using the Cisco terminal. Using the following commands. The first step should list the virtual local area network (VLAN) interfaces that are available. We need to select one to monitor.

```
sh vl br
```

Next is the setup for monitoring the VLAN. For that, we will use the monitor command to collect NetFlow data. By default, it will collect both egress and ingress traffic.

```
monitor session 1 source vlan <vlan id>
```

Next, you also have to set up a destination for the monitoring data. Here, we will send the data to an interface.

```
monitor session 1 destination interface <interface>
```

There should then be a collection tool at the end of that interface that will monitor the traffic.

6.2.2 Network TAPs

The other method of sending network traffic to the monitoring tools is through the use of network TAP, which is a hardware device that connects directly to the network wire and provides a way to access the data flowing across. Network technicians install these TAPs inline on the network; an internal splitter creates a copy of the traffic, which the monitor tool receives without disrupting network flow. Here are the pros and cons of using network TAPs.

- Pros:
  - Reliable, with no risk of packet loss.
  - Does not affect network performance.
- Cons:
  - Requires physical installation.
  - More expensive compared to using SPAN ports.

Figure 6.2 is an example of a network TAP. The good thing about TAP is that they work passively and they do not require a power.

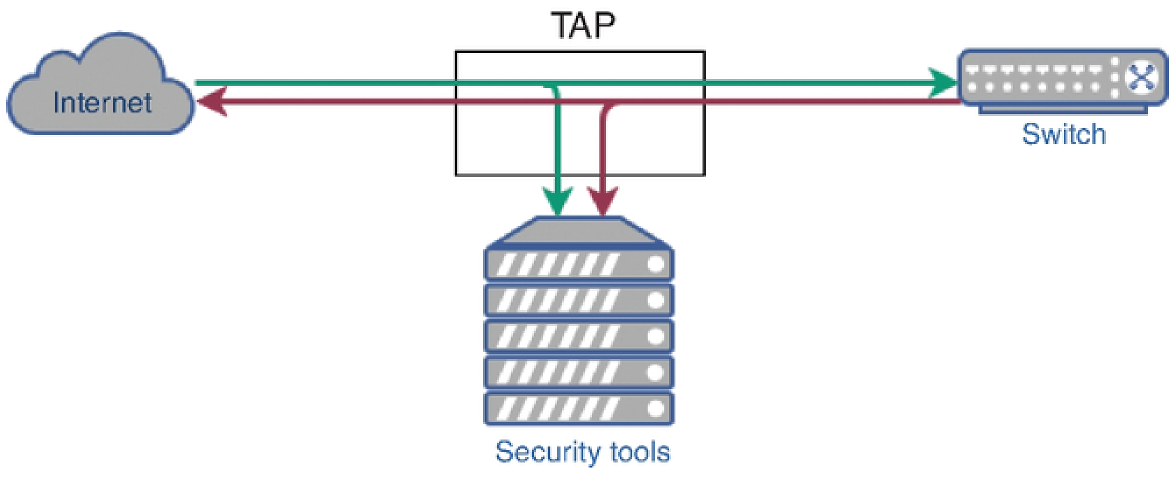


Figure 6.2 Network TAP.

6.2.3 Agent Based

There is also a third option that differs from the other two, which is agent based. This type of monitoring works by having a small program installed on the device. This program then sits on the device and constantly monitors all the network traffic that is happening on the device. The reason endpoint agents have become more prolific is the increase in computing power, allowing them to handle having an agent running in the background without affecting the user experience. With agents, you no longer have to consider the placement of an agent, such as with SPAN or TAPs. All that is required is to deploy these agents to all the devices on the network, and you will start collecting all the networks that are going on the device. Maybe the most important reason for adopting agents is because of network encryption, as most traffic that is happening today is encrypted by default. This makes it impossible to identify what is happening within the network package. Using an agent can help mitigate this problem as they will hook into the encryption library used by the operating system, allowing it to monitor the unencrypted traffic. A later chapter will explain the Extended Detection and Response (XDR) agent, which includes this feature. It has become a standard deployment on most networks. This means that agents are becoming one of the most common ways to monitor the network. Of course, this does not mean it is not without having its own disadvantages.

- Pros:

    - Reliable, collects all the traffic on the network.
    - Monitoring encrypted traffic.
- Cons:
    - Can be shutdown by an attacker.
    - Requires the data to be sent to a centralised storage for analysis.
    - Might not be able to deployed of all devices, like IoT.

6.2.4 Handling Encrypted Traffic

As I alluded to earlier in the chapter, encrypted network traffic has become the standard in most places, making it difficult for us to identify what is happening within the network packages. According to studies, 86% of all webpages use HTTPS as the default protocol, and with DNS over HTTP (DOH) also becoming a default standard, we will soon not have the ability to identify what is going on within our environment internet wise.

Currently, with TLS 1.2, it is possible to set up an SSL proxy that would generate a SSL certificate for the site on the fly that was trusted by the system. This way, it was possible to decrypt the traffic and inspect what was going on within the packages. This is no longer possible with TLS 1.3, which was released in 2018, with an adoption rate of 60% when looking at the top 1 million most visited websites by the end of 2021. TLS 1.3 enabled a feature called complete forward secrecy, making the encryption ephemeral, or to put it more simply. Each session generates its own encryption key, making it temporary. Meaning the solution must now be able to retrieve the encryption key, which is never sent over the network and instead calculated on both ends of the network connection.

Another concern is privacy regulation. It might not be legal to decrypt the traffic and inspect the network traffic. And for that reason, I recommend talking with the legal before implementing any deep package inspection. Skipping the regulation concerns for now, what option do you have as a security engineer? The first is forcing all traffic to TLS 1.2 or below. That way, it is still possible with an SSL proxy to decrypt the packages. The other is the usage of agents on endpoints, which we described previously that hook into the encryption library of the device and decrypts the traffic. Eventually, TLS 1.2 will cease to be supported, leaving only the agent approach for decrypting network traffic.

6.2.5 The Different Type of Network Collecting Format

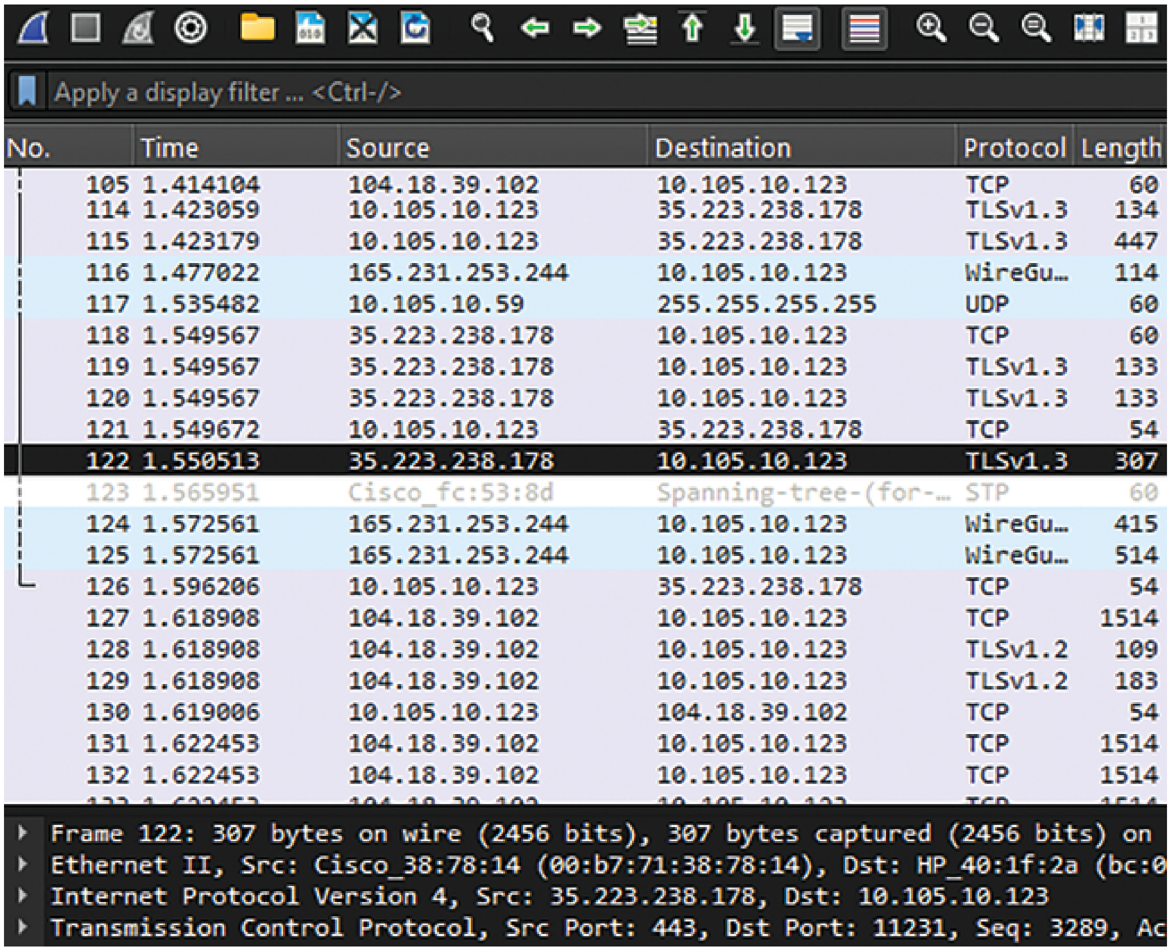
Whether you choose to set up a TAP or a switch with a SPAN port. We have now looked at different methods of sending the data to a monitoring system. The next step is to figure out which monitoring solution to use. To answer this question, we first need to understand the different approach of working with network packages. The most common include:

- Full package capture (PCAP)[2]
- NetFlow traffic[3]
- Logs

What formats to choose depends on the use case, storage and much more. Let's go into a little more detail about each type of data to understand the advantages and disadvantages of each approach.

6.2.6 PCAP

PCAP is the most complete capture of them all, as it is a copy of the entire network packets. This lets us inspect the content of every captured packet, unless it's encrypted, allowing us to reconstruct the network traffic that has happened on the network. We can even extract any files sent over the network. This allows security analysts to see everything that is happening on the network, giving them great visibility of what is happening on the network, especially for incident response. It is rare to see organisations that have set up PCAP. The reason is cost and the lack of knowledge and skills for how to capture and analyse the package data. Then there is also the problem of size. When capturing the entire network package for an organisation, there is a lot of data being generated. Storing all of this data can be difficult. That is why some organisations only store data for X amount of data, and when the store is full, the oldest data gets overwritten. Another concern is privacy of the employees as security personnel can see what they are doing, whether it is business-related or personal matter. The most common tool for capturing PCAP is Wireshark,[4] pictured in Figure 6.3. Tshark is the CLI version of Wireshark; another CLI option is Tcpdump.[5] These tools are all PCAP and analysis tools. Wireshark is the only GUI-based, and Tshark and Tcpdump are command-line. These tools all have each of their own pros and cons.



```
No.   Time      Source             Destination           Protocol Length
  105 1.414104  104.18.39.102      10.105.10.123         TCP        60
  114 1.423059  10.105.10.123      35.223.238.178        TLSv1.3   134
  115 1.423179  10.105.10.123      35.223.238.178        TLSv1.3   447
  116 1.477022  165.231.253.244    10.105.10.123         WireGu…   114
  117 1.535482  10.105.10.59       255.255.255.255       UDP        60
  118 1.549567  35.223.238.178     10.105.10.123         TCP        60
  119 1.549567  35.223.238.178     10.105.10.123         TLSv1.3   133
  120 1.549567  35.223.238.178     10.105.10.123         TLSv1.3   133
  121 1.549672  10.105.10.123      35.223.238.178        TCP        54
  122 1.550513  35.223.238.178     10.105.10.123         TLSv1.3   307
  123 1.565951  Cisco_fc:53:8d     Spanning-tree-(for-…  STP        60
  124 1.572561  165.231.253.244    10.105.10.123         WireGu…   415
  125 1.572561  165.231.253.244    10.105.10.123         WireGu…   514
  126 1.596206  10.105.10.123      35.223.238.178        TCP        54
  127 1.618908  104.18.39.102      10.105.10.123         TCP      1514
  128 1.618908  104.18.39.102      10.105.10.123         TLSv1.2   109
  129 1.618908  104.18.39.102      10.105.10.123         TLSv1.2   183
  130 1.619006  10.105.10.123      104.18.39.102         TCP        54
  131 1.622453  104.18.39.102      10.105.10.123         TCP      1514
  132 1.622453  104.18.39.102      10.105.10.123         TCP      1514

▸ Frame 122: 307 bytes on wire (2456 bits), 307 bytes captured (2456 bits) on
▸ Ethernet II, Src: Cisco_38:78:14 (00:b7:71:38:78:14), Dst: HP_40:1f:2a (bc:0
▸ Internet Protocol Version 4, Src: 35.223.238.178, Dst: 10.105.10.123
▸ Transmission Control Protocol, Src Port: 443, Dst Port: 11231, Seq: 3289, Ac
```

Figure 6.3 Wireshark.

One such disadvantage with Wireshark is the limitation of the file size it can process. You can overcome this limitation by first filtering out the unwanted traffic with Tshark or Tcpdump, thus reducing the file size to one Wireshark can load for further analysis.

6.2.6.1 Black Box PCAP Flight Box

Having package capture of the network in incident response is important as it allows us to reconstruct what has happened on the network, and in case of a cyber attack, it allows us to go back in time and see in detail what happened on the network. This is where a network 'black box' comes into the picture. Think of it like a flight black box on aeroplanes. It is a server that is constantly recording PCAP of the network, placed at a key location on the network that allows it to record the most useful traffic for incident response.

The server storage should be a couple of terabyte in size and with Tcpdump installed. You will then setup a span port or network TAP to route traffic to the black box. You then use Tcpdump to listen to all incoming traffic. Here is an example of a black box using Tcpdump, which is listening on a specified interface and will create 10 files of network packets, each with a file size of 1 GB. After creating 10 files, the program overwrites the oldest packets, continuing until Tcpdump stops.

```
tcpdump -ni <interface> -W 10 -C 1000 -w <filename>
```

This will give you 10 PCAP with a size of 1 GB of data to analyse in case of an incident response. This is just an example; you should modify the command to fit your organisation needs.

6.2.7 NetFlow

NetFlow is a protocol that captures the metadata about network traffic. The creators of NetFlow designed it for network engineers to troubleshoot network problems, not for forensic purposes. This, of course, does not stop us from using them for forensics' purpose. The data stored in NetFlow includes IP addresses, ports, timestamps and much more. Which can be useful for understanding what activities are happening on the network, like detecting anomalies and identifying potential security threats. What NetFlow does not capture is the content of the packets themselves. Which can be beneficial if privacy is a concern with monitoring employees or if there is limited storage. The source of NetFlow comes from routers, switches and other network devices that can typically export NetFlow data. These are some key fields in a NetFlow record:

- Source and destination IP addresses
- Source and destination ports
- Protocol (Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP), etc.)
- Timestamps (start and end times of the flow)
- Number of packets and bytes exchanged

I have yet to find a good tool for analysing NetFlow data; what I do when working with NetFlow data is convert the NetFlow data into CSV format, which then gets loaded into Python using Pandas[6] Dataframe, and from here I can analyse the data as I please.

6.2.7.1 NetFlow Collection

When setting up collection of NetFlow, you need careful planning for which devices send NetFlow traffic to ensure that you do not get duplicate rows. You should also record the flow direction you are monitoring.

- Unidirectional is having each direction of a network connection be recorded in separate rows. That means for each connection you will have two rows, one from source -> destination and another row from destination -> source.
- Bidirectional is the recording of source <-> destination in both directions in a single row.

On a Cisco router, you can set up NetFlow using the following command:

```
IP flow-export destination <destination port on the
collector> 2055
```

The router will export all flows to 192.168.1.1 with the destination UDP port 2055. It is also possible to define which version of NetFlow you want to use. If you want to use version 9, you can use the following command.

```
ip flow-export version 9
```

We also need to specify which interface we will be listening to.

```
interface <interface>
```

You should now collect NetFlow data off the Cisco router and having them sent to a NetFlow collector such as Ntop.[7]

6.2.8 Logs

Logs are records of events or network connections that have occurred on the system and can come from many sources. A good thing about logs is that they are often enabled by default. The most common places to find logs related to network traffic include:

- Firewall logs
- Intrusion Detection/Prevention System (IDS/IPS) logs
- Proxy logs
- DNS logs
- Web server logs

The problem with logs is that they can vary in format and structure, depending on the system generating them. The most common log formats include:

- Syslog[8]: A standard log format used by many network devices and Unix-based systems. Syslog messages contain a timestamp, a facility (source), a severity level and a message.
- W3C Extended Log File Format[9]: A standard log format used by web servers, such as IIS and Apache. It contains information about client requests, server responses and other transaction data.
- Windows Event Log: The native logging system in Windows operating systems. It records events from various sources, including the operating system, applications and security components.

Another downside of logs is that they have already analysed and parsed them into an output format. Making it impossible to go back in time and look at the packages for what have happened. Take Snort,[10] as an example; it is an Open Source Intrusion Prevention System (IPS). Snort uses rules that define what malicious network activity looks like, and it then uses those rules to find packets that match against them and generates alerts for the users to respond to. A problem arises when a new attack pattern is discovered, as we cannot retroactively apply rules. Making it impossible to create a new detection rule and have it check for past attacks. The only way to have Snort re-analyse the traffic is to resend the packages through Snort and have them analysed again. When it comes to the process of analysing logs, it is to take them and parse and normalise them into a consistent format. From here, you can import the data into an analysis tool, such as the ELK stack.

6.3 Implementing Network Monitoring with Suricata

Now that we understand the different approaches of monitoring network packages; before we come to the implementation phase of network monitoring, let's design how we would like to implement the network monitoring for Juice Factory. Then update both the network diagram and transaction map to fit the new design.

Because of the limitations of the docker environment, there are a few options available for placing the monitoring while also having it be easy to deploy. Then there is also the choice of what tool for collection and parsing of the network packet. For our project, we will use Suricata to monitoring the host machine's egress point. This setup allows us to monitor all outbound traffic from the host machine, which will also include the docker environment. In the network diagram in Figure 6.4, the network monitoring tool is placed at the egress point of our environment.

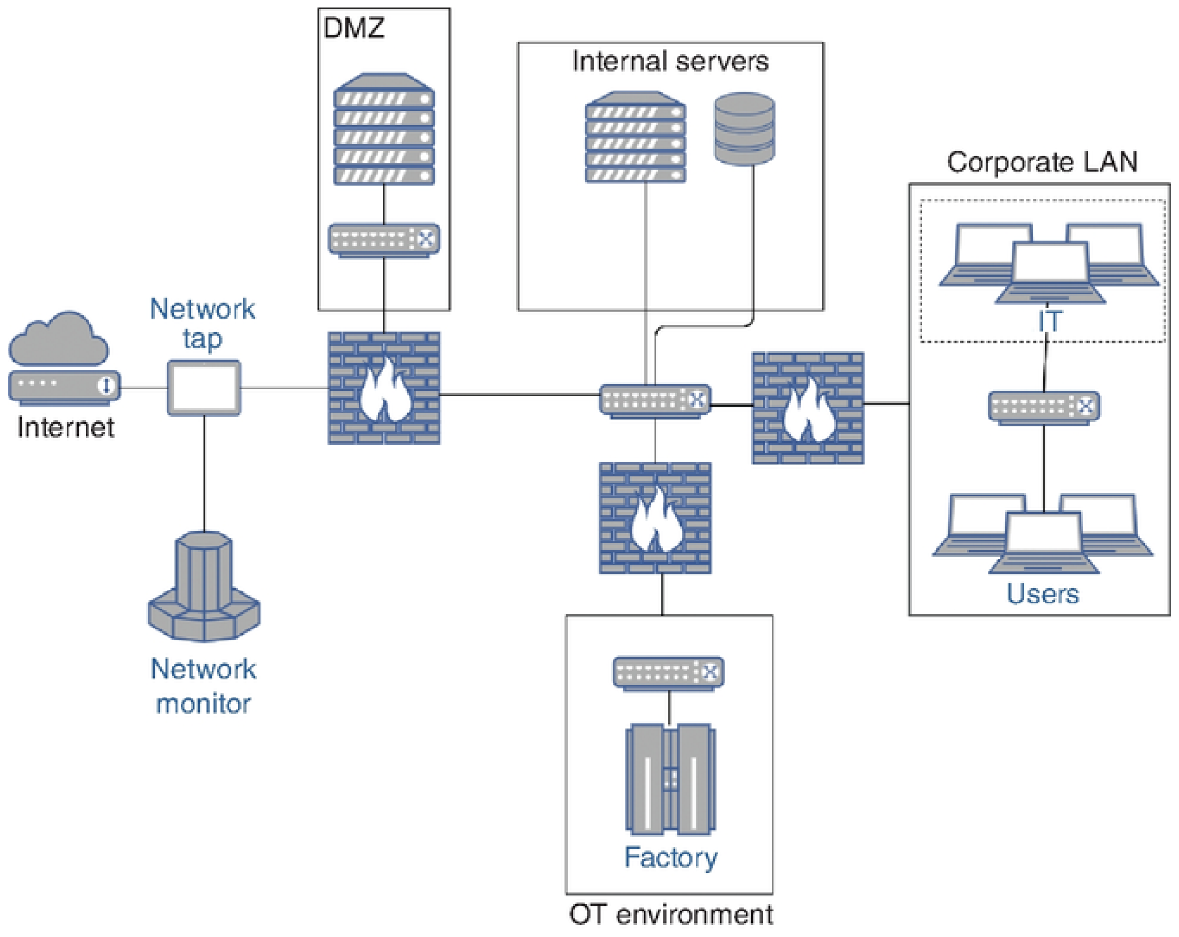


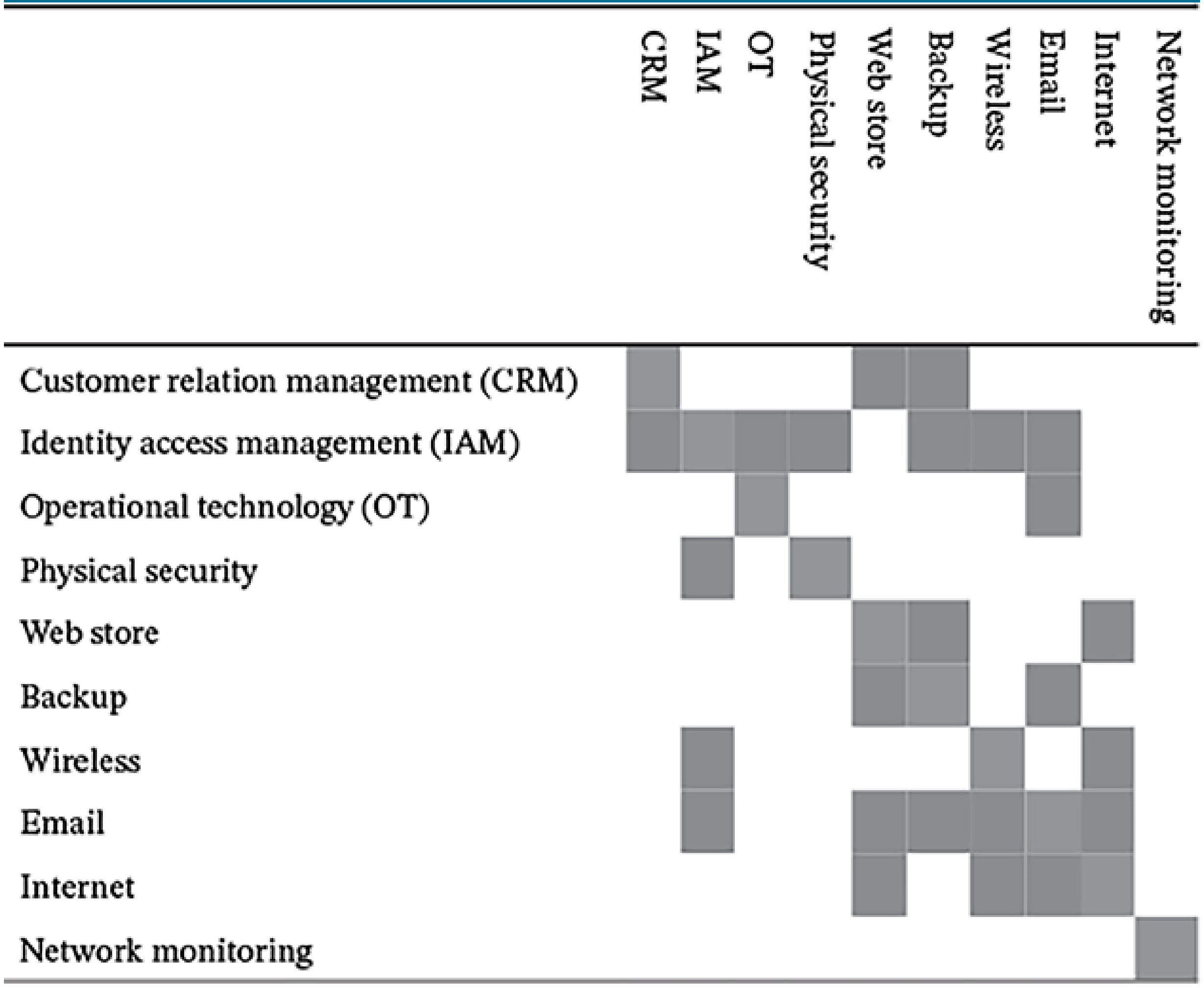
Figure 6.4 Network diagram.

Let us also update the transaction map to include the network monitoring we are going to implement.

As you can see in the transaction map in Table 6.1, at the moment the network monitoring tools do not interact with anything; this is because of the out-of-bound design of the setup. In a later chapter, we will send the logs to a centralised logging system called SIEM. For our network monitoring project, we will utilise Suricata. It is a powerful and versatile tool for network monitoring. Suricata is an open-source network threat detection engine renowned for its ability to perform real-time network intrusion detection system (IDS) and network security monitoring (NSM). It operates by inspecting network traffic and analysing the data for signs of potential threats or malicious activities and alerting the user. Allowing the user to react to a wide range of cyber attacks.

TABLE 6.1

Transaction map.



	CRM	IAM	OT	Physical security	Web store	Backup	Wireless	Email	Internet	Network monitoring
Customer relation management (CRM)	■				■	■				
Identity access management (IAM)	■	■	■	■		■	■	■		
Operational technology (OT)			■					■		
Physical security		■		■						
Web store					■	■			■	
Backup					■	■		■		
Wireless		■					■		■	
Email		■			■	■	■	■	■	
Internet					■		■	■	■	
Network monitoring										■

One of the key strengths of Suricata is its flexibility and comprehensive feature set, which makes it suitable for both small- and large-scale network environments. Additionally, Suricata can integrate with various other security tools and platforms, enabling piping the data into a centralised log system. This allows for robust and cohesive monitoring strategy.

While there are other noteworthy network IDS tools available, like Zeek (formerly known as Bro) and Snort. Both these tools are powerful network analysis tools with its own set of capabilities and advantages. The advantages of Snort are that it is one of the most well-known IDS; this means that most IDS solutions accept its rule format. This also includes

Suricata. That means that the detection rules we will look at in a moment also work for Snort. There are many network monitoring tools that exist on the market. I therefore recommend that you to look at the different solutions for yourself, to understand the differences between them all and the pros and cons for each. In this book, I have chosen to focus on Suricata.

To maintain a simple approach, we will be concentrating on monitoring all traffic coming from the host machine, as this will also include the docker environment. This approach ensures that we can effectively observe and analyse all network communications happening. While it is technically feasible to monitor the network traffic occurring between individual Docker containers. Such an in-depth and granular approach is beyond the scope of this book. Monitoring inter-container traffic involves additional complexities and configurations that could detract from our primary goal of providing a high-level approach to the different elements of Zero Trust Architecture. For us to monitor the network of our environment, it means we have to change the Docker Compose Configuration to allow for network monitoring:

```
suricata:
  image: jasonish/suricata:latest
  container_name: suricata
  network_mode: host
  volumes:
    - ./suricata/logs:/var/log/suricata
    - ./suricata/rules:/var/lib/suricata/rules
    -
./suricata/suricata.yaml:/etc/suricata/suricata.yaml
  command: -i eth0
  tty: true
  stdin_open: true
  restart: unless-stopped
```

In Docker, network monitoring can be challenging because of the abstraction and isolation provided by containers. For this project, we will use the Jasonish/Suricata image to monitor the network traffic as it comes with Suricata pre-installed. That is not all. Specific Docker settings also need to be configured. These settings ensure that the container has the permissions and network access to monitor the network. Let's delve into the settings that need to be configured.

The first setting is 'network_mode: host'. This allows the Suricata container to share the host's machine network stack. Providing the container with direct access to the host network interfaces, bypassing the Docker network isolation. As a result, Suricata now has the ability to monitor all network traffic that is accessible to the host machine. However, it's important to note that using 'network_mode: host' can have security implications, as it provides the container with elevated network privileges. You should use 'network_mode: host' cautiously and only when necessary. The command '-i eth0' tells that Suricata should capture traffic from the eth0 interface.

6.3.1 Working with Suricata

Suricata operates by using a set of rules that define how it will react to the traffic it sees. Configuration files hold these rules. Suricata uses the Yet Another Markup Language (YAML) format for its configuration file `suricata.yaml`.[11] The `rule-files` setting in this configuration file determines which rule files Suricata imports. Here is an example of configuring Suricata to import multiple rule files.

```
rule-files:
  - suricata.rules
  - local.rules
```

When it comes to the creation of detection rules[12] for Suricata, they all follow the same structure of:

```
Action [transport protocol] ip port <-> ip port
(msg:"log message";
      flow:to_server; app-layer-protocol:protocol;
sid:id; rev:1;)
```

We can split the Suricata rules structure into three parts.

a. Action – specifies the action to take if the rule matches a network package.
b. Protocol – IP addresses, ports, and direction of the connection that the rules should match on.
c. Additional rule options that define the specifics of the rule.

To make it more clear, let’s look at a real example, using the following rule.

```
alert tcp any any -> any !22 (msg:"SSH connection to
non-standard
    port"; flow:to_server; app-layer-protocol:ssh;
sid:1000001;
    rev:1;)
```

This rule will create an alert when there is an outbound Secure Shell (SSH) connection that is going to a port other than 22, which is the default port for SSH.

For the rule, `alert` is the action Suricata will take if it matches against any network packages. `tcp any any -> any !22` are the header information Suricata is going to be looking for. This rule is looking for any TCP connection coming from any IP and port going outbound to any IP address that is not on port 22. The last part of the rule `msg:"SSH connection to non-standard port"; flow:to_server; app-layer-protocol:ssh; sid:1000001; rev:1`. Upon matching, the rule should alert with the message ‘SSH connection to non-standard port’. The `app-layer-protocol` field identifies the protocol being monitored, which is SSH in this case.

That is just a single detection rule for Suricata. You might have noticed with this type of granularity, the systems will quickly reach hundreds of rules. Far too many for them to be managed manually. Which is why having tools to help with rule management is important. For that reason, I would like point to a project from Google called PulledPork.[13] It is a Perl script that automates downloading of rules and importing rules into Suricata. PulledPork uses a combination of unique signature IDs, SIDs and regex to disable, change or enable rules. This helps keeping all the rules up-to-date, without having to manually go through them.

```
./pulledpork.pl -o /usr/local/etc/Suricata/rules/ \
-u http://www.example.org/reg-
rules/Suricatarules.tar.gz \
-i disablesid.conf
```

The above will fetch the Suricatarules.tar.gz from example.com and put the rules files from the tarball into the output path '/usr/local/etc/Suricata/rules/' the '-i' parameter tells pulledpork where the disablesid.conf is located; this file is used for disabling rules by their unique signature ID.

The alerts that are being generated by Suricata are written to the JSON file called `eve.json`. This file is stored inside the folder `suricata/logs`. The system formats the alerts as JSON, a flexible and widely used format ideal for storing and exchanging structured data, and allows for easy integration with other systems that can take the JSON data for further analysis or visualisation. The following is an example of an alert generated by Suricata:

```
  "timestamp": "2024-07-15T06:38:54.654889+0000",
  "flow_id": 1952924432853292,
  "in_iface": "eth0",
  "event_type": "http",
  "src_ip": "192.168.65.1",
  "src_port": 63898,
  "dest_ip": "192.168.65.7",
  "dest_port": 2375,
  "proto": "TCP",
  "pkt_src": "wire/PCAP",
  "tx_id": 0,
  "http": {
      "hostname": "api.moby.localhost",
      "url": "/v1.45/containers/json?all=1",
      "http_user_agent": "Docker-Client/26.1.4
  (windows)",
      "http_content_type": "application/json",
      "http_method": "GET",
      "protocol": "HTTP/1.1",
      "status": 200,
      "length": 18940
  }
```

Each alert generated in the eve.json file contains several fields, each providing specific information about the detected event. Here's a breakdown of the key fields in the provided example:

- **timestamp**: The date and time the alert was generated.
- **flow_id**: A unique identifier for the flow of packets associated with this alert.
- **in_iface**: The network interface source of the packet that triggering the alert (e.g., eth0).
- **event_type**: Describes the type of event detected.
- **src_ip**: The source IP address of the package
- **src_port**: The source port.
- **dest_ip**: The destination IP address.
- **dest_port**: The destination port.

- **proto**: The protocol used in the traffic (e.g., TCP).
- **tx_id**: A transaction ID that helps in identifying and correlating related transactions or requests within the same session.
- **http**: A nested JSON object containing specific details about the HTTP event.

By parsing and analysing these alerts, network administrators and security professionals can gain valuable insights into network activity, detect potential security incidents and allow us to take actions to protect our networks.

6.4 Blackhole and Darknet

When you google darknet, it would most likely direct you to website explaining what Tor is or news articles about criminal activities happening on the darknet. When we are talking about darknet in this section, we are referring to the unused IP address block within your network.

It is common for organisations to ignore the network blocks they do not use. This is a missed opportunity for monitoring the network. A better usage of this is to route all the traffic going to the unused part of the network to a Blackhole. This Blackhole is a monitoring system that will analyse the network headed for the unused IP range. The reason we would like to do this is that malware like to scan the network looking for opportunities to exploit. This can cause packages being sent to the unused part of the network. Having a Blackhole monitoring system allow us to detect scanning behaviour. This is basically a win-win situation, as all traffic sent to the Blackhole is bogus by nature. Only mis-configured or malicious traffic should go to the Blackhole. In the case of mis-configured traffic, it is an opportunity to tune the network to optimise the flow of the network. For malicious traffic, it’s an attack that has just been detected.

When it comes to designing this kind of monitoring system. The first thing we need is a dedicated router. It can be any lower-end router. The router’s sole function is forwarding packets to the Blackhole interface, similar to Figure 6.5. The Blackhole system needs two interfaces. One for the sniffing interface and another for managing the server. For the monitoring tool, things like Suricata, Snort, Zeek and many others are all good options. The management interface can be anything you are comfortable providing remote access to the server like SSH.

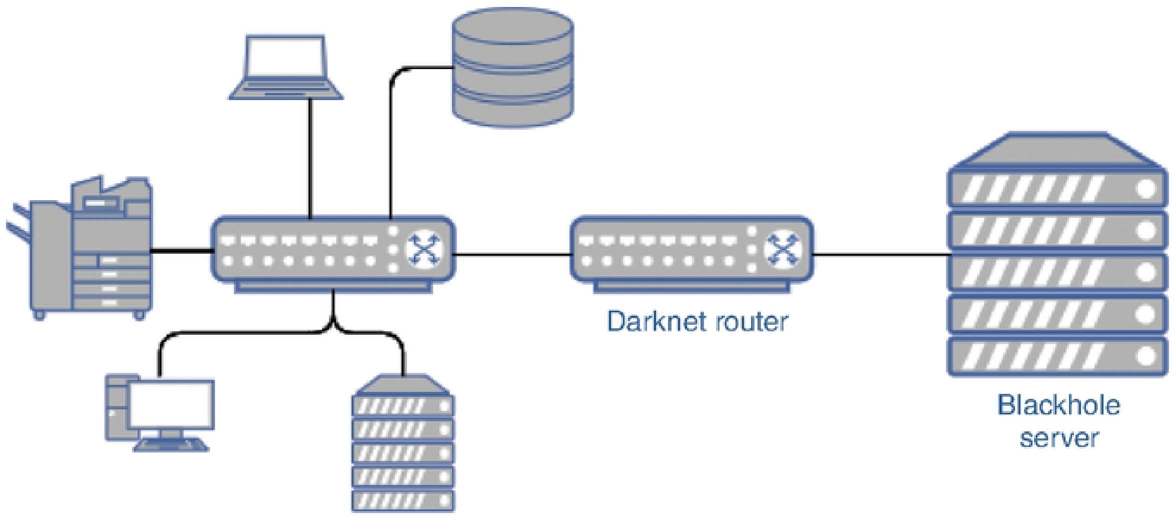


Figure 6.5 Darknet architecture.

6.5 Summary

Network monitoring is a vital aspect of maintaining security in a Zero Trust Architecture. By continuously monitoring network traffic, particularly egress traffic, organisations can detect and respond to threats, ensuring the safety of our systems and data. In our project, we leveraged Suricata to monitor network traffic from our host machine, providing us with insights and protection against potential threats. In a real-world scenario, more comprehensive setups involving network TAPs or SPAN ports would ensure full visibility and minimal impact on network performance. Remember, monitoring is just the first step of the process. Having a response plan to any alerts being generated by our monitoring system has equivalent importance; there is no point in going through the process of setting up network monitoring if none is going to be looking at it.

Notes

1 https://suricata.io/

2 https://en.wikipedia.org/wiki/PCAP

3 https://www.cisco.com/c/dam/en/us/td/docs/routers/asr920/configuration/guide/netmgmt/fnf-xe-3e-asr920-book.html

4 https://www.wireshark.org/

5 https://www.tcpdump.org/

6 https://pandas.pydata.org/

7 https://www.ntop.org/

8 https://en.wikipedia.org/wiki/Syslog

9 https://www.w3.org/TR/WD-logfile.html

10 https://www.snort.org/

11 https://docs.suricata.io/en/latest/configuration/suricata-yaml.html

12 https://docs.suricata.io/en/latest/rules/index.html

13 https://github.com/shirkdog/pulledpork

Chapter 7
Identity Access Management and Jump Box

From your office window, you can see the production facility that serves as Juice Factory's crown jewel. As CISO protecting these critical systems keeps you up at night. You know that identity and access management will be crucial for securing the environment, but implementing it properly requires a deep understanding of how the engineering team uses these systems. With this in mind, you head to the engineering department, determined to learn how they work and what controls would best protect their operations without hindering productivity.

This chapter is about dealing with the trust we put into the authentication and identity processes. How can we use the idea of Zero Trust to help reduce this risk inherent in identity access management (IAM)? Remember, the whole idea of Zero Trust is the processes of removing trust from systems and processes. How do we apply this idea to authentication and identity? Before we can go deeper into this question, we first need to have a basic understanding of what IAM is, and then we will go over how we can use Zero Trust to help secure the environment.

7.1 Identity Access Management

IAM is the process of verifying the identification of the person trying to get access to the system. Upon successful verification, it grants the user privileges according to their account's access level.

There are many ways identities can be managed. The one most people are familiar with is the local accounts on your computer. This system allows us to have multiple accounts on the same system, with different privileges. The way a computer verifies your identity can vary widely depending on the operating system in question, in most cases, when you try to login to the computer. It checks the username and password provided by you against an internal database, and if there is a match, then it gives you access to the system. For organisations, it is no longer feasible to have each computer handle its own user. This requires an enterprise IAM solution that centralises user management by storing all user and system information in a single database. One of the most used systems for this is Microsoft active directory (AD) and the cloud equivalent Azure AD. These two systems provide a centralised location for system administrators to manage the identification and access of all users in their environment.

One of the major headaches of IAM is how do we verify users' identities? The most common method is a combination of username and password. This type of authentication is called 'Something you know'. The idea of this kind of authentication is that you are the only one supposed to know the password of your account. The problem with this approach is that people are bad at remembering passwords and for that reason are quick to reuse the same password for multiple accounts. And when one of those accounts gets compromised, that then leads to the potential of all accounts sharing the same password being compromised. To solve these problems, we introduced new authentication methods, including 'something you have, like your mobile device' and 'something you are'. Think fingerprint, face and other biometric features. These authentication methods are harder for a threat actor to get a hold of or copy. The good thing is that we did not stop here; instead of just relying on a single authentication method, it has become a standard to use multiple authentication. This is a combination of the three methods of authenticating yourself. You can implement multi-factor authentication (MFA) in many ways, most commonly by combining a username and password (something you know) with an additional factor, such as a time-based code from an authenticator app on your phone (something you have). Let's look at the most common way of implementing MFA.

7.2 Multi-factor Authentication

The idea of two-factor authentication is to identify a user using at least two out of three factors. Early versions of multi-factor systems were clumsy; the two most common types of multi-factor were the smart card combined with a pin or the usage of a time-based code using an RSA secureID token. This system was only used in environments that needed the highest level of security; think governance environments. With the emergence of the smartphone, that changed; now suddenly everybody has a second factor in their pocket. This led to all kinds of innovation method for we can do MFA. For the sake of completeness, let's summarise the different two-factor authentication, how they work and how secure they are.

7.2.1 SMS

When logging into a service or application, you first login with your username and password; it then sends a onetime code to your phone, which you should type into the application. A common question to ask about SMS, is it okay to use text messages as your MFA? The short answer is that using SMS for MFA is not as secure as other options but better than not using MFA. The long version is that threats actor perform an attack called SIM-jacking, a method where they call your phone company pretending to be you. Saying that they just bought a new phone and would like to have your account transferred to a new SIM card, they then put SIM card inside a burner phone. This means, all the MFA messages you would get via text to your phone go straight to the threat actor instead. This type of attack is used against high-value target, as it requires additional effort on the threat actor side.

7.2.2 Email

Works just like with SMS, but instead of sending the code to your phone, it sends it to your email. It's also important to protect your email with MFA.

7.2.3 Authentication Soft Token

Authenticators use an Internet Engineering Task Force (IETF) algorithm to generate onetime codes called time-based onetime passwords (TOTPs). When you want to log into an account. It will ask for a onetime code. You then open the authenticator application on the phone and look up the application you want to log into. On the screen, there is a countdown. Every 30 seconds, the app will generate a new code. You then have to enter the six-digit code before the timer runs out. Any authenticator application can serve as a second factor, as they all use the same standard algorithm.

7.2.4 Push Authentication

Push authentication no longer uses a code. When you need to log into your account, the application pushes a notification to the authentication application on the phone. Then by opening the authenticator and pushing a button that says, 'Yes, that is me', thereby giving you access to your account. This type of authentication is vulnerable to an attack called MFA Fatigue, in which a threat actor continuously sends MFA notification until the victim presses 'Yes', giving the threat actor access to the account.

7.2.5 Passkey

Passkey is the newest kid on the identification block, a standard created by the FIDO alliance.[1] This new method provides passwordless account access, thus preventing passwords from being remembered, stolen or compromised. Instead, you use your phone or another supported device to prove you are who you say you are before giving you access to your account. A lot of things happen in the background. The crucial benefit of passkeys is their immunity to theft and reuse, unlike passwords.

In practices, there are two ways of experiences when using passkey. The first and simplest is when you try to login into an account with a device that has the passkey on the device itself. In these cases, when you try to login, the application should recognise that the passkey is available on the device, and ask if you would like to login your account; by pressing 'Yes', it will prompt for either a pin, fingerprint or face; once it has verified your identity, it will open your account. The other experience is when logging into an account that does not have the passkey on the device. Here there should be an option called login with passkey on the web page; press that button, and it should present you with a QR code that needs to be scanned using your phone. Once that is complete, you should have access to your account.

7.3 Credential Rotation

Having MFA is critical to just meet the minimum level of security, and as I talked about previously, some of the MFA method available do not eliminate the possibility of being compromised. Therefore, we recommend credential rotation; if a threat actor steals credentials, immediate rotation renders those credentials useless. Leaked or reused credentials are one of the most common methods for attackers to get access to systems. The assumption we made is that credentials can and will become compromised; therefore, they need to be rotated. The idea is that credentials have a higher risk of being misused over time. If you were to look at cyber criminal forums, you quickly realise that password dump is common and often also free for everyone to download. With the threat actors armed with the username and password, they can try to get access to your account. This is one of the reasons credential rotation became recommended. The whole idea resulted in the rotation of password at a constant interval. The password policy I have seen most common is every third month. Constant password rotation greatly inconveniences end users, and when inconvenienced, end users find the easiest solution, often changing only the final number, e.g. changing 'Summer2024' to 'Summer2025'.

This approach goes against best practice of NIST 800-63B SP Digital Identity Guidelines: Authentication and Lifecycle Management.[2] This guide from the National Institute of Standards and Technology shows how to create and keep digital identities safe. The document addresses MFA, password storage and much more. For password rotation, they come with the following statement. Verifiers SHOULD NOT require memorised secrets to be changed arbitrarily (e.g., periodically). However, verifiers SHALL force a change if there is evidence of compromise of the authenticator.

This recommendation is based on studies that have shown forcing users to change passwords periodically, resulting in weak passwords or them being written down. It is also important to note that not all accounts are born equal. A better solution is to have fine-grained password policies. This allows password policies to be set per user or group, where accounts with higher privileges have a stronger password requirement than the rest of the organisation.

What about leaked credentials? Online services scan public data for leaked credentials and alert users if their credentials are compromised. The most well known for providing this kind of service is Have I Been Pwned,[3] which is run by Google. Have I Been Pwned also provides domain-wide search service at no cost. Whenever a data breach reveals an email from that domain, Have I Been Pwned will notify you of their discovery, allowing to ensure that the user changes their password to mitigate the risk of it being misused.

This does not mean credential rotation is all bad, just for the end user. For critical accounts like local administrator and service accounts, automatic credential rotation is possible. Both these accounts types are a common target for threat actor. For local admin, these passwords are often the same across an organisation, as computers get setup using the same base image. One thing that can help us is Local Administrator Password Solution (LAPS).[4] The way LAPS works is that it will automatically rotate local administrator passwords and store them in AD, with the length and rotation frequency set centrally using a group policy. The system protects the password stored in AD using access control lists. Where only specific groups can access it, it also audits which users retrieved the password. With LAPS, the local administrator account can stay enabled, without the risk of mass compromise, using techniques like pass-the-hash on the local administrator account. LAPS requires the installation of a Windows LAPS agent on every machine that going to be using it. And any group policies that attempt to set the local administrator password need to be removed.

The other is service accounts that are created for a specific application to function. The problem is that administrators often never change service account passwords for fear of breaking the application. To add to the whole thing, they often also have elevated privileges to perform their tasks. Using AD Managed Service Account if it is supported by the application. Using Managed Service Accounts allows the passwords to be managed by AD instead and automatically rotated, removing the need for administrators to rotate the passwords manually.

Remember, usernames and passwords are a major weakness in organisations. It was believed that passwords would have died off by now, but they are in use on more devices and applications than ever. The best solution to mitigate the risk of password is to enforce MFA on all accounts.

7.4 Single Sign-on

Single sign-on (SSO) is the process of using a single account to sign in into multiple applications. The core idea of SSO is that a user or application attests their identity once to a trusted source. When a user then needs to access another system, the user then directs that system to the trusted source to figure out how to authenticate the request. This allows the user to only remember a single username and password combination, and once logged, they do not have to re-authenticate. You have probably already experienced SSO, whenever a website provides you with the option to sign in with your Facebook or Google account. When you select ‘sign in with’, the application is then sent directly to the trusted source based on the sign-in partner of your choice, which is then responsible for authenticating your request.

Before SSO became widespread, users would have to create a new account for every application they wanted to access. Making it impossible for users to remember all these passwords, resulting in people just reusing the same password for multiple accounts. That resulted in when one website got compromised, the same username and password combination could then be used on other websites to gain access. Another concern that SSO solves is the implementation of service identification and authentication; remember that having a vulnerability in the authentication process is part of the top 10 most common vulnerabilities. Ask yourself this, have their authenticated process been tested against common attacks? With SSO, we trust that the trusted partner has done their due diligence with securing their platform. Having a general overview of SSO is not enough for a security professional. Let’s go a step deeper into the two most common methods of implementing SSO using OAuth and SAML.

7.4.1 OAuth

OAuth is used for general internet users and consists of three elements:

- The user

- The identity provider (authoritative source of user identity, like Google)
- The service provider (the application that the user is trying to access)

When a person wants to sign in to the service provider, instead of using a separate set of credentials, they ask the identity provider instead to verify their identity.

1. User asks the service provider if he can use an SSO to log-in.
2. Service provider tells the user system to get an asymmetric key from the identity provider.
3. User asks Google for a key to let the service provide valid his credentials.
4. Identity provider sends a key back to the user.
5. The user then sends that key over to the service provider.
6. The service provider sends the key to the identity provider, to verify if the user is valid.
7. The identity provider looks up the key in the database and sends back a response that the user is valid.

In OAuth, none of the three parties exchanges passwords. They simply pass asymmetric keys to each other. Look at Figure 7.1 to see a graphic of the flow that is being performed.

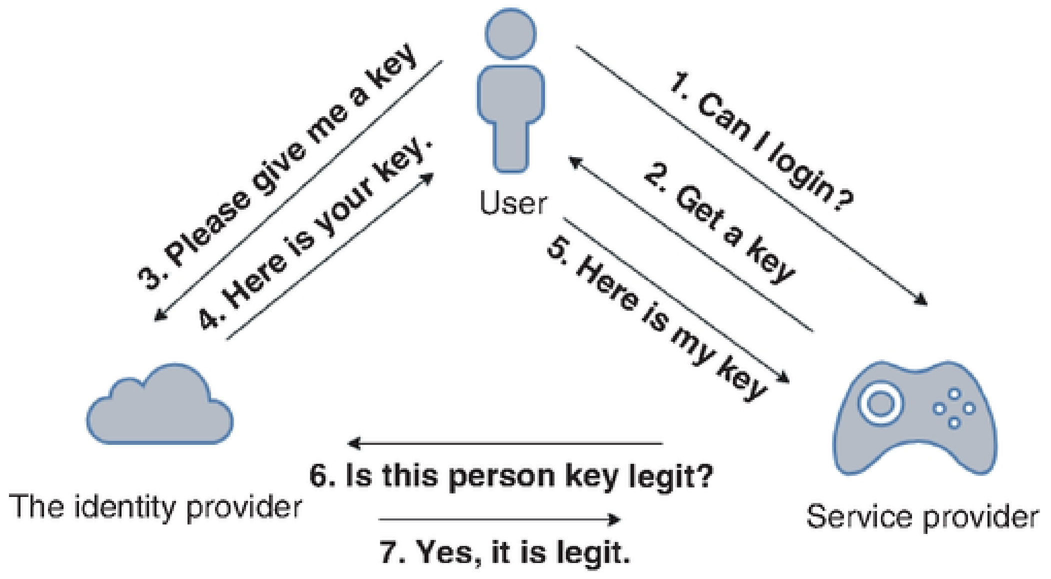


Figure 7.1 OAuth flow.

The general idea of OAuth is that users pick an organisation that they trust to act as an identity provider on their behalf that other service providers can use to verify them against.

7.4.2 SAML

The process for SAML is similar to OAuth, but more robust. Enterprises typically use SAML. Instead of sending an asymmetric key like OAuth, the identity provider sends an encrypted package containing user information such as personal information, groups, roles and other useful sign-in data. This additional information can be used to enforce privileges such as the level of access the user has to access the application.

1. User begins the process to sign in to the service provider.
2. Service provider tells the user system to retrieve a user package from the identity provider.
3. User asks Google for a user package for that specific application.
4. To verify that the service provider is who they say they are, the identity provider asks the service provider for its key.
5. Service provider sent its public key to the identity provider.

6. The identity provider encrypts the user information with the service provider key and sends it to the user.
7. The user then sends the encrypted key over to the service provider.
8. The service provider opens the package with its private key and makes a decision about what the user has access to within the application.

See Figure 7.2 to get a graphic of the flow of the authentication process for SAML. This requires that the organisation has a SAML identity provider to utilise this feature. There are multiple SAML identity providers on the market, such as Gsuite (Google for business), Azure AD and many more.

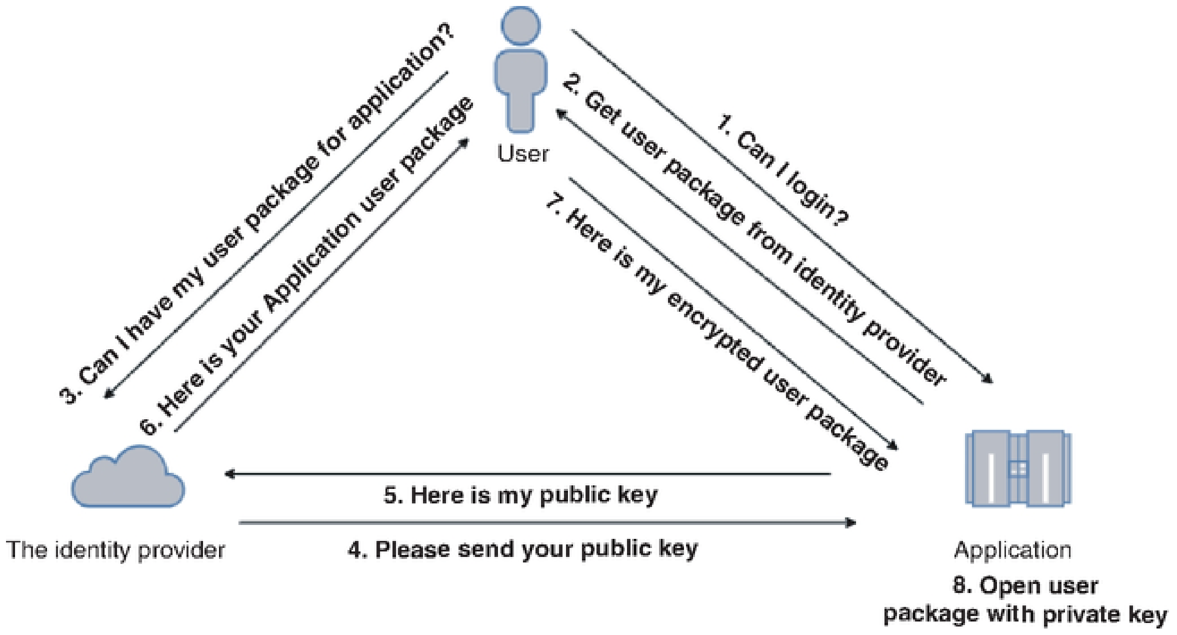

User
1. Can I login?
2. Get user package from identity provider
3. Can I have my user package for application?
6. Here is your Application user package
7. Here is my encrypted user package
5. Here is my public key
4. Please send your public key
The identity provider
Application
8. Open user package with private key

Figure 7.2 Security markup language flow.

Today, users can easily take advantage of SSO for their everyday internet usage thanks to OAuth. And corporate can create a robust IAM policy following Zero Trust using SAML. This will require a little more planning and effort implementing, but it is something we all should be pursuing.

7.5 Applying Zero Trust to IAM

IAM is at the core of any IT enterprise. Identity is what enables people to do their jobs. The problem with identity is that it is tempting to 'trust' the users. With Zero Trust, the goal is to remove this trust wherever possible. The way we do this in IAM is by removing the trust we have given to user and stripping them of all privileges that are not necessary for them to perform their tasks. I do not know how often I have seen more domain admin, any of the other privilege roles combined. This privilege access is often given because they needed to do an administrative task, and it was just easier to give them full admin access, instead of specifying the specific admin privilege needed, because we trusted that the user would not misuse their privilege or the account won't become compromised.

The approach of applying Zero Trust to IAM is the practice of using an Identity Governance group. Due to the fact that IAM is not being an IT process, but a business process, this group should be made up of business owners and people from HR, Legal, Risk, Privacy and IT. This group defines owners of identities, often the supervisor. It is then the identity owner's job to ensure access gets revoked as people leave the organisation, change position and anything else that could initiate a change in user privilege.

For cloud, IAM is everything. With on-premise environments, we could rely on the isolation from the internet for protection. With the cloud, those controls are no longer effective, as everything is available from everywhere. And because cloud has to support a wide range of clients, there are many ways for a user to authenticate themselves. This means that not only do you have to identify every way that one can authenticate themselves, but also you have to be sure that each of them is sufficient protection from attacks. For example, if you are using Azure AD, I would recommend looking at projects like MFASweep[5] by Beau Bullock; the purpose of the tool is to verify that MFA is enabled on all 10 different ways to authenticate yourself to Azure AD.

We can further remove trust from the system. Just because you sometimes need administrative privilege to do your job does not mean you constantly need to have that privilege available. It would be better to only have this privilege at the specific time when it's needed. This method of 'just in time' privilege administrator forces users to further validate their identification. When they want to do high-privilege task, it only grants users access to the privileged role for a set period to achieve a specific goal. This way, we also get an audit trail of who had privilege access at the given time, but also what was the purpose of this privilege access request. There are a lot of elements one has to consider when it comes to IAM. We can take all the ideas and concepts of IAM. We can break it into three parts.

- Identity Governance and Administration (IGA): The group consists of IT, security and business leaders who define the policy of IAM.
- Privileged Identity Management (PIM): The system that is responsible for managing all the identities and their access level. Think of this like a giant database.
- Privileged Access Management (PAM): The system that enforces the rules created by the IGA against the identities in the PIM.

You can buy these services from vendors like in cloud or build and deploy them in-house, or a combination of both. Regardless of choice, a collection of these three parts must be present for any IAM program. The last thing is that the IAM is the key to the organisation. If threat actors take control of the IAM system, they can bypass all security controls. Therefore, protecting the IAM is a top priority of any security professional, and ensuring that any changes made to the system are only done so by authorised personnel.

7.6 Importance of Separation of Duties

> You should not use the same account for browsing the internet and for performing administrative tasks on the domain controller.

Separation of duties (SoD) is a fundamental security principle that ensures no single individual has complete control over any critical function or asset. This is often seen in finance where the person responsible for adding suppliers and clients to the finance system and the person responsible for paying the bills are two separate people, thereby reducing the risk of insider threat. This principle can be duplicated for how we handle IT administrative tasks within an environment.

Administrators should not use the same account and computer that they use for everyday activities like checking email or browsing the web, which has an inherent risk of involved in interacting with the internet. And then uses the same account for performing administrative tasks. Splitting these activities into separate accounts and computer makes it possible for us to reduce the risk of compromising administrative accounts through phishing attacks, malware infections or similar threats. Here are some of the key reasons for separating duties:

1. Minimise Insider Threats: By dividing responsibilities among different individuals, organisations can reduce the risk of malicious actions by insiders.
2. Error Reduction: It reduces the likelihood of errors, as multiple individuals are involved in critical processes, providing checks and balances.
3. Compliance: Many regulatory standards and frameworks, such as PCI DSS and SOX, mandate the SoD to ensure robust security practices.
4. Accountability: Clearly defined roles and responsibilities enhance accountability and traceability of actions within the system.

Remember separating privileges is not enough. Because of the level of access these accounts have. This means that administrators' accounts should be better protected than any other accounts, requiring additional authentication before access is granted.

7.7 Jump Box

> Just like you do not want a single account to be used for both basic and administrative activity, nor should your computer.

Just like how we remove trust from accounts by separating standard privilege from administrative privilege. The same idea should be applied to computers, by separating basic and administrative activities. This way, we can remove the trust that IT administrative computers are secure. The problem with keeping everything on a single computer is that account information and passwords are stored inside the device's memory. Making it possible for an attack with system-level access to the device to dump the memory of the device and find the credentials used for the administrative account stored within the device memory. Instead of having two physical workstations, employees can utilise remote desktop or virtual desktop infrastructure (VDI). This is where a Jump Box comes into play. A Jump Box is a dedicated system, which is a device that creates a bridge across two separate security zones. That is to say that there must be a unique jump box for each bridge between security zones. Another important setup of Jump Box is that when they are not in use, they should be destroyed just like how Docker containers are destroyed when no longer in use. This allows us to destroy all sensitive information that is stored within the system's memory. The problem with having Jump Box persistence is that they will quickly become a target for threat actors, as they know that Jump Box is a source of administrative accounts. You can read more about how to secure privilege access to at Microsoft documentation.[6] Some advantages of using a Jump Box are:

1. Isolated Environment: Provides a controlled and isolated environment for administrative tasks.
2. Access Control: Enforces strict access controls and monitoring of administrative sessions.
3. Reduced Attack Surface: Limits the exposure of critical systems to potential threats from the broader network.

Another key factor is how these different accounts are managed. It would be insecure to manage all accounts in the same AD. Having a single point of authentication would reduce the separation between the networks, resulting in a star network topology, as shown in Figure 7.3.

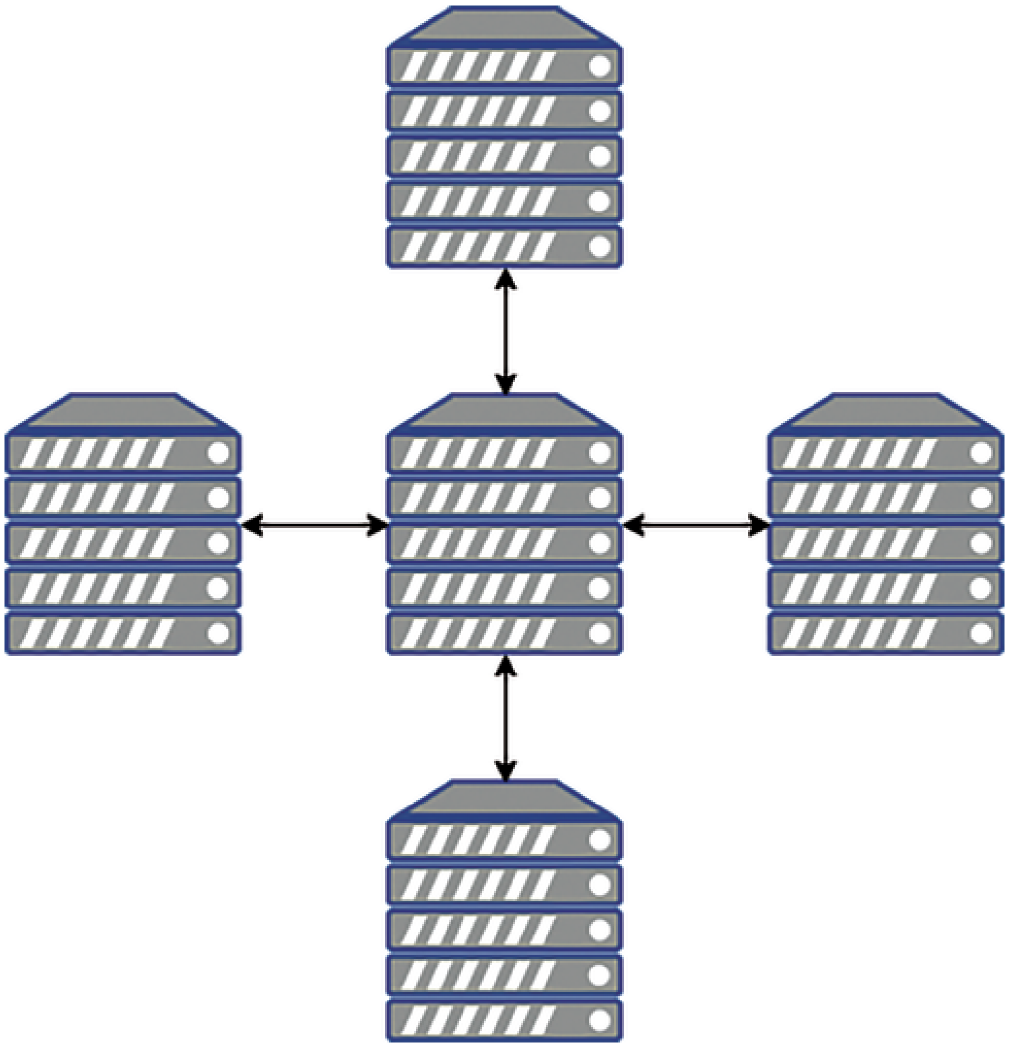


Figure 7.3 **Star topology.**

With the AD being the hub for all the networks, compromise the hub and everything is under your control. That is the reason we need to try to separate the network as much as possible to reduce the possibility of threat actor being able to pivot from one network to another. We should never trust anything from other networks, including users. You might be wondering how many accounts do you need to manage? That, of course, depends on the environment and how much access you need. For Juice Factory, an administrator that needs access to all networks would need at least four accounts.

- Basic User – for function such as email and web browser
- Operational Technology (OT) Administrator – doing administrative work on the OT network

- DMZ Administrator – doing administrative work on the demilitarised zone (DMZ) network
- Corporate Administrator – doing administrative work on the Corporation network

This also requires you to mange three separate active directories. This way, if one of the active directories is compromised, the threat actor does not have access to the entire environment. I know this sounds like a lot of work to build and manage afterwards, especially for older networks with a lot of different systems. Remember that for Juice Factory, each zone we created has a different level of risk to the organisation and level of trust. Threat actors constantly attack the DMZ because of its internet exposure. Compare that to the OT network, which is isolated from the internet.

7.7.1 Simulation of Jump Box in Our Project

In a real organisation, solutions like Citrix or similar VDI tools are often used to implement Jump Boxes. That is not possible for our project. Instead, we will simulate the idea of a Jump Box using a Docker container with web VNC that opens the human–machine interface (HMI) page on the OT environment, allowing us to control the production line from the Jump Box. We also remove the direct access to both the HMI and PLC; this ensures that all interactions with the OT environment must happen through the Jump Box. First, let’s update the transaction map to reflect the changes we are going to make in our environment. See Table 7.1 for the updated transaction map.

TABLE 7.1

Transaction map.

	CRM	IAM	OT	Physical security	Web store	Backup	Wireless	Email	Internet	Network monitoring	Jumpbox
CRM	■				■	■					
IAM	■	■	■	■		■	■	■			
OT			■					■			■
Physical security		■		■							
Web store					■	■			■		
Backup					■	■		■			
Wireless		■					■		■		
Email		■			■	■	■	■	■		
Internet					■		■	■	■		
Network monitoring										■	
Jump Box											■

The transaction map explicitly defines what interaction the OT environment needs to perform its task. Therefore, based on the transaction map, there is no need for the OT environment to have access to the internet and should therefore be isolated from the internet, including both ingress and egress. We'll add a Jump Box to the OT network in the network diagram to show the changes we made; Figure 7.4 shows the Jump Box acting as a bridge between the corporate and OT networks.

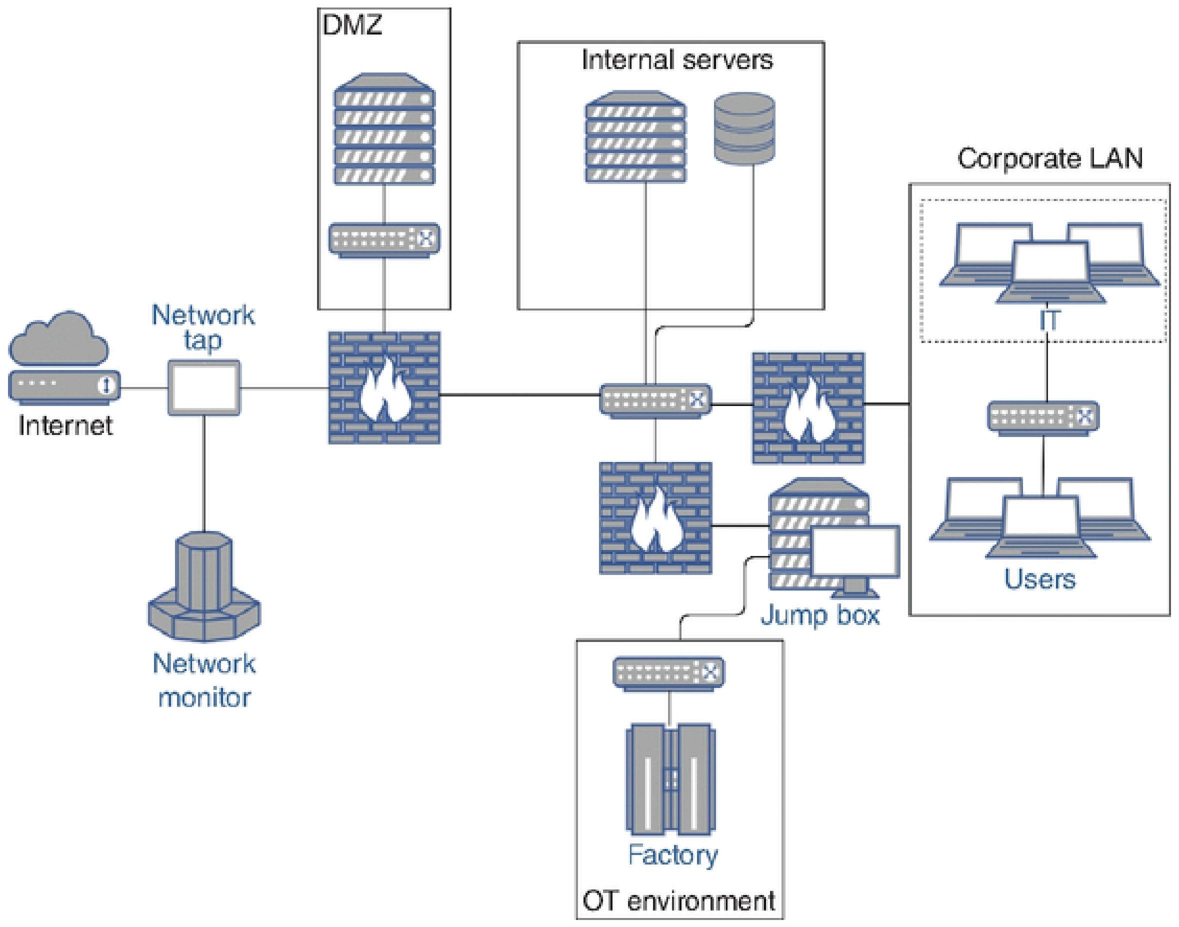


Figure 7.4 Network diagram.

Key points of this setup:

1. Network Isolation: We use Docker's network settings to ensure the OT network is isolated and does not have direct internet access.
2. Access Control: The Engineer Jump Box is configured to access the HMI server within the OT network, providing a secure and controlled access point.
3. Internal Networks: By setting the OT network as internal, we ensure that containers within this network cannot connect to the internet, adhering to the Zero Trust principle of minimising trust zones.

The first step is to isolate the OT network from the internet. This can be achieved by setting the *internal* network flag on the *production* network configuration, isolating the network. This implementation aligns with the Zero Trust principle of 'need to know' where only the thing that is explicitly necessary is permitted. For the Jump Box, we are going to simulate the idea by utilising the ‘linuxserver/chromium’ docker image. This is a containerised version of the Chromium browser that is accessible via Web VNC, which is accessible from our host machine web browser.

We set this Jump Box to have access to both the OT network and the host machine, working as a bridge between them. In real-world implementation, there would be a need to provide additional credentials unique to the OT environment to access the HMI server from the Jump Box. This is outside of the scope of this book, as I have chosen not to simulate any IAM solutions in our environment. The following is the Docker configuration changes that are needed to isolate the OT environment and create the Jump Box we will be using.

```
services:
  bottle-filling_plc:
    image: paradoxxs/virtuaplant:plc
    expose:
      - 5020
      - 3000
    networks:
      - Production
  bottle-filling_hmi:
    image: paradoxxs/virtuaplant:hmi
    environment:
      - PLC_SERVER_IP=bottle-filling_plc
      - PLC_SERVER_PORT=5020
    expose:
      - 3000
    depends_on:
      - bottle-filling_plc
    networks:
      - Production
  Engineer_Jump box:
    image: lscr.io/linuxserver/chromium:latest
    container_name: Engineer_Jump box
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
      - TITLE=Jump box
      - CHROME_CLI=http://bottle-filling_hmi:3000
optional
    ports:
      - 3000:3000  http
      - 3001:3001  https
    shm_size: "1gb"
    restart: unless-stopped
    networks:
      - Production
      - Corpation
    depends_on:
      - bottle-filling_hmi
networks:
  Production:
    name: Production
    internal: true
  Corpation:
    name: Corpation
    internal: false
```

With the Jump Box added to the Docker compose, all you have to do is spin up the environment again and head over to localhost:3000, and you should see the HMI interface to the OT environment. Benefits of Network Isolation:

1. Protection Against External Threats: By preventing the OT network from accessing the internet, we reduce the risk of external attacks.
2. Control of Internal Communications: We have granular control over which services can communicate within the network.
3. Simplified Security Management: Limiting internet access simplifies the security management of the OT environment, focusing on essential interactions only.

7.8 Summary

In this chapter, we implement a Jump Box, which maintains separation between the OT and the corporate network. This ensures that we only perform administrative tasks on the OT environment in a controlled environment. Another is the isolated of the OT networks from the internet, further limiting the risk to the network. Although our simulated setup demonstrates these principles and provides an example of how to secure yourself using a Jump Box and separating duties from each other. Remember what we learned about the principles of least privilege, privileged access should be limited to the moment in time when it is needed using just-in-time and the idea of separating administrative task from ordinary activities. With these concepts implemented in your environment, you will be far more secure than most organisation out there today.

Notes

1 https://fidoalliance.org/

2 https://pages.nist.gov/800-63-3/sp800-63b.html

3 https://haveibeenpwned.com/

4 https://learn.microsoft.com/en-us/windows-server/identity/laps/laps-overview

5 https://github.com/dafthack/MFASweep

6 https://learn.microsoft.com/en-us/security/privileged-access-workstations/privileged-access-devices

Chapter 8
Endpoint Detection and Response

> As you check your email, an alert from the network monitoring team catches your attention: malicious traffic detected from a satellite office workstation. This situation highlights a critical gap in your Zero Trust implementation – while you've established network segmentation and monitoring, you lack the ability to verify and validate endpoint security status in real-time. The choice between sending an incident responder or shipping the computer back to headquarters represents an unacceptable delay. It's time to implement comprehensive endpoint monitoring.

First, let's look back at what we have achieved so far. We segmented the network to help mitigate the possibility of lateral movement. Gained visibility into the network using a network monitoring system. Looked at identifying and separating the engineering 'normal' activity from their administrative activities on the operational technology (OT) network, using Jump Box. The next project we will look at is getting visibility into all the endpoints in our organisation. This will help us determine if they have been compromised. The goal is to know if we can trust the system. One solution to this problem is Endpoint Detection and Response (EDR). This is a software agent that sits on the computer and constantly monitors the system for any threats and responds in real time, by either blocking or alerting to malicious activities. With all the agents controlled from a centralised location, making it easier for security analysts to monitor and respond to any security incidents that might occur on thousands of endpoints with relative ease.

In this chapter, we will explore the significance of EDR and its new counterpart, Extended Detection and Response (XDR) and its essential features, including its role in Zero Trust Architecture (ZTA). We will also introduce Velociraptor, which is an open-source EDR tool, and demonstrate how it can be deployed within our simulated environment.

8.1 Core EDR Components

EDR solutions comprise four fundamental components that work together to provide comprehensive endpoint security. Understanding these components is crucial for implementing EDR effectively within a Zero Trust framework.

8.1.1 Detection Engine

The detection engine serves as the primary sensor, continuously monitoring endpoint activity for potential security threats. Unlike traditional antivirus (AV) solutions that rely primarily on signature-based detection, modern EDR detection agents employ multiple detection methodologies. These include behavioural analysis, machine learning algorithms and real-time pattern matching. The agent monitors system activities, such as process creation, file system changes, network connections and memory modifications. This comprehensive monitoring enables the detection of both known threats and potential zero-day exploits that might evade traditional security measures.

8.1.2 Data Collection and Analysis

The data collection component maintains a detailed record of endpoint activities, creating a comprehensive audit trail that proves invaluable during incident investigation. This component collects and processes various types of telemetry data, including process execution chains, network connections, file system modifications and user activities. The analysis engine correlates this data across multiple dimensions, identifying patterns and potential security implications that might not be apparent when examining individual events in isolation. This capability is crucial in a Zero Trust environment, where continuous validation of a security posture is essential.

8.1.3 Management Console

The management console provides centralised control and visibility across all protected endpoints. This interface allows security teams to configure policies, monitor alerts, investigate incidents and coordinate response actions. In a Zero Trust context, the management console plays a crucial role in maintaining security policy consistency and providing evidence of security control effectiveness. The console must be secured appropriately, as it represents a high-value target for attackers because of its privileged position within the security infrastructure.

All these components work together to create a comprehensive endpoint security solution that supports Zero Trust principles by providing continuous monitoring, rapid response capabilities and detailed audit trails.

8.2 Comparison to Traditional AV

Traditional AV solutions focus on detecting and blocking known malware using signature-based detection. AV relies on a database of known malware signatures and heuristic analysis to identify suspicious patterns. Once malware is detected, AV will typically try to quarantine and remove the threat with minimal user intervention, offering protection against common threats like viruses, worms, Trojans and spyware. However, AV's scope is generally limited to individual endpoints and known threats.

EDR provides a more holistic approach to endpoint security, monitoring a broader range of sources beyond file signatures, including network activities, processes and commands. This enables EDR to detect, investigate and respond to both known and unknown threats. EDR combines signature-based detection with behavioural analysis, machine learning and threat intelligence, providing advanced capabilities like automated responses, threat hunting, forensic analysis and remote remediation.

Unlike AV's reactive approach, EDR continuously monitors endpoint activities in real time, identifying suspicious behaviour that might bypass traditional detection methods. It provides detailed forensic data, which helps security teams respond effectively to incidents by offering insights into the threat. EDR systems can automatically isolate infected endpoints, block malicious processes and initiate remediation steps, ensuring a swift and efficient response to incidents.

In a Zero Trust security model, where every access request requires verification and no entity enjoys inherent trust, EDR plays a critical role. EDR is essential for any organisations, no matter its size or complexity. EDR's ability to provide advanced detection, detailed forensic analysis and automated response is crucial in high-risk environments and for organisations adhering to strict security regulations.

Key Advantages of EDR:

1. **Visibility into Encrypted Traffic**: As more web traffic becomes encrypted, traditional monitoring tools struggle to inspect it. EDR solutions, installed directly on endpoints, can monitor activities even in encrypted environments, identifying potential threats.
2. **Comprehensive Threat Detection**: EDR collects detailed information about endpoint activities, such as file changes, process executions and network connections, enabling detection of both known and emerging threats.
3. **Rapid Incident Response**: EDR solutions enable quick responses to incidents by isolating infected devices, terminating malicious processes and conducting remote investigations. Some EDR tools provide remote shell access for executing commands on compromised endpoints.
4. **Enhanced Forensics**: EDR provides tools for detailed forensic analysis, including capturing memory dumps and historical logs. These features help security teams investigate incidents and understand the full scope of breaches.

8.2.1 Extended Detection and Response

XDR builds on the principles of EDR. XDR extends the functionality by integrating additional data sources into the fold, by including its protective capabilities to devices like network devices, Internet of Things (IoT) and cloud services. XDR can provide a holistic view of the security landscape by correlating security alerts from multiple sources into a single place, enabling more comprehensive threat detection and response to incidents.

Advantages of XDR:

1. Broader Visibility: XDR aggregates data from multiple sources, providing a more complete picture of the security environment.
2. Improved Correlation: By correlating data from different security layers, XDR can identify sophisticated threats that might go unnoticed when data is siloed.
3. Streamlined Operations: XDR consolidates alerts and workflows, reducing the burden on security teams and enabling faster, more efficient incident response.

Of course, these additional functionalities of XDR also come with additional financial costs.

8.3 Adding EDR to Our ZTA

From a Zero Trust perspective, EDR poses a slight problem. The thing with security tools like EDR is that they should be deployed on as many devices as possible while being controlled from a centralised location. This poses a big challenge, where the tool management interface ever becomes compromised, an adversary could take control of the entire network without a single alert ever being created. The key to resolving this conflict lies in applying Zero Trust principles to the EDR system itself. This involves implementing strict access controls, continuous monitoring and granular permissions. For example, limit EDR communications to specific ports and protocols, monitoring all traffic for deviations from expected patterns. Next, we must protect access to the centralised control panel with strong authentication, as discussed in Chapter 7. Splitting up the security team privileges roles from the ordinary ones, with the privileges account having additional protection, preferably with just-in-time access for administrative functions. This is thinking in terms of zero trust. Let’s visualise these relationships between the different systems, using the transaction map as shown in Table 8.1.

TABLE 8.1

Transaction map.

	CRM	IAM	OT	Physical security	Web store	Backup	Wireless	Email	Internet	Network monitoring	Jumpbox	EDR
Customer relation management (CRM)	■				■	■						■
Identity access management (IAM)	■	■	■	■		■	■	■				■
OT			■					■			■	■
Physical security		■		■								
Web store					■	■			■			■
Backup					■	■		■				
Wireless		■					■		■			
Email		■			■	■	■	■	■			
Internet					■		■	■	■			
Network monitoring										■		
Jump Box			■								■	■
EDR	■		■		■						■	■

Look at where there are transactions between protected surfaces, and then identify what controls need to be put in place to reduce trust in the system. The EDR system requires access to multiple network segments to monitor endpoints effectively, but each connection represents a potential security risk. To mitigate these risks, we implement several controls:

- Network Access Controls: Limit EDR communication to specific ports and protocols, monitored by network security tools
- Authentication Requirements: Implement strong authentication for EDR management access, including multi-factor authentication

- Administrative Separation: Split administrative roles between EDR management and other system administration duties
- Continuous Monitoring: Monitor EDR system behaviour for anomalies that might indicate compromise

By implementing these controls, we maintain the EDR system's effectiveness while adhering to Zero Trust principles. The goal is not to eliminate all trust 'which would be impossible' but to minimise implicit trust and verify all critical actions. This approach allows us to leverage EDR's powerful security capabilities while maintaining a strong security posture aligned with ZTA.

8.4 Velociraptor: An Open-source EDR

We require, for this project, an open-source system that runs in a Docker container and whose agents are deployable to Windows, Mac and Linux. Lucky for us, there is such a system. This is where Velociraptor comes into the picture. Velociraptor is an advanced open-source EDR tool that provides comprehensive endpoint monitoring and response capabilities, including supporting deployment clients on Windows, Mac and Linux platforms. It offers a range of features that are typically found in commercial EDR solutions, making it a versatile choice for any cybersecurity program without the costs that come from proprietary software. That is everything we need for a modern EDR solution. With Velociraptor being an open-source project, it benefits from community contributions and regular updates, ensuring it remains at the forefront of what can be expected from EDR tools. The collaborative nature of its development fosters continuous improvement and innovation, enhancing its capabilities and ensuring transparency and trustworthiness. Another significant feature of Velociraptor is its ability to perform remote live forensic investigations. This allows security teams to conduct file analysis, process inspection and memory captures in real-time without having physical access to the device. Enabling rapid identification and analysis of suspicious behaviours and potential threats, crucial for responding to incidents as they unfold and reducing the time it takes to understand and mitigate potential security breaches. I think Velociraptor should serve as a benchmark for any commercial EDR product. If they do not have the same feature set as Velociraptor, then you should look elsewhere. Here are some of the key features of Velociraptor:

1. Cross-platform Support: Velociraptor agents can be deployed on various operating systems, ensuring consistent endpoint visibility across the organisation.
2. Live Forensics and Incident Response: Velociraptor allows security teams to perform live forensic investigations, including file analysis, process inspection and memory captures.
3. Scalability: Designed to handle large-scale deployments, Velociraptor can manage thousands of endpoints efficiently.
4. Community-driven: As an open-source project, Velociraptor benefits from community contributions and regular updates, ensuring it remains a cutting-edge tool for endpoint security.

Since the book is not specifically about Velociraptor, if you would like to learn more about the features of Velociraptor, you can check out their documentation[1] over at their website.

8.5 Deployment of Velociraptor in Our Environment

Deploying Velociraptor in our Docker environment is quite easy. For this project, we will use the image called ‘wlambert/velociraptor’, which is a Velociraptor docker image that is deployment ready. We will deploy Velociraptor for our project setup, configuring it to monitor and respond to security incidents on an endpoint container created specifically for this purpose. I will go into details about how to install the Velociraptor on this client and make sure that it communicates with Velociraptor. Deploying the Velociraptor client to all containers that we have in our environment is out of the scope of this book. When you can deploy the client on one server, then you can do it in multiple containers. It is just repeating the same process for every container. Let’s look at the changes we are making to the Docker configuration file:

```
velociraptor:
  container_name: velociraptor
  image: wlambert/velociraptor
  volumes:
    - ./velociraptor:/velociraptor/:rw
  environment:
    - VELOX_USER=${VELOX_USER}
    - VELOX_PASSWORD=${VELOX_PASSWORD}
    - VELOX_ROLE=${VELOX_ROLE}
    - VELOX_SERVER_URL=${VELOX_SERVER_URL}
    -
VELOX_FRONTEND_HOSTNAME=${VELOX_FRONTEND_HOSTNAME}
  ports:
    - "8000:8000"
    - "8001:8001"
    - "8889:8889"
  restart: unless-stopped
  networks:
    - corporation
    - dmz
    - Production
velo_client:
  container_name: velo_client
  image: ubuntu
  tty: true
  volumes:
    - ./velociraptor:/velociraptor/:ro
  networks:
    - corporation
```

We configured Velociraptor in this setup to access all network segments, thus allowing it to monitor and respond to incidents across the entire environment. While this provides comprehensive coverage, it also highlights a potential weakness: if Velociraptor's security gets compromised, an attacker gains access to every connected device. Thus, securing the EDR solution itself is paramount to the overall security of the entire organisation. Doing this for Velociraptor is not part of this book.

8.6 Working with Velociraptor

With the changes made to our environment and spun up, the next step is to head over to Velociraptor web interface at https://localhost:8889. You'll see a window login box, similar to the one shown in Figure 8.1. The username and password are 'admin:admin'. If that doesn't work, check out the .env file for the field VELOX_PASSWORD.

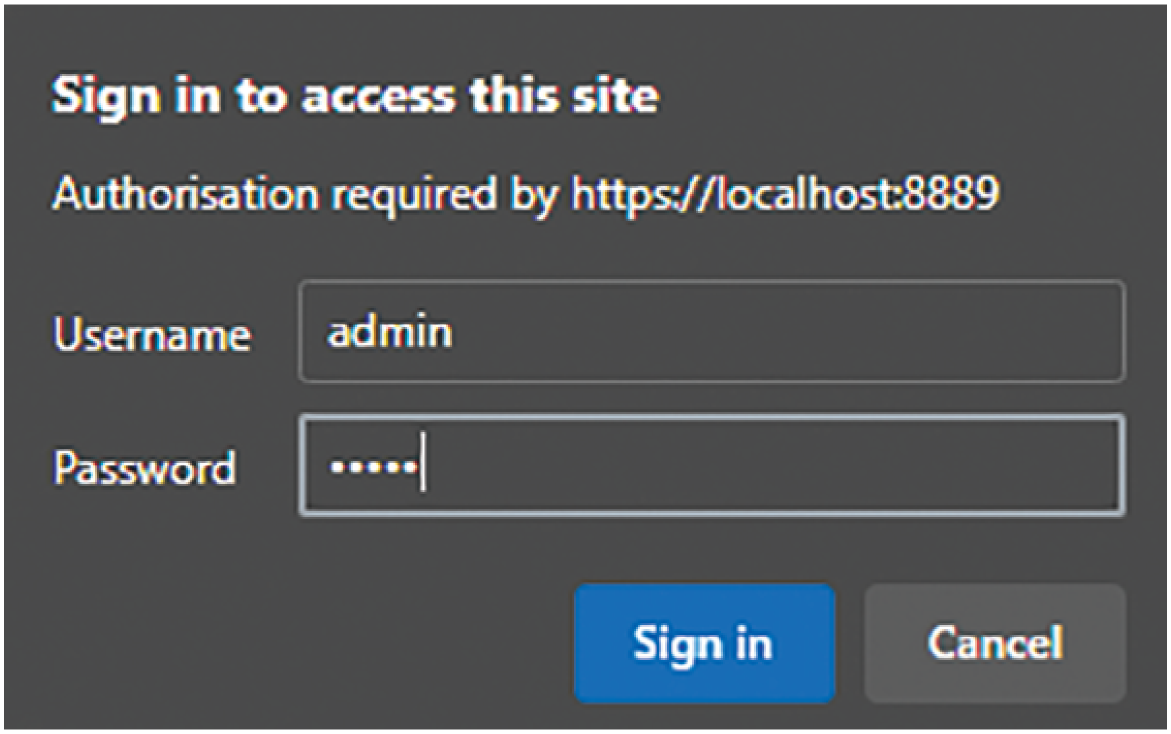

Sign in to access this site
Authorisation required by https://localhost:8889
Username
admin
Password
•••••
Sign in
Cancel

Figure 8.1 **Velociraptor login.**

With the correct username and password entered, you should now be logged into Velociraptor. The client container should already communicate with Velociraptor. Head over to the home page by clicking on the house icon in the top left corner. It should display the current server status of Velociraptor. The 'Current connected Client' graph should already show that one client is connected (Figure 8.2).

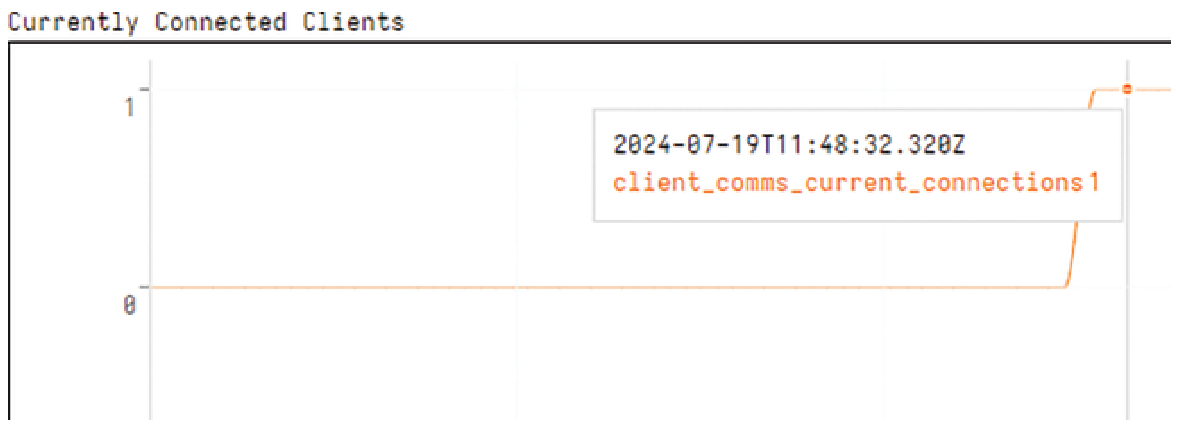


Figure 8.2 **Velociraptor client graph.**

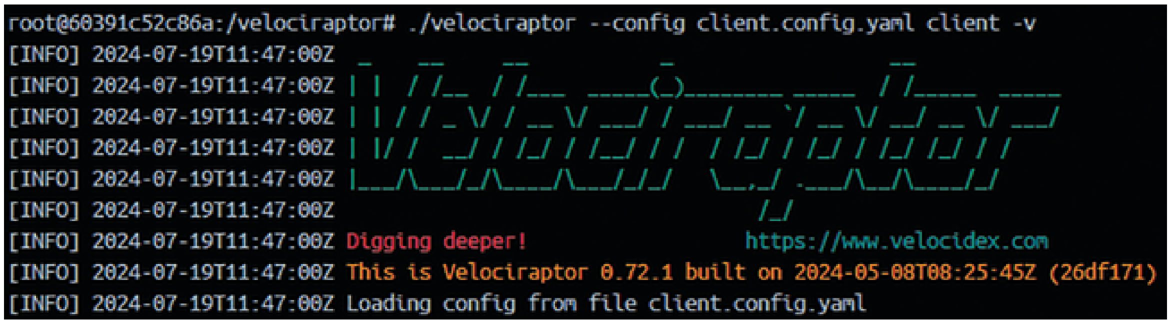
I would also like you to try your hand at adding a client of your own, just to get an idea of how it is done. You can read more about the documentation on Velociraptor webpage.[2] We will set up the Velociraptor server to act as a client as well. First, we have to get shell access to the Velociraptor server. This is quite simple: just open a terminal and run the following command:

```
docker exec -it velociraptor /bin/bash
```

You should now have shell access to the Velociraptor container and be in the '/velociraptor' directory. You can verify this by typing 'pwd' in the terminal. To add the server as a client to Velociraptor, run the following command:

```
./velociraptor --config client.config.yaml client -v
```

If the command succeeds, you should see output similar to Figure 8.3. When it comes to installing the Velociraptor agent on other devices. What you have to do is copy both the Velociraptor binary file and the configuration file to the system, which you would like to install the agent on, and then execute the binary with the same parameters as we did previously.



```
root@60391c52c86a:/velociraptor# ./velociraptor --config client.config.yaml client -v
[INFO] 2024-07-19T11:47:00Z  _    __     __           _                 __
[INFO] 2024-07-19T11:47:00Z | |  / /__  / /___  _____(_)________ _____  / /_____  _____
[INFO] 2024-07-19T11:47:00Z | | / / _ \/ / __ \/ ___/ / ___/ __ `/ __ \/ __/ __ \/ ___/
[INFO] 2024-07-19T11:47:00Z | |/ /  __/ / /_/ / /__/ / /  / /_/ / /_/ / /_/ /_/ / /
[INFO] 2024-07-19T11:47:00Z |___/\___/_/\____/\___/_/_/   \__,_/ .___/\__/\____/_/
[INFO] 2024-07-19T11:47:00Z                                   /_/
[INFO] 2024-07-19T11:47:00Z Digging deeper!                  https://www.velocidex.com
[INFO] 2024-07-19T11:47:00Z This is Velociraptor 0.72.1 built on 2024-05-08T08:25:45Z (26df171)
[INFO] 2024-07-19T11:47:00Z Loading config from file client.config.yaml
```

Figure 8.3 Velociraptor install client.

Let's try to inspect one of the clients that has been added to Velociraptor. Head back to the Velociraptor webpage, then click on the 'Search Clients' search bar as shown in Figure 8.4, and press enter.

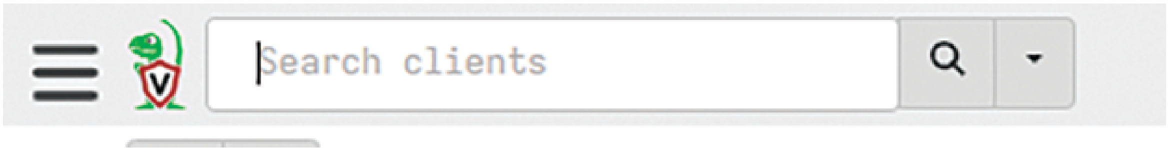
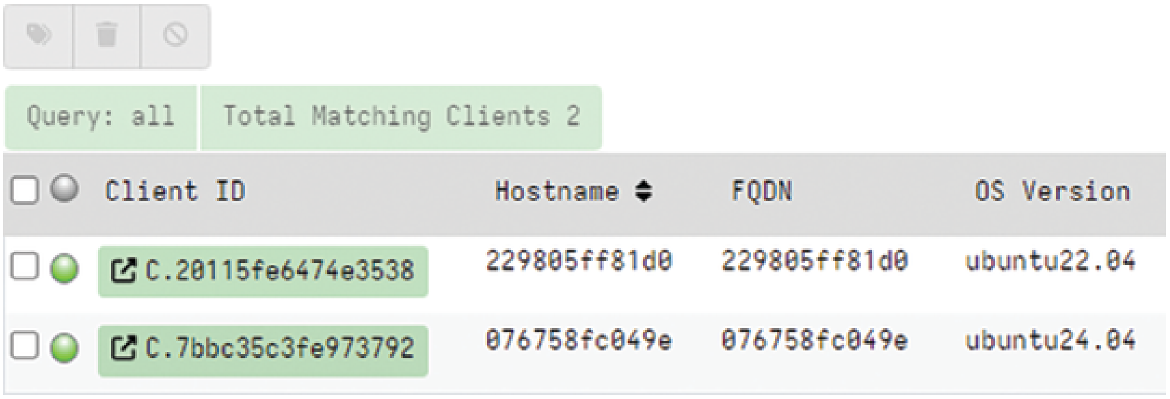


Figure 8.4 Velociraptor search bar.

It should now display all the clients that are connected to Velociraptor. The two clients that have been added to the Velociraptor should both have a grey status, as you can see in Figure 8.5.

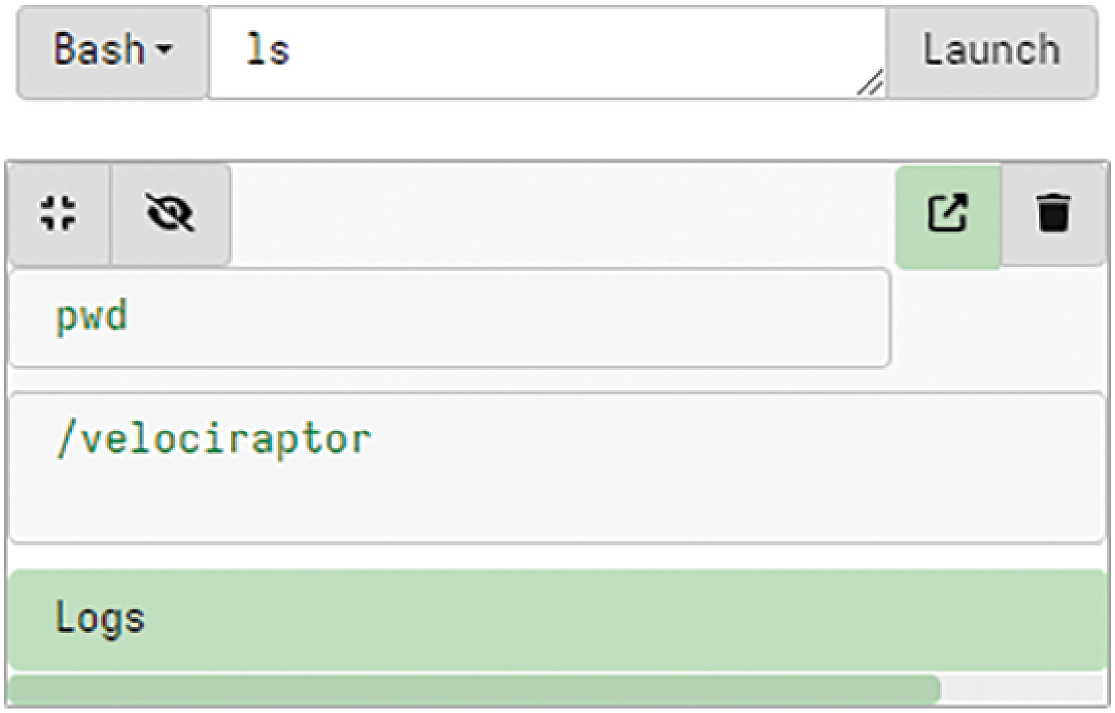


Query: all   Total Matching Clients 2

	Client ID	Hostname	FQDN	OS Version
●	C.20115fe6474e3538	229805ff81d0	229805ff81d0	ubuntu22.04
●	C.7bbc35c3fe973792	076758fc049e	076758fc049e	ubuntu24.04

Figure 8.5 Velociraptor clients.

Click on one of the 'Client id' on one of the clients, and it should take you to the client page. This page will show you additional information about that specific client. From here, you run commands directly on the client. This is done by clicking on the 'shell' button. And from here you can run any command you want to execute on the client by simply pressing launch, as shown in Figure 8.6. To view the output of the command, click on the eye icon. This will display the output of the command.



```
ls
```

```
pwd
```

```
/velociraptor
```

Figure 8.6 **Client shell.**

Let us also do a hunt. The first step is to press the hunt icon shown in Figure 8.7.

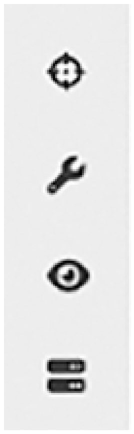


Figure 8.7 **Velociraptor menu.**

Then, press the '+' icon to create a new hunt, as shown in Figure 8.8.

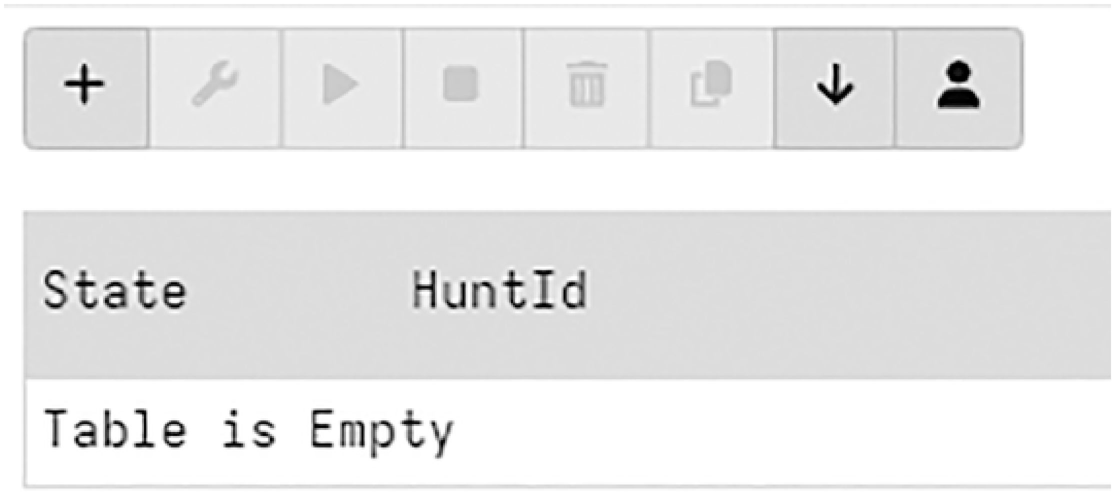
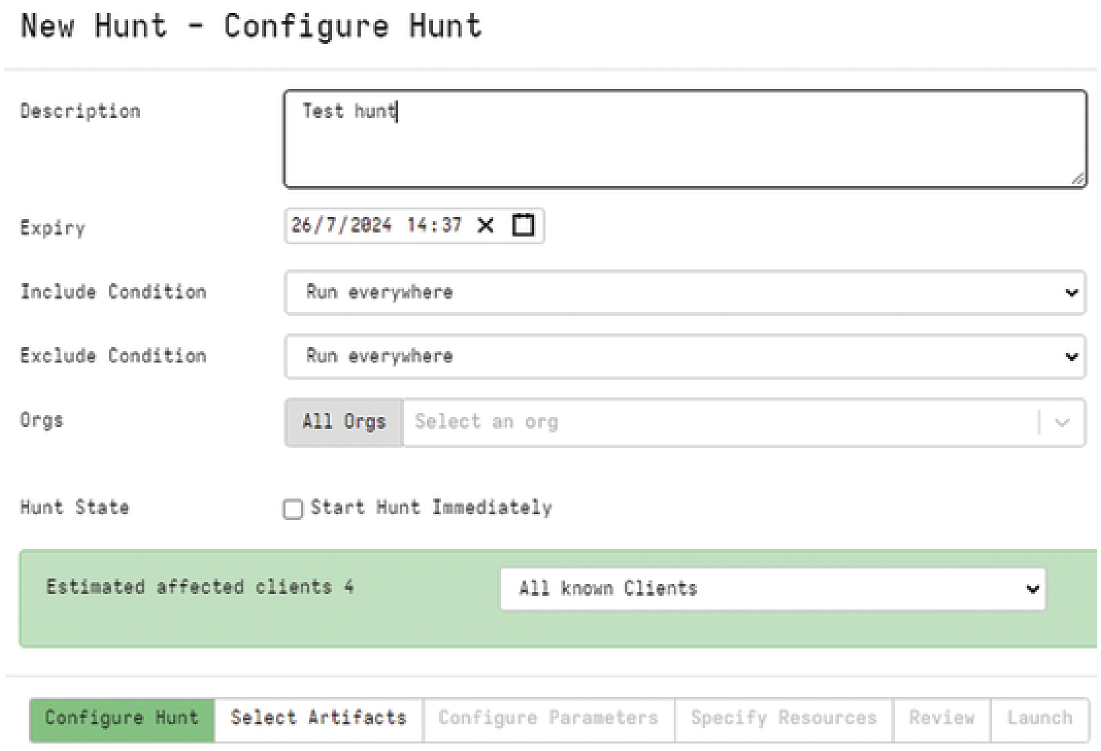

State HuntId

Table is Empty

Figure 8.8 **Add new hunt.**

Please give your hunt a name. This can be whatever you want. You can see in Figure 8.9 that I gave it the name ‘Test hunt’. When you are doing a real hunt, the name should be something more descriptive. Then press the ‘Select Artifacts’ at the bottom.

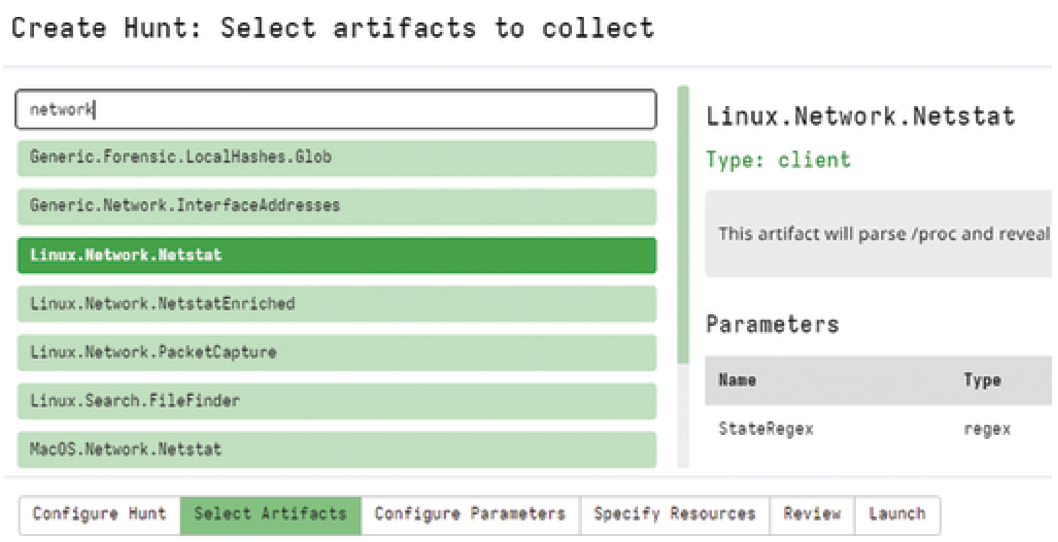

New Hunt - Configure Hunt

Description	Test hunt
Expiry	26/7/2024 14:37
Include Condition	Run everywhere
Exclude Condition	Run everywhere
Orgs	All Orgs Select an org
Hunt State	Start Hunt Immediately

Estimated affected clients 4 All known Clients

Figure 8.9 **Configuration of the hunt.**

We are going to keep it simple and select ‘Linux.Network.Netstat’, like I did in Figure 8.10.


Create Hunt: Select artifacts to collect

network

- Generic.Forensic.LocalHashes.Glob
- Generic.Network.InterfaceAddresses
- Linux.Network.Netstat
- Linux.Network.NetstatEnriched
- Linux.Network.PacketCapture
- Linux.Search.FileFinder
- MacOS.Network.Netstat

Linux.Network.Netstat

Type: client

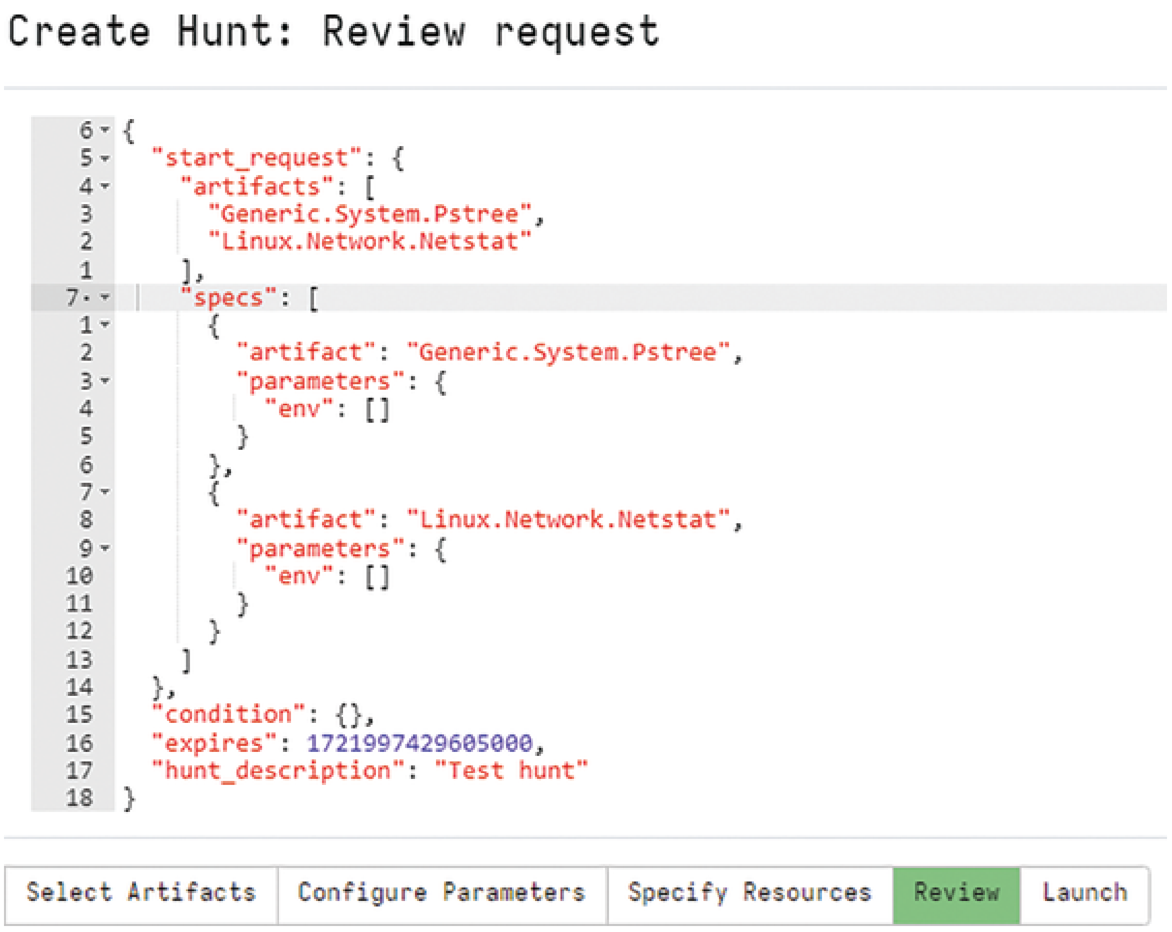
This artifact will parse /proc and reveal

Parameters

Name	Type
StateRegex	regex

Figure 8.10 Hunt artefact.

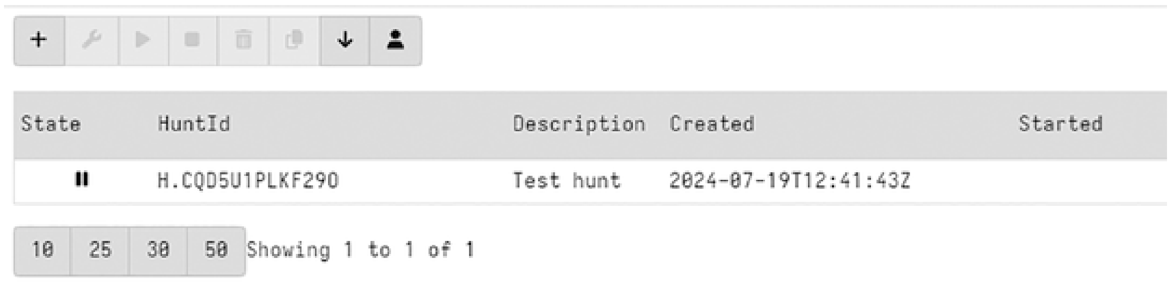
Now press the Review at the bottom of the page, You should see a similar page as in Figure 8.11. This will display what is going to run on the clients.


Create Hunt: Review request

```
{
  "start_request": {
    "artifacts": [
      "Generic.System.Pstree",
      "Linux.Network.Netstat"
    ],
    "specs": [
      {
        "artifact": "Generic.System.Pstree",
        "parameters": {
          "env": []
        }
      },
      {
        "artifact": "Linux.Network.Netstat",
        "parameters": {
          "env": []
        }
      }
    ]
  },
  "condition": {},
  "expires": 1721997429605000,
  "hunt_description": "Test hunt"
}
```

Figure 8.11 Hunt review.

Now that we have an idea of what is going to be run on the clients. Now select Launch. It will start the hunt and be loaded into the queue. The output should look something like in Figure 8.12.

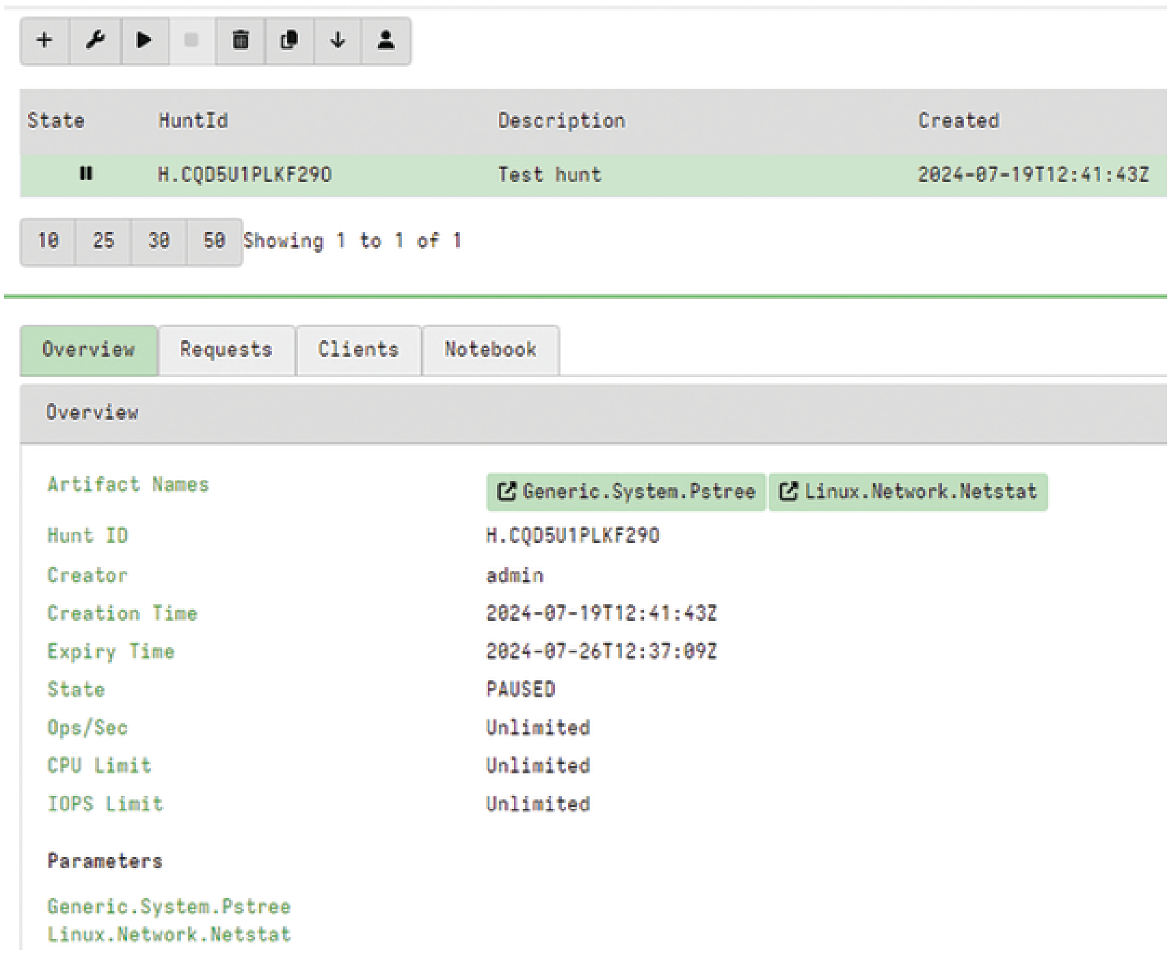


State	HuntId	Description	Created	Started
⏸	H.CQD5U1PLKF290	Test hunt	2024-07-19T12:41:43Z	

10 25 30 50 Showing 1 to 1 of 1

Figure 8.12 **Hunt queue.**

To start the hunt, select the hunt you created, and press the play button above. The hunt should change colour to grey, displaying that the hunt is currently active. Similar to what is shown in Figure 8.13.



State	HuntId	Description	Created
⏸	H.CQD5U1PLKF290	Test hunt	2024-07-19T12:41:43Z

10 25 30 50 Showing 1 to 1 of 1

Overview Requests Clients Notebook

Overview

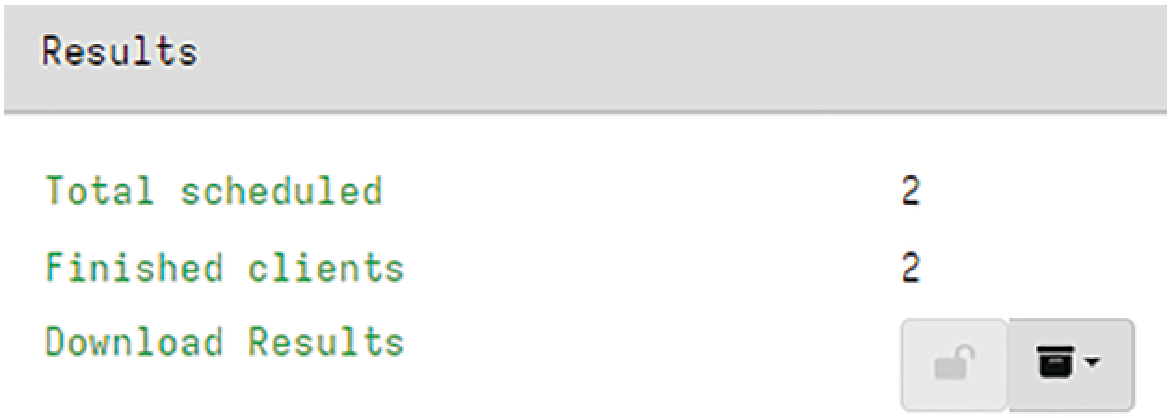
Artifact Names	Generic.System.Pstree Linux.Network.Netstat
Hunt ID	H.CQD5U1PLKF290
Creator	admin
Creation Time	2024-07-19T12:41:43Z
Expiry Time	2024-07-26T12:37:09Z
State	PAUSED
Ops/Sec	Unlimited
CPU Limit	Unlimited
IOPS Limit	Unlimited

Parameters

Generic.System.Pstree
Linux.Network.Netstat

Figure 8.13 **Starting the hunt.**

You should get a pop-up window; just press 'Run it!' It will take a moment for it to complete. When it is done, you should see that 'Finished Clients' is two, as displayed in Figure 8.14.



Results

Total scheduled	2
Finished clients	2
Download Results	

Figure 8.14 **Hunt finish.**

To download the results of our hunt, press the 'Download Results' button. The code snippet below is a cutout of the results I got from the hunt I performed.

```
{"inode":"602960","State":"Established","uid":"0","Proc
essInfo":{"

Pid":1,"Command":"velociraptor_c1\n","CommandLine":"/

velociraptor/clients/linux/velociraptor_client_repacked
\u0000--

config=/velociraptor/client.config.yaml\u0000client\u00
00-v\
    u0000","Filename":"socket:
[602960]","Type":"socket","Inode":"602960"},"LocalAddr"
:{"IP
    ":"172.20.0.3", "Port":44776},"RemoteAddr":
{"IP":"172.20.0.2",
    "Port": 8000}, "FlowId":"F.CQD5U1PLKF290.H",
"ClientId":"C.7
    bbc35c3fe973792", "Fqdn":"076758fc049e"}
```

That is how to do a simple hunt. We have not even begun to touch what we can do with Velociraptor. Remember that the documentation[3] on their site has a lot of information on how to use it. We will come back to Velociraptor later in the book on how to integrate it with an Security Information and Event Management solution.

8.7 Application Allow Listing

Everything we have talking about in the chapter on EDR is all about monitoring. Why stop at just monitoring the activities happening in our system? Just like we can especially define what that connection is allowed to be made on the network. The same can be done with the application that is being run on the endpoints. This is done by especially defining what executable is allowed to be executed on the system. Best of all, this feature is already built into Windows systems by default; it just has to be configured and enabled.

To achieve this, we first have to configure AppLocker. For this, we need to open the Local Security Policy on your Windows System. Press the Windows key and then type ‘Local Security’. It should bring up a menu like the one in Figure 8.15; here you select Local Security Policy.

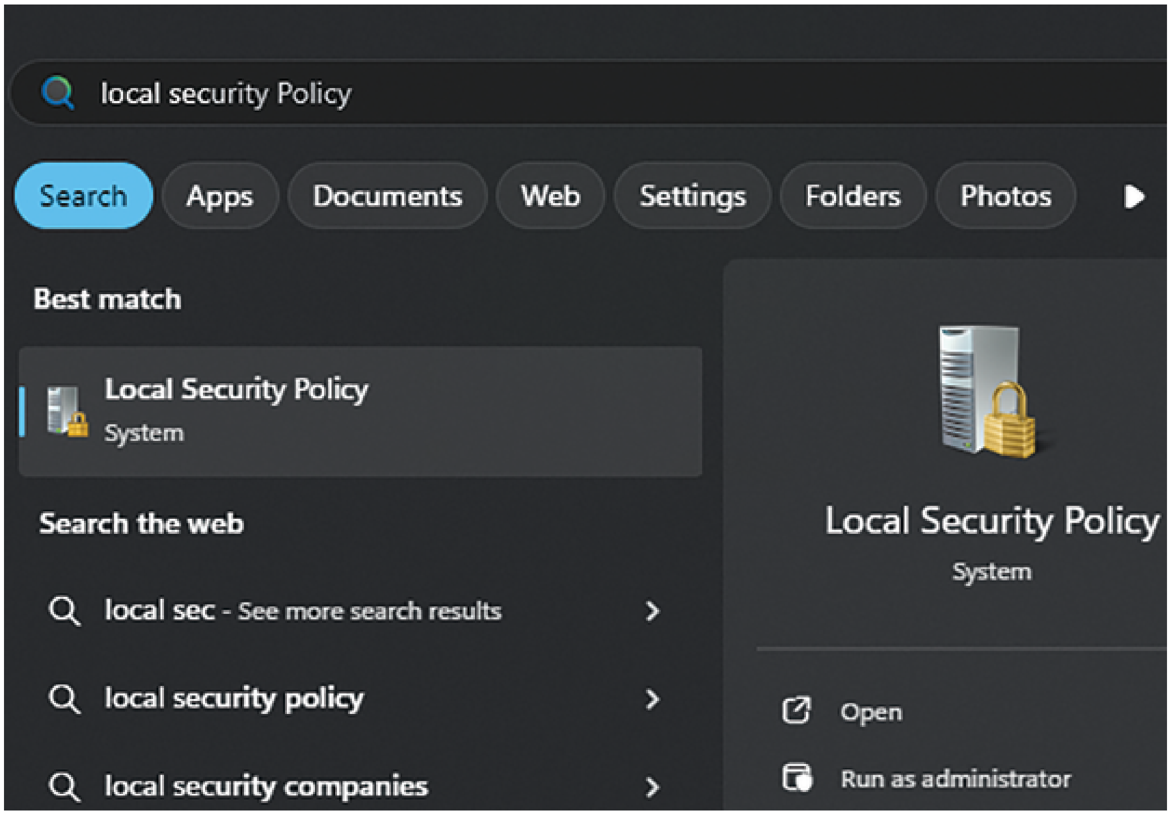
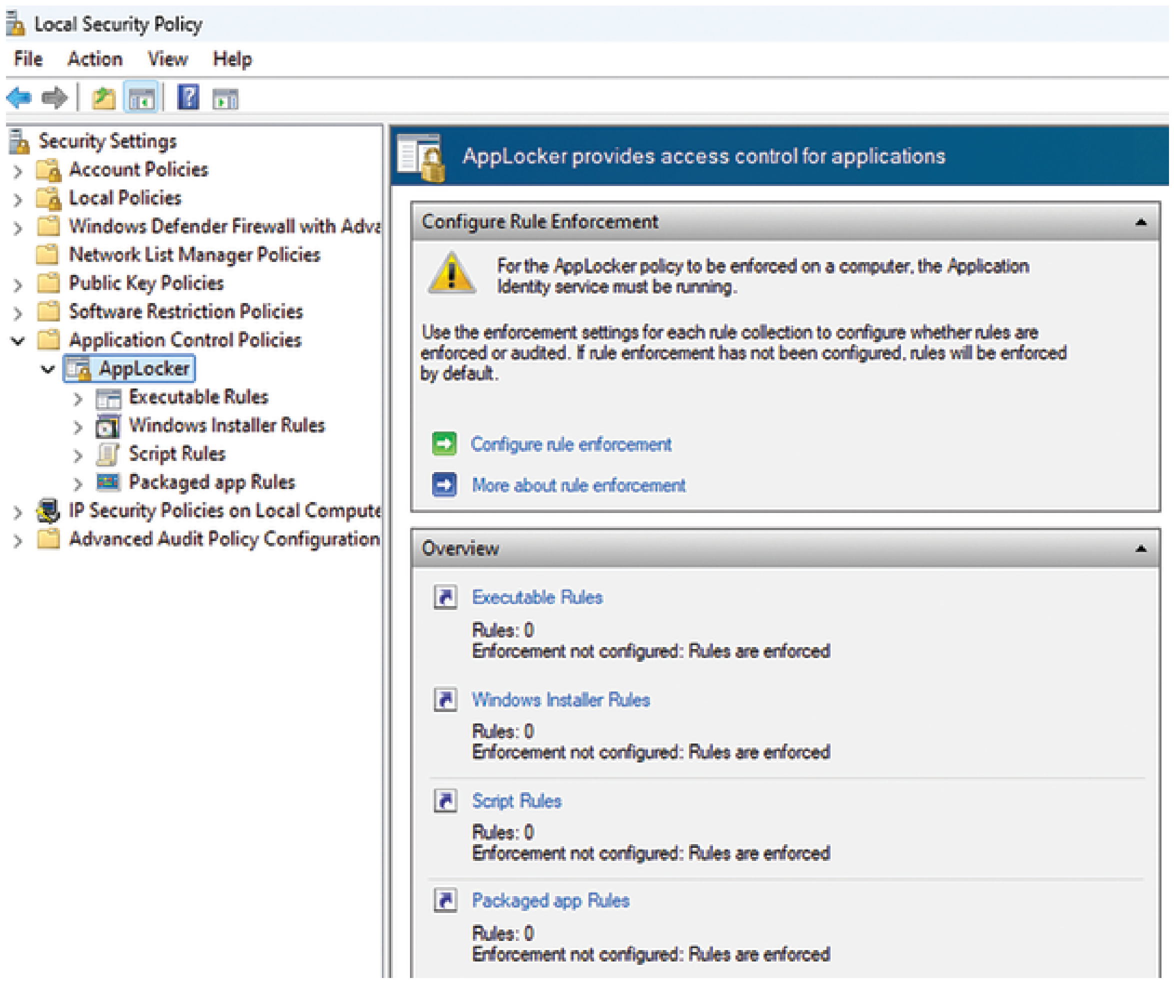


Figure 8.15 **Local security.**

Next, we will need to configure AppLocker. Go to `Security Settings > Application Control Policies > AppLocker`, as shown in Figure 8.16. In the right-hand pane, you will see there are 0 rules enforced for all policies.


Local Security Policy
File Action View Help
Security Settings
Account Policies
Local Policies
Windows Defender Firewall with Adva
Network List Manager Policies
Public Key Policies
Software Restriction Policies
Application Control Policies
AppLocker
Executable Rules
Windows Installer Rules
Script Rules
Packaged app Rules
IP Security Policies on Local Compute
Advanced Audit Policy Configuration

AppLocker provides access control for applications

Configure Rule Enforcement

For the AppLocker policy to be enforced on a computer, the Application Identity service must be running.

Use the enforcement settings for each rule collection to configure whether rules are enforced or audited. If rule enforcement has not been configured, rules will be enforced by default.

Configure rule enforcement

More about rule enforcement

Overview

Executable Rules
Rules: 0
Enforcement not configured: Rules are enforced

Windows Installer Rules
Rules: 0
Enforcement not configured: Rules are enforced

Script Rules
Rules: 0
Enforcement not configured: Rules are enforced

Packaged app Rules
Rules: 0
Enforcement not configured: Rules are enforced

Figure 8.16 AppLocker rules.

We will only add in the default rules. The reason we are choosing the defaults rules is that way we are less likely to brick the system.

Please select each of the above Rule groups (Executable, Windows Installer, Script and Packaged) and do the following for each of them; right click in the area that says 'There are no items to show in this view'. And then select 'Create Default Rules', as shown in Figure 8.17. This should generate a subset of rules for each group. Next, we need to enforce the rules we just created. Select AppLocker on the far left pane. Then, you will need to select 'Configure Rule Enforcement', shown in Figure 8.18. This will open a pop-up; you will need to check Configured for each set of rules.

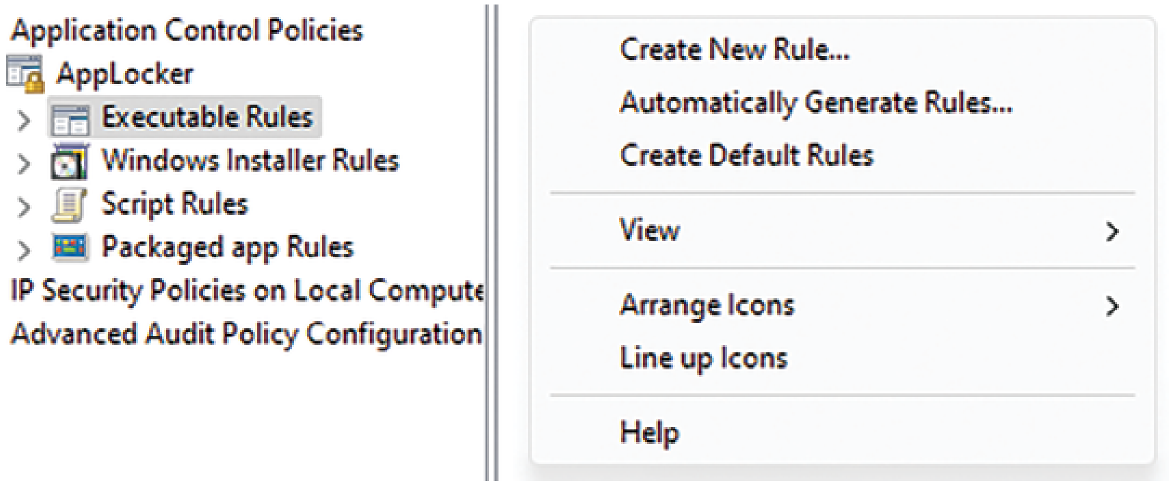
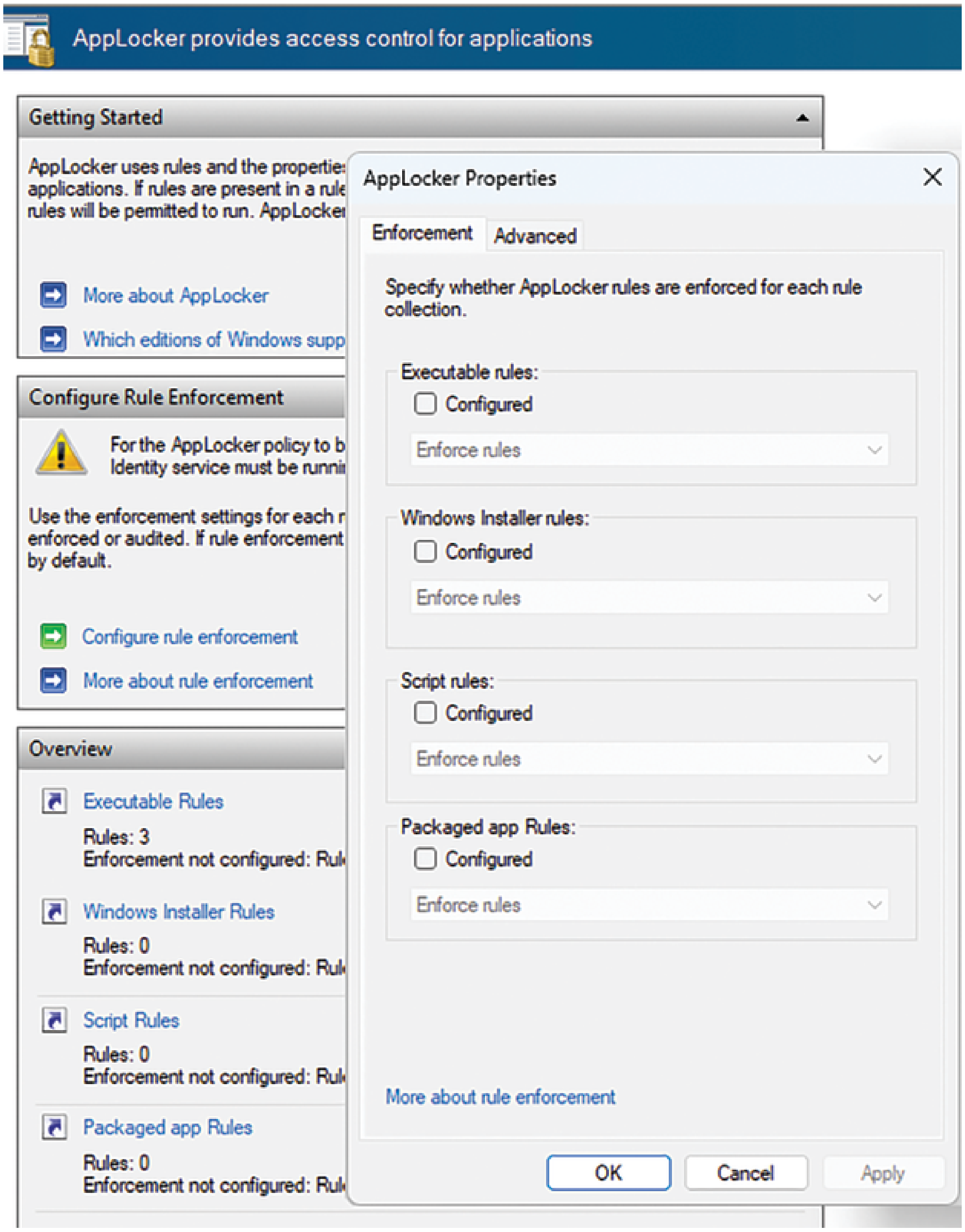


Figure 8.17 Create default rule for AppLocker.



AppLocker provides access control for applications

Getting Started

AppLocker uses rules and the propertie
applications. If rules are present in a rule
rules will be permitted to run. AppLocke

More about AppLocker

Which editions of Windows supp

Configure Rule Enforcement

For the AppLocker policy to b
Identity service must be runni

Use the enforcement settings for each r
enforced or audited. If rule enforcement
by default.

Configure rule enforcement

More about rule enforcement

Overview

Executable Rules
Rules: 3
Enforcement not configured: Rul

Windows Installer Rules
Rules: 0
Enforcement not configured: Rul

Script Rules
Rules: 0
Enforcement not configured: Rul

Packaged app Rules
Rules: 0
Enforcement not configured: Rul

AppLocker Properties

Enforcement | Advanced

Specify whether AppLocker rules are enforced for each rule collection.

Executable rules:
☐ Configured
Enforce rules

Windows Installer rules:
☐ Configured
Enforce rules

Script rules:
☐ Configured
Enforce rules

Packaged app Rules:
☐ Configured
Enforce rules

More about rule enforcement

OK | Cancel | Apply

Figure 8.18 **Enforce AppLocker.**

We are still not done yet. We also need to start the Application Identity service. Press the Windows key and typing ‘Services’, as shown in Figure 8.19; press enter. This will bring up the Services App.

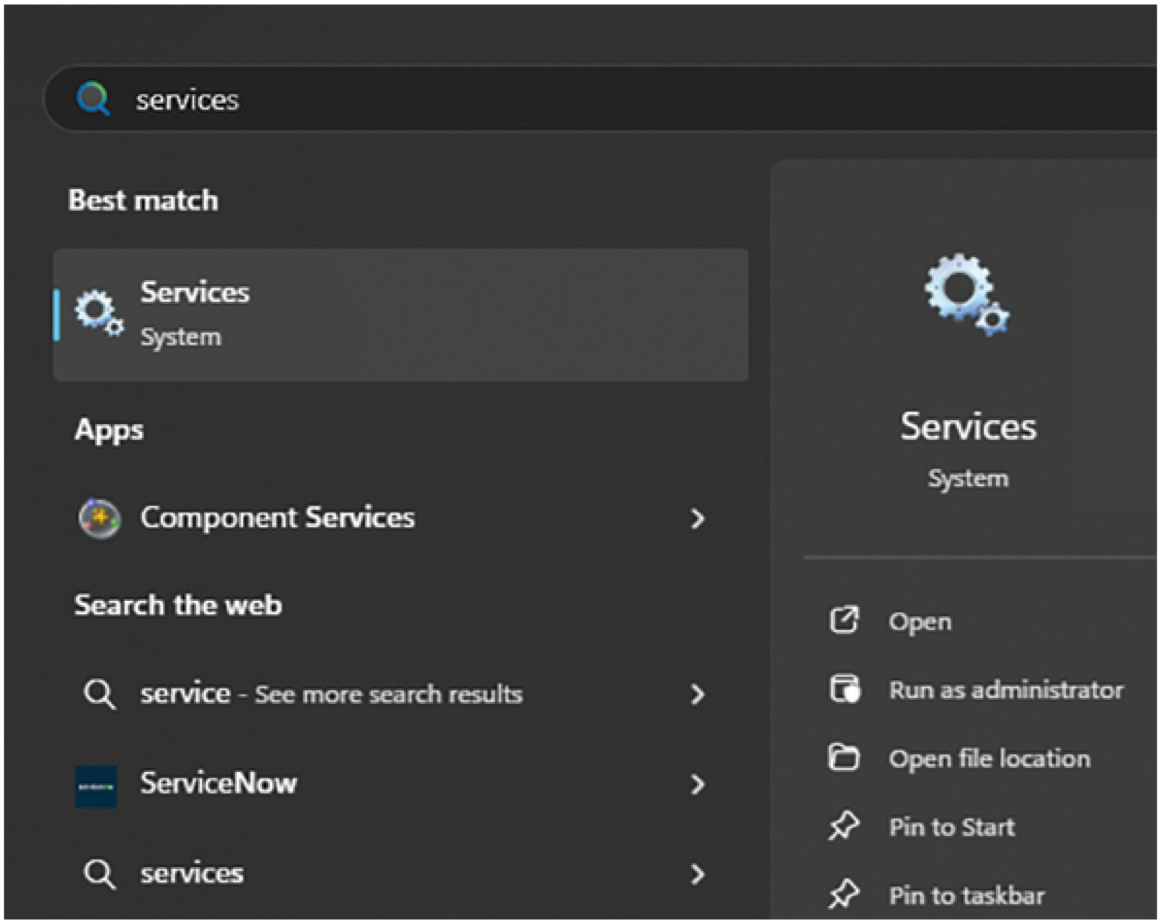
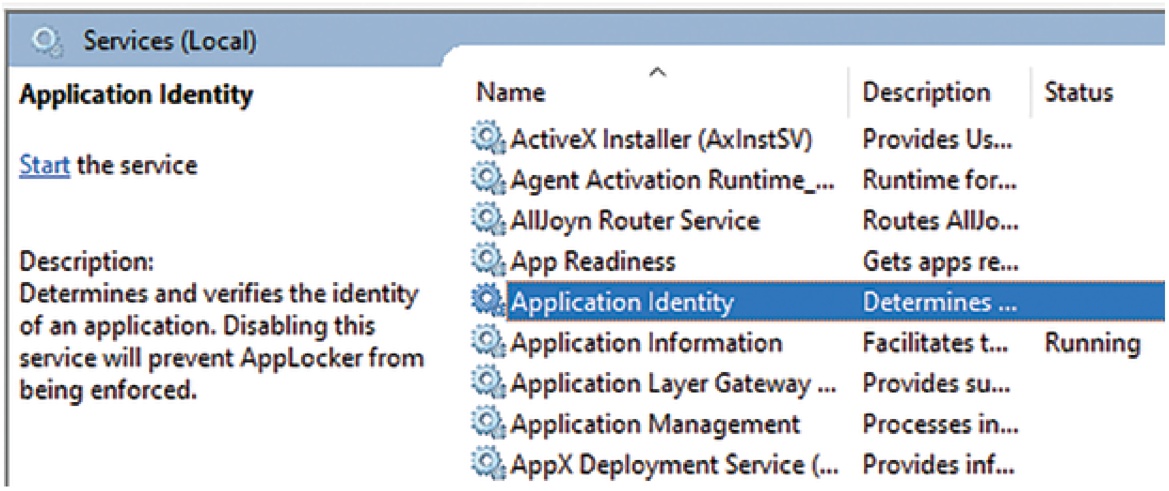


Figure 8.19 **Search bar service.**

With the service window open, find and select Application Identity. With the Application Identity service selected, press the 'Start' button, as shown in Figure 8.20. This will start the service.



Services (Local)

Application Identity

Start the service

Description:
Determines and verifies the identity of an application. Disabling this service will prevent AppLocker from being enforced.

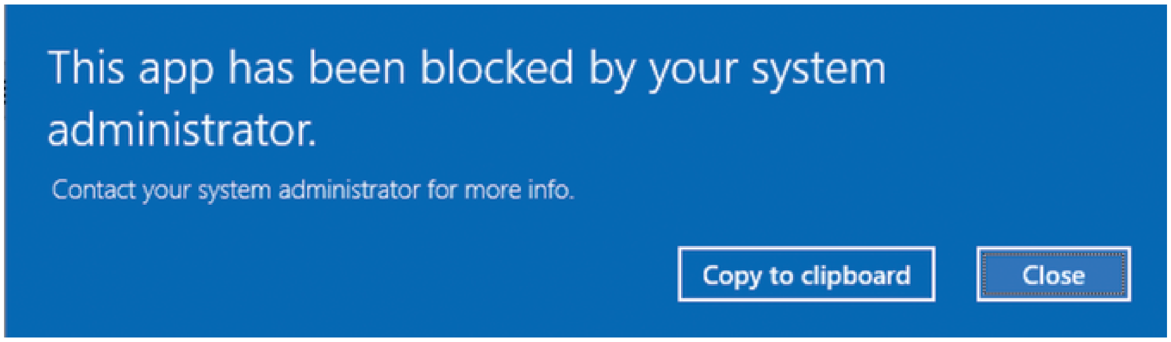
Name	Description	Status
ActiveX Installer (AxInstSV)	Provides Us...	
Agent Activation Runtime_...	Runtime for...	
AllJoyn Router Service	Routes AllJo...	
App Readiness	Gets apps re...	
Application Identity	Determines ...	
Application Information	Facilitates t...	Running
Application Layer Gateway ...	Provides su...	
Application Management	Processes in...	
AppX Deployment Service (...	Provides inf...	

Figure 8.20 **Windows services.**

Next, open a command prompt and run gpupdate to force the policy changes we have just made to the system.

```
gpupdate /force
```

Now, when you try to execute any executable (.exe) file that is not located inside 'programfile' folder. It should return with an error message, similar to shown in Figure 8.21.


This app has been blocked by your system administrator.

Contact your system administrator for more info.

Figure 8.21 AppLocker message.

It is important that you also lockdown the 'programfile' folder, by removing user permissions to write to the folder. This will remove the possibility of a threat actor, just adding an executable to the folder and just executing it from there. Application allow listing can also be applied to Linux machines; the difference here is that instead of having a built-in policy engine, you can use an application called AppArmor,[4] which comes pre-installed on some Linux distribution; to check if AppArmor is available, use the command `'aa-status'`. Using AppArmor it is possible to achieve a similar goal, as we got on Windows.

8.8 Summary

EDR is a vital component of modern cybersecurity strategies, providing critical visibility and response capabilities. With the rise of encrypted traffic, EDR's role in monitoring endpoint activities is becoming even more significant. In this chapter, we manage to deploy Velociraptor, which is an open-source EDR tool that offers a robust suite of features suitable for a wide range of environments, by adding Velociraptor to our environment and showcased some of the features available in Velociraptor.

Notes

1 https://docs.velociraptor.app/docs/

2 https://docs.velociraptor.app/docs/deployment/clients/

3 https://docs.velociraptor.app/docs/

4 https://apparmor.net/

Chapter 9
Security Information and Event Management

> During your morning meeting with the security analysts, a concerning pattern emerges. The team is spending hours each day manually checking multiple security systems, making it nearly impossible to correlate events or identify sophisticated attacks that span multiple systems. This fragmented approach not only reduces team efficiency, but also increases the risk of missing critical security incidents. It's clear that centralising security monitoring has become a business imperative.

With the current setup, we have two places that are currently monitoring for security-related events. That being network monitoring using Suricata, and the endpoint protection with Velociraptor. With our small setup of just two monitoring tools, it is not too painful to check each monitoring system for potential incidents. That a completely different story for large organisations, it is not unrealistic to have between 15 and 20 security tools, and monitoring all of them for alerts can be difficult and time-consuming.

That is what Security Information and Event Management (SIEM) is supposed to help with by centralising all the alerts. This is done by collecting all the events from the different security tools into a single place. Enable analysts to manage and analyse these logs in a centralised location. Another benefit of SIEM is that it makes it possible to correlate security alerts from different tools, allowing for new detection possibilities. When selecting a SIEM, it is important to consider the volume of data that the system needs to handle, as the amount of logs by different systems, applications and devices can be enormous. Then there is also the variety in the format of the logs generated, which needs to be parsed and analysed by the SIEM system. Having centralised log management provides several benefits:

- Visibility and Correlation: Centralising logs allows security teams to have a unified view of all activities across the network. This visibility is essential for identifying and correlating events that may indicate a security incident.

- Efficiency in Incident Response: A centralised repository enables faster and more efficient incident response. Security analysts can access all relevant logs from a single location, speeding up the process of investigating and mitigating threats.
- Compliance and Reporting: Many regulatory frameworks require organisations to retain and monitor logs. Here, a centralised log management system simplifies compliance by collecting, storing and making logs easily accessible for auditing and reporting purposes.
- Proactive Threat Detection: Advanced SIEM systems can analyse logs in real time to detect anomalies and potential security threats. By centralising logs, organisations can leverage SIEM tools to enhance their threat detection capabilities.

There exists a whole host of SIEM solutions out on the market. At the time of writing, the most popular are Splunk, QRadar and Azure Sentinel. This is constantly changing, as new ones get added to the market. I would like to highlight that most SIEM solutions are priced based on the volume of events that are sent to the SIEM. Depending upon your network, the volume of logs can be so large that the price of the SIEM will break the security budget. I always felt that this way of monetising is counterintuitive to what an SIEM is supposed to be. Because of this miniaturisation model, you have to think about what logs are being sent to the SIEM and their importance for identifying and mitigating threats. This removes the idea of being a centralised log management system.

It is natural to think: why spend all these resources on monitoring? Would it not be better to focus on protecting instead? That way, we can just spot the attacks from ever happening in the first place. The thing about prevention is that it will fail at some point. When that happens, it is critical to have a method of detecting and responding to the attacks that go through the defences. 'Prevention is nice, detection is a must'. That is the reason we use the majority of our time on detection capabilities.

9.1 SIEM Architecture

An SIEM system's effectiveness depends on its architectural design and the integration of its core components. The security events generated by the security tools will run through these steps in order to become part of the SIEM:

1. Security tools generate logs and events.
2. Collection methods gather and forward data to the processing engine.
3. The processing engine normalises and enriches the data.
4. Enriched data is stored and indexed.
5. The analysis interface queries stored data for visualisation and investigation.

Understanding these steps and their interactions is crucial for building a robust security monitoring capability. At the heart of any SIEM implementation are four fundamental components that work together to provide comprehensive security monitoring.

9.2 Log Collection

Before we can get started with building our SIEM, we need to have logs for us to process. For this, we need to find a way to collect the logs from the different systems existing on the network. This might seem basic, but a lot of planning needs to be done in order to facilitate log collection. The complexity of the setup depends upon how many types of systems exist in the environment. The two most common methods of collecting logs are using an agent that sits in the system. The other is agentless, using scripts, application programmable interfaces (APIs) or other tools for collecting the logs.

- Agent: Requires software to be installed on the host.
- Agentless: Requires credentials to access and collect logs from the system.

Agentless collection works by having a server that remotely logs in to the system and pulls the logs from the system. This requires the collector to have credentials for every remote system on the network, and can reach the systems over the network. You can use PowerShell on Windows, SSH on Linux and Mac or an API for cloud services to accomplish this process. The major advantage of this is not needing to deploy and maintain a logging agent on every system. The downside is that it cannot handle an infinite number of host systems, requiring there to be multiple collectors. It also introduces a security risk by having a single system being trusted by all the other systems on the network. A threat actor will most likely try to capture these credentials, and pivot to new targets. To minimise this risk, always limit their access to only what is strictly necessary for that service account(s) to collect the logs. A good idea is to split up the service account's permission, so no single account has access to all the devices on the network.

For a log agent, the feature set of modern log agents is long and listing them all would be too much and what features are important depends on your environmental needs. Here are some things to consider when evaluating what agent to deploy. Its ability to integrate with the current environment. The system overhead of having the agent constantly running on the system, and the ability to orchestrate the agent in a centralised manner. Nothing is worse than having to redeploy agents every time you need to reconfigure them. A bonus is its ability to encrypt the logs in transit to the SIEM. For open source agents Fluentd[1] is an excellent choice and has many of the same features seen in commercial tools. Another popular tool we will use in this chapter is Logstash[2] is not open-source, but free to use and has many of the basic features required. That is not all; organisations also need to plan their log collection. If this is not properly done, it can result in missing or unparsed logs. To create an effective log collection plan, you need to understand the most common log sources that are used by the different systems and their relevance for your organisation's detection capability. Let's go a little deeper into the most common types of log formats that are being used.

9.2.1 Syslog

The number of devices that support Syslog is plentiful. Nearly all network devices, servers and system administrators' applications support Syslog. Both Linux and Mac support Syslog natively. Windows requires the installation of a third-party agent that takes the Windows event logs and sends them out as Syslog. Because of the widespread use of Syslog, we need to understand Syslog and ensure that our system can parse the logs correctly. Syslog has been around for a long time, dating back to the 1980s, a time when security was not a major concern. Sending logs in plain text over User Datagram Protocol (UDP) made sense then; the problem is they are accessible to anyone looking at the network traffic. Over time, developers have updated Syslog multiple times, changing the layout of the logs and adding support for Transmission Control Protocol (TCP) and Transport Layer Security (TLS) encryption. This also means that not all instances of Syslog are using the same version, which can lead to inconsistency. For the log layout, Syslog consists of five fields, as shown below. In the following order, priority (PRI), timestamp, source host, source process and message.

```
<14>Mar 4 11:20:45 host sudo: {message}
```

One of the most important field of Syslog to understand is the PRI field, which is an integer surrounded by '<>'. This integer is a mathematical calculation that stores the Syslog facility and severity fields. This method minimises the amount of data needed to be sent over the network, which was important back in the day. We use the following calculation to identify the facility and severity of the log message.

- Facility = PRI/8 (rounded down to the nearest whole number)
- Severity = PRI – (8 * facility number)

Following the above example, we performed the following calculation for facility and severity level.

- Facility = 14/8 = 1.75. This is rounded down to 1, which is user-level messages
- Severity = 14 – (8 * 1) = 6. So, the severity is 6, which is informational

Below you can see a snippet of the facilities and severity codes that exist.

Numerical Code Facility:

0. kernel message

1. user-level messages

2. mail system

3. system daemons

4. security/authorisation messages

5. messages generated internally by syslogd

6. line printer subsystem

7. network news subsystem

8. Unix-to-Unix Copy (UUCP) subsystem

9. clock daemon

10. security/authorization messages

11. FTP daemon

12. NTP subsystem

13. log audit

14. log alert

Security Code:

0. Emergency

1. Alert

2. Critical

3. Error

4. Warning

5. Notice

6. Informational

7. Debug

As I mentioned above, varying implementations of Syslog prevent consistent messaging. The layout of one vendor and application can vary dramatically from the next. This may require a parser for each application, adding a lot of overhead to the log collection. When it comes to parsing, Syslog is not that difficult, it's just time-consuming. Parsing is usually done using regular expressions (regex). Regex works by specifying a pattern that it matches against. Going into depth about regex is beyond the scope of this book. There are almost an infinite number of places to learn regex to go sites like regexlearn,[3] to learn how to do searches using regex.

A limitation for Syslog that you should be aware of is the message size. For logs sent over UDP, the maximum size is 1024 bytes and for TCP, the maximum it 4096 bytes. Most logs will fit within these ranges. That does not mean everything will, especially for Windows event logs, which can be over 30 KB in size, yet many agents choose to use Syslog to transmit these logs. The way they do this is by splitting the log into multiple packages. This adds an additional requirement to the receiver of the logs. They must now also have the ability to reassemble the fragmented logs.

9.2.2 Windows Events

With logging on Windows, it uses the built-in logging feature called Windows Events. This system separates events into different channels. The three most common are Application, Security and System. But there are many more. Each event type is assigned a unique ID, enabling quick searches for specific events that interest you. They even stored these in a proprietary binary format, but later switched to XML format. The parameter-based field storage allows for easy parsing and automatic field extraction. The downside of the Windows event log is there are many events happening in the system that it does not log. This includes things like command-line execution. For that, we have to look elsewhere.

9.2.3 Sysmon

Since most organisations are Windows-based, we also need to take a look at Sysmon, which is available on all modern Windows systems. Sysmon uses the built-in Windows API calls to generate the logs, making the performance impact on the system minimal. Sysmon provides the ability to generate logs with detailed information about process creations, network connections and changes to file creation time. For example, Sysmon can monitor every process that is being created, including the parent process that spawns new processes, and provides a hash of each new process. Collecting Sysmon allows you to see exactly what is taking place inside your Windows hosts. Having this level of detail is great for incident response, but also troubleshooting for the system administrator. Install Sysmon on all systems to ensure log availability during incidents. However, the volume of data generated might make sending all Sysmon logs to the SIEM impractical. It might be a good idea to do some filtering first, to limit the amount of data being sent. When it comes to installing Sysmon, the first step is to download Sysmon from Microsoft.[4] The following command is used to install Sysmon with the configuration.

```
Sysmon64.exe -accepteula -i sysmonconfig-export.xml
```

To view the logs generated by Sysmon, just open **Event Viewer**. To find the output of Sysmon, head over to `Applications and Services Logs` > `Microsoft > Windows > Sysmon > Operational`. You might have noticed the 'sysmonconfig-export.xml' file. When installing Sysmon, the '-i' flag tells Sysmon, which configuration file should be used to define what logs Sysmon will be collecting. A good thing is that this configuration file is written in XML, making it easily editable. The configuration file we are using is just a starting point.[5] Remember to adjust the configuration file to fit your environment's needs. You can read more about the configuration file over at Microsoft.[6] The main thing that you should pay attention to is the usage of onmatch="include" and onmatch="exclude". 'onmatch="exclude"' logs all events except those specified. In the code example below, it will log all drivers except for the ones with the signatures of Microsoft and Windows.

```
<DriverLoad onmatch="exclude">
  <Signature condition="contains">microsoft</Signature>
  <Signature condition="contains">windows</Signature>
</DriverLoad>
```

With onmatch="include", it will only log the event that is especially specified. The configuration example below configures Sysmon to only log network connections to port 80 or 443.

```
<NetworkConnect onmatch="include">
  <DestinationPort>443</DestinationPort>
  <DestinationPort>80</DestinationPort>
</NetworkConnect>
```

When you are satisfied with the Sysmon configuration, you can easily deploy it to other systems using a group policy object. We will not explain how to do this in this book.

9.3 Data Processing Engine

The processing engine converts raw log data to a standardised format suitable for analysis. This component performs several critical functions:

- Log parsing and normalisation to convert varied data formats into a consistent structure.
- Event enrichment by adding contextual information, such as threat intelligence or asset data.
- Data correlation to identify relationships between events from different sources.
- Real-time analysis to detect security incidents as they occur.

9.4 Log Enrichment

With the logs collected from different sources, it is a good idea to perform log enrichment. This is either done by the collection process or an internal capability within the SIEM. Enrichment is simply the process of adding additional context to the log. This will not only help the security analyst better understand the context around the logs, it can also help with detection capability, giving additional attributes to build detection rules around. We often enrich domains with information such as the registrant name, IP address and Advanced Shipping Notice (ASN) to better assess them. This is just an example—what and how you enrich your logs depends on your security setup and the capabilities that they come with.

9.5 Storage and Retention

The storage component maintains both hot (frequently accessed) and cold (archived) security data. This dual-storage approach balances the need for rapid access to recent events with cost-effective retention of historical data. Storage requirements must account for:

- Regulatory compliance retention periods.
- Investigation and forensics needs.
- Performance requirements for data retrieval.
- Cost considerations for long-term storage.

9.6 Analysis and Visualisation Interface

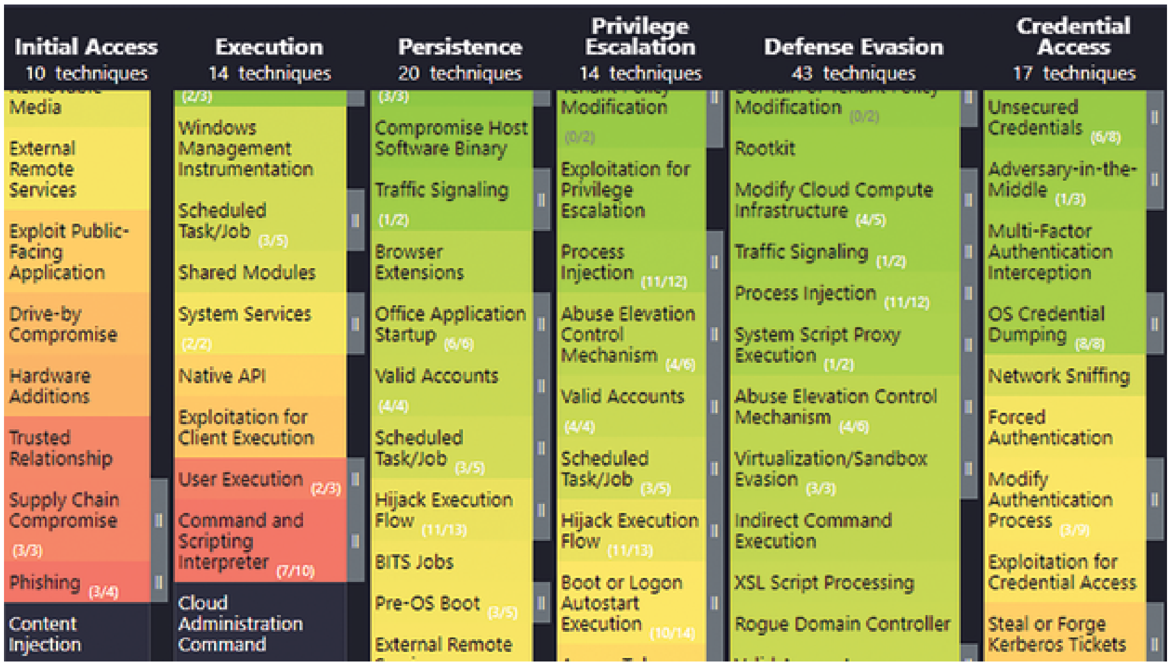
The analysis interface provides security analysts with visual interfaces to help investigate incidents and monitor security posture. Key capabilities include:

- Real-time dashboards for security monitoring
- Advanced search capabilities for investigations
- Automated reporting for compliance and metrics
- Alert management and incident response workflow integration

9.7 What to Log?

As stated above, the problem with many SIEM solutions is that we pay for the volume of the logs that are being ingested by the SIEM. To combat this, what we need to do is to maximise the value per log. Each log must be directly linked to an alerting rule within the SIEM. All other logs are just nice to have. They should first be collected at the point of incident, to save on the security budget. The problem is, I cannot just tell you what logs you should collect, because it depends on the environment you are working in. Which should define what log you are interested in. Making it impossible to create a list of the logs that should be collected and sent to the SIEM. That does not mean I want to leave you empty-handed.

To answer this question, we have to look at a tool called MITRE ATT&CK.[7] The full name is Adversarial Tactics, Techniques and Common Knowledge (ATT&CK). It is a knowledge database of adversary tactics and techniques based on real-world observations. It provides a common language for describing cyber adversary behaviour when conducting attacks on organisations. ATT&CK is organised into matrices, which are divided into tactics, each representing different stages of the attack lifecycle, e.g. initial access, execution, persistence and privilege escalation. Within each tactic, there are techniques and sub-techniques that describe the specific methods the threat actors used to achieve their objectives. How does this help us with logging? The way we are going to use ATT&CK is by taking the logs you are sending to the SIEM and mapping them to a technique or sub-technique of MITRE ATT&CK. That way, you are sure that each log we are sending actually has a value. Even better, you can identify which techniques you do not have any monitoring for. To help with this mapping, Mitre has created an application called ATT&CK Navigator,[8] it is a web application that provides a graphical interface for exploring and annotating the ATT&CK matrices. ATT&CK Navigator allows users to create, annotate and share layers that can be overlaid on the ATT&CK matrix. This allows you to create a visual representation that clearly identifies which techniques should be prioritised in your logging efforts, similar to what is shown in Figure 9.1. This visual approach makes it easier to communicate priorities across teams and stakeholders, ensuring everyone is aligned with the most critical areas of focus.



Initial Access 10 techniques	Execution 14 techniques	Persistence 20 techniques	Privilege Escalation 14 techniques	Defense Evasion 43 techniques	Credential Access 17 techniques
Media	(2/3)	(3/3)	Modification (0/2)	Modification (0/2)	Unsecured Credentials (6/8)
External Remote Services	Windows Management Instrumentation	Compromise Host Software Binary	Exploitation for Privilege Escalation	Rootkit	Adversary-in-the-Middle (1/3)
Exploit Public-Facing Application	Scheduled Task/Job (3/5)	Traffic Signaling (1/2)	Process Injection (11/12)	Modify Cloud Compute Infrastructure (4/5)	Multi-Factor Authentication Interception
Drive-by Compromise	Shared Modules	Browser Extensions	Abuse Elevation Control Mechanism (4/6)	Traffic Signaling (1/2)	OS Credential Dumping (8/8)
Hardware Additions	System Services (2/2)	Office Application Startup (6/6)	Valid Accounts (4/4)	Process Injection (11/12)	Network Sniffing
Trusted Relationship	Native API	Valid Accounts (4/4)	Scheduled Task/Job (3/5)	System Script Proxy Execution (1/2)	Forced Authentication
Supply Chain Compromise (3/3)	Exploitation for Client Execution	Scheduled Task/Job (3/5)	Hijack Execution Flow (11/13)	Abuse Elevation Control Mechanism (4/6)	Modify Authentication Process (3/9)
Phishing (3/4)	User Execution (2/3)	Hijack Execution Flow (11/13)	Boot or Logon Autostart Execution (10/14)	Virtualization/Sandbox Evasion (3/3)	Exploitation for Credential Access
Content Injection	Command and Scripting Interpreter (7/10)	BITS Jobs		Indirect Command Execution	Steal or Forge Kerberos Tickets
	Cloud Administration Command	Pre-OS Boot (3/5)		XSL Script Processing	
		External Remote		Rogue Domain Controller	

Figure 9.1 Mitre navigator.

To effectively implement ATT&CK Navigator in your logging strategy, consider following these steps:

1. Begin with an initial assessment by creating a baseline layer in Navigator that represents the current logging capabilities. This baseline will serve as the starting point.
2. Next is gap analysis. Use your baseline layer to highlight techniques where logging is inadequate or missing. This visual representation of gaps will guide your efforts.
3. After the gap has been identified, it is time for prioritisation. Use Navigator's scoring feature to rate techniques based on risk and current capabilities. This prioritisation will help you focus on the most critical areas first.
4. With your gaps identified and prioritised, develop an action plan to address the gaps in your logging. This plan should outline specific steps, timelines and resources needed to improve your logging capabilities.
5. When executing your plan, focus on implementing controls for the most critical systems first. Regularly update the Navigator layers to

reflect these improvements, providing a clear visual representation of the progression you are making.

Finally, make it a practice to review and update your Navigator layers regularly. This ongoing process will help you adapt to changing threat landscapes and ensure your logging strategy remains effective and up-to-date. This approach ensures your strategy aligns with the latest understanding of adversary tactics and techniques, providing a sound foundation for your overall cybersecurity posture.

9.8 Zero Trust SIEM

SIEM overlaps with Zero Trust core design principles of continuously monitoring the environment, allowing us to get a clear view of what is going on. We use this knowledge to define what controls and alerts that need to be implemented into our environment. When working with SIEM, there are also trust concerns that need to be handled. Just like you would do with other systems, start from the inside out. Since we are working with the SIEM at the moment, that is where we will be starting. Start by defining who needs access to the system – that would, in most cases, only be the security team. We should also ask what the system needs access to, which would be where the logs exist; these logs should either be pushed or pulled to the SIEM. Finally, the SIEM itself must log all activities to ensure the trust we have in the security people is not misused. Like any other security tool, having strong authentication is critical. It is essential to ensure that only authorised users have access to the sensitive data stored within the SIEM. That means that we not only have to protect the data at rest with encryption but also as it is transferred to the SIEM from security tools.

For our project, we will be throwing out all the security best practices, in favour of convenience. We aim to require as minimum steps from your side as possible for setting up the SIEM. That also means that we will be leaving it completely open to everyone with access to the system. This should, of course, never be the case for any system in production. First, let’s add SIEM to our transaction map. Here, we define that the SIEM will be interacting with both Velociraptor and Suricata. The updated transaction map can be seen in Table 9.1.

TABLE 9.1

Transaction map.

	CRM	IAM	Physical security	Web store	Backup	Wireless	Email	Internet	Network monitoring	Jumpbox	EDR	SIEM	
Customer relation management (CRM)	■				■	■							
IAM	■	■	■	■		■	■	■					■
OT			■					■			■		■
Physical security		■		■									
Web store					■	■			■				■
Backup					■	■		■					
Wireless		■					■		■				■
Email		■			■	■	■	■	■				■
Internet					■		■	■	■				
Network monitoring										■			■
Jump box			■								■		
Endpoint Detection and Response	■		■		■							■	■
SIEM		■	■		■			■		■		■	■

Now that we have an idea of what components the SIEM needs to communicate with, let's go to the implementation phase.

9.9 Implementing SIEM with ELK

The SIEM used in our project will be a software suite consisting of Elasticsearch, Logstash and Kibana. This suite is also known as the ELK stack. This software package allows us to centralise the logs from both Suricata and Velociraptor into a single place. This design uses Elasticsearch as the database, the place where all logs will be stored from different systems. Kibana for visualising the data stored inside Elasticsearch and the last piece is Logstash. It is the log ingestion platform that sends the data from different sources to Elasticsearch. This suite provides a powerful and scalable platform for both log management and analysis. In the updated network diagram shown in Figure 9.2, we can see the SIEM has been added to the network, inside the security zone, giving it access to the security environment that we need to collect logs from.

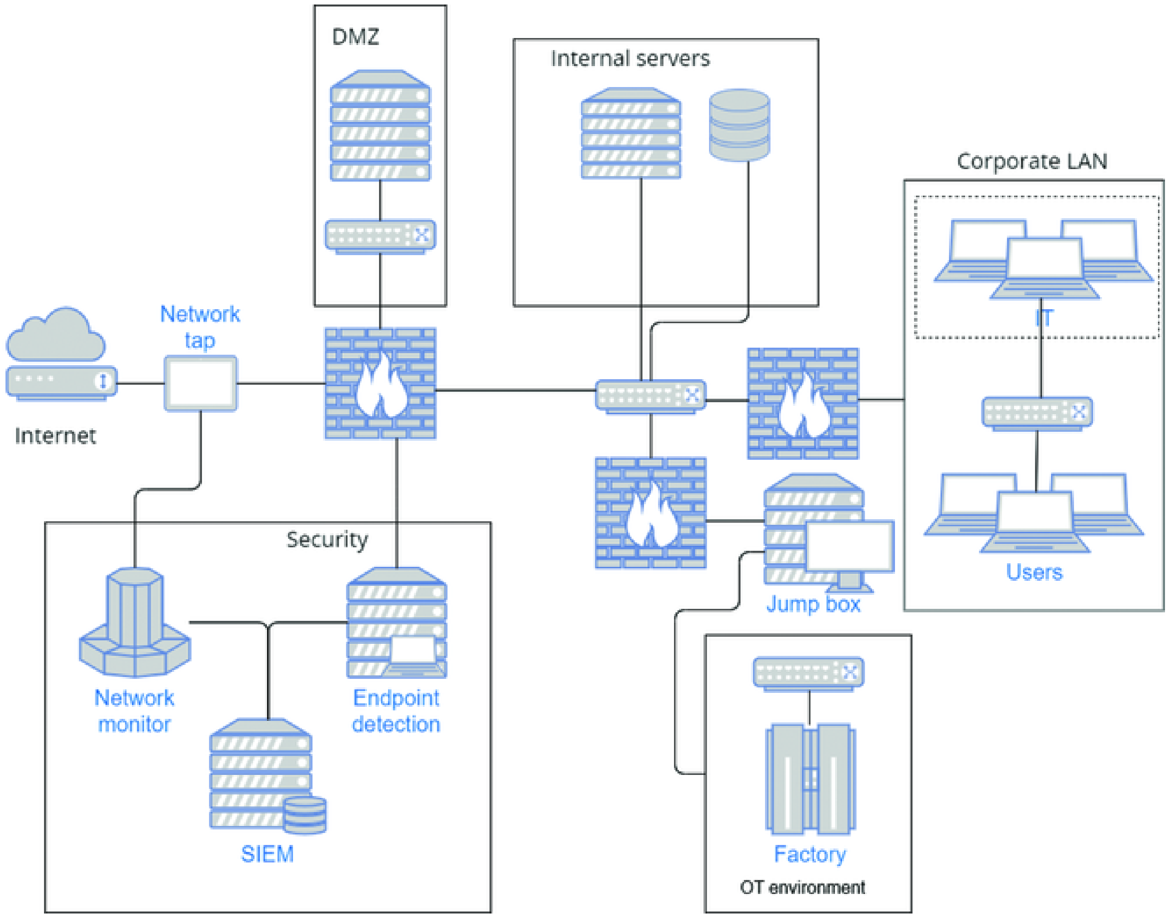


Figure 9.2 Network diagram for SIEM.

Now that we have an idea of what we are going to be implementing, let's get started building it. Let's look at the Docker configuration file and changes that we need to make for this SIEM implementation. This includes the following new containers: Elasticsearch, Kibana and Logstash.

```
Logstash:
  container_name: Logstash
  depends_on:
    - elasticsearch
    - kibana
  image: Logstash:8.14.1
  labels:
    co.elastic.logs/module: Logstash
  user: root
  volumes:
    - "./suricata/logs:/var/log/suricata"
    -
"./Logstash.conf:/usr/share/Logstash/pipeline/Logstash.
conf:ro"
  environment:
    - xpack.monitoring.enabled=false
    - ELASTIC_HOSTS=https://es-container:9200
  networks:
    - security
elasticsearch:
  container_name: es-container
  image:
docker.elastic.co/elasticsearch/elasticsearch:8.14.1
  environment:
    - xpack.security.enabled=false
    - "discovery.type=single-node"
  ports:
    - 9200:9200
  networks:
    - security
kibana:
  container_name: kb-container
  image: docker.elastic.co/kibana/kibana:8.14.1
  environment:
    - ELASTICSEARCH_HOSTS=http://es-container:9200
  depends_on:
    - elasticsearch
  ports:
    - 5601:5601
  networks:
    - security
```

This Docker Compose configuration adds the Logstash, Elasticsearch and Kibana containers to the environment, including port mappings for external access to Elasticsearch (9200) and Kibana (5601). You could secure this design even further by restricting access to Elasticsearch from the host machine, as we will not be communicating directly with the database. With the SIEM environment in place, the next step is to integrate the logs from Suricata and Velociraptor into Elasticsearch. We will start with integrating the logs from Suricata into Elasticsearch. This is done by having Suricata export logs to a place that Logstash can read from. This step should have already been done in the previous chapter, and be available on the host machine. Then there is also the configuration of Logstash to have the permission to read logs from the host and parse them, and have them sent to our Elasticsearch. All of this should already be set up for you. The following guide is just to help you understand how it works.

Giving the Logstash container access to the suricata logs is quite easy. Since we have already piped the logs from the Suricata to our host machine. From there, all we have to do now is to take the logs and have them piped from the host to the Logstash container. This is done in the volume part of the Logstash configuration.

```
Logstash:
  container_name: Logstash
  ...
  volumes:
    - "./suricata/logs:/var/log/suricata"
    -
"./Logstash.conf:/usr/share/Logstash/pipeline/Logstash.
conf:ro"
  ...
```

The "./suricata/logs:/var/log/suricata" under volume settings. Tells Docker to give the Logstash container access to the files on the host located at "./suricata/logs" relative to the Docker compose file location. I have already set up everything, so when you spin up the SIEM environment, everything should be ready to go. For configuring the behaviour of Logstash, that is done by editing the logstash configuration file, called Logstash.conf. Below is how Logstash is configured for our environment to achieve the goal of integrating Suricata into Elasticsearch.

```
input {
  file {
    path => "/var/log/suricata/eve.json"
    start_position => "beginning"
    sincedb_path => "/dev/null"
    codec => "json"
  }
}
filter {
  date {
    match => ["timestamp", "ISO8601"]
  }
}
output {
  stdout {}
  elasticsearch {
     hosts=> "es-container:9200"
     index=> "Suricata"
  }
}
```

To make this simpler to understand, let's take the configuration and split it into three sections: input, filter and output. Starting with the input section of the configuration. This specifies what data we would like Logstash to be reading. For our case, it is the Suricata logs, which are available to the Logstash container at '/var/log/suricata/eve.json'.

```
input {
  file {
    path => "/var/log/suricata/eve.json"
    start_position => "beginning"
    sincedb_path => "/dev/null"
    codec => "json"
  }
}
```

- file: Specifies the input plugin to read from a file.

- path: The file path where Logstash will try to read logs from, in this case, ‘/var/log/suricata/eve.json‘.
- start_position: When Logstash starts, it reads the file from the ‘beginning‘. This is useful if you want to process the entire file from the start.
- sincedb_path: This setting is used to track the current position in the file. Setting it to ‘/dev/null‘ means Logstash will not keep track of the position, and it will read the file from the beginning every time it starts.
- Codec: Specifies the codec used to decode the input data. ‘JSON‘ means the input data is in JSON format and should be parsed as JSON.

Filter section defines which data read from the input should be sent to the output. In these cases, the only requirement is that the timestamp be in ISO8601 format.

```
filter {
  date {
    match => ["timestamp", "ISO8601"]
  }
}
```

- filter: Specifies the filter plugin to process the events.
- date: The date filter parses dates from fields to be used as Logstash event timestamps.
- match: Defines the field and the date format. Here, it matches the ‘timestamp‘ field in the JSON input and expects it to be in ‘ISO8601‘ format (e.g. ‘2021-07-03T12:34:56Z‘).

The last part of the configuration is the output section, which specifies what should happen to the data at the end. We will be sending it to the Elasticsearch database.

```
output {
  stdout {}
  elasticsearch {
    hosts => "es-container:9200"
    index => "Suricata"
  }
}
```

- output: Specifies the output plugin to send processed events to destinations.
- elasticsearch: Sends events to an Elasticsearch instance.
- hosts: The address of the Elasticsearch server, here it is ‘es-container:9200’.
- index: The name of the Elasticsearch index where the events will be stored. Here, it is ‘Suricata’.

Those are the steps that have to be taken in order to integrate Suricata with Elasticsearch. The same process can be done with any other logs you would like to integrate into Elasticsearch. The process might be a little different, depending on the log format.

For integrating Velociraptor with Elasticsearch, it comes with built-in support, but it has to be set up manually. You can read more about the integration here.[9] First head over to https://localhost:8889 to access Velociraptor. Once you have access to the server, press on server events, similar to what is shown in Figure 9.3.

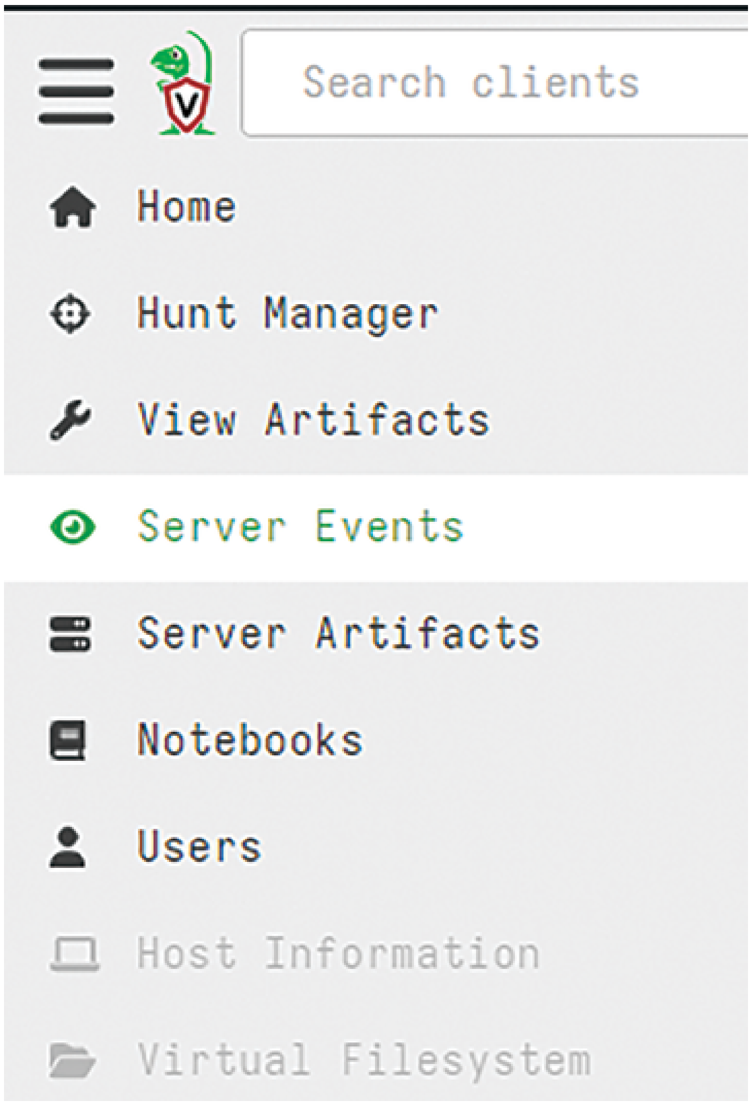


Figure 9.3 Server event.

Click on the edit icon, displayed in Figure 9.4

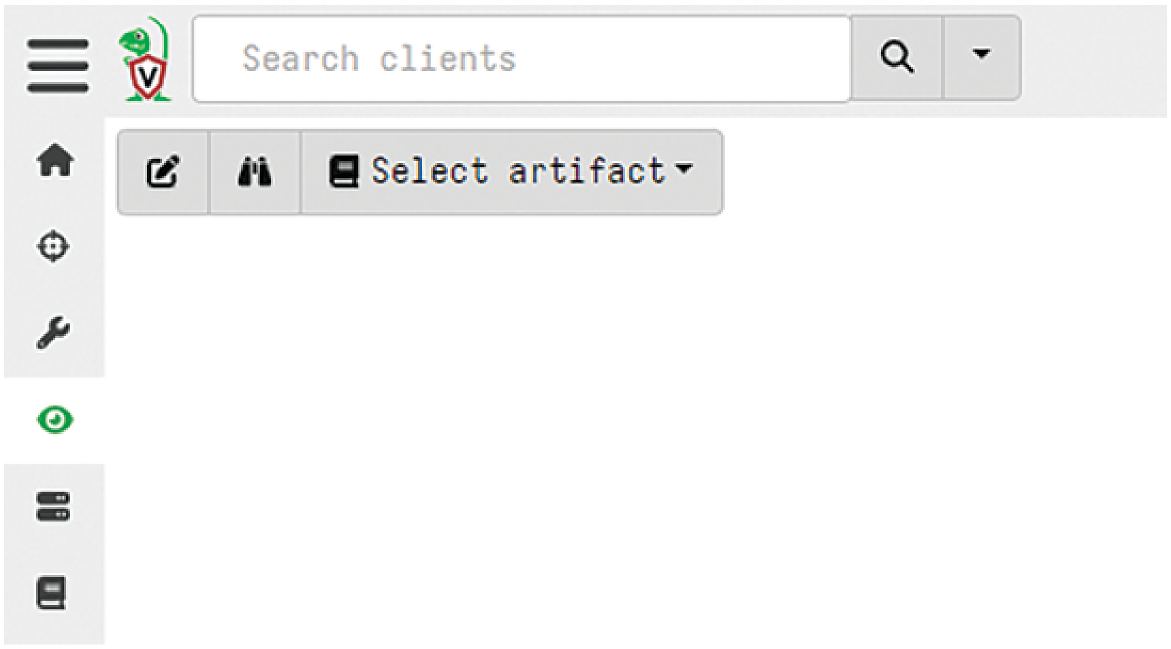
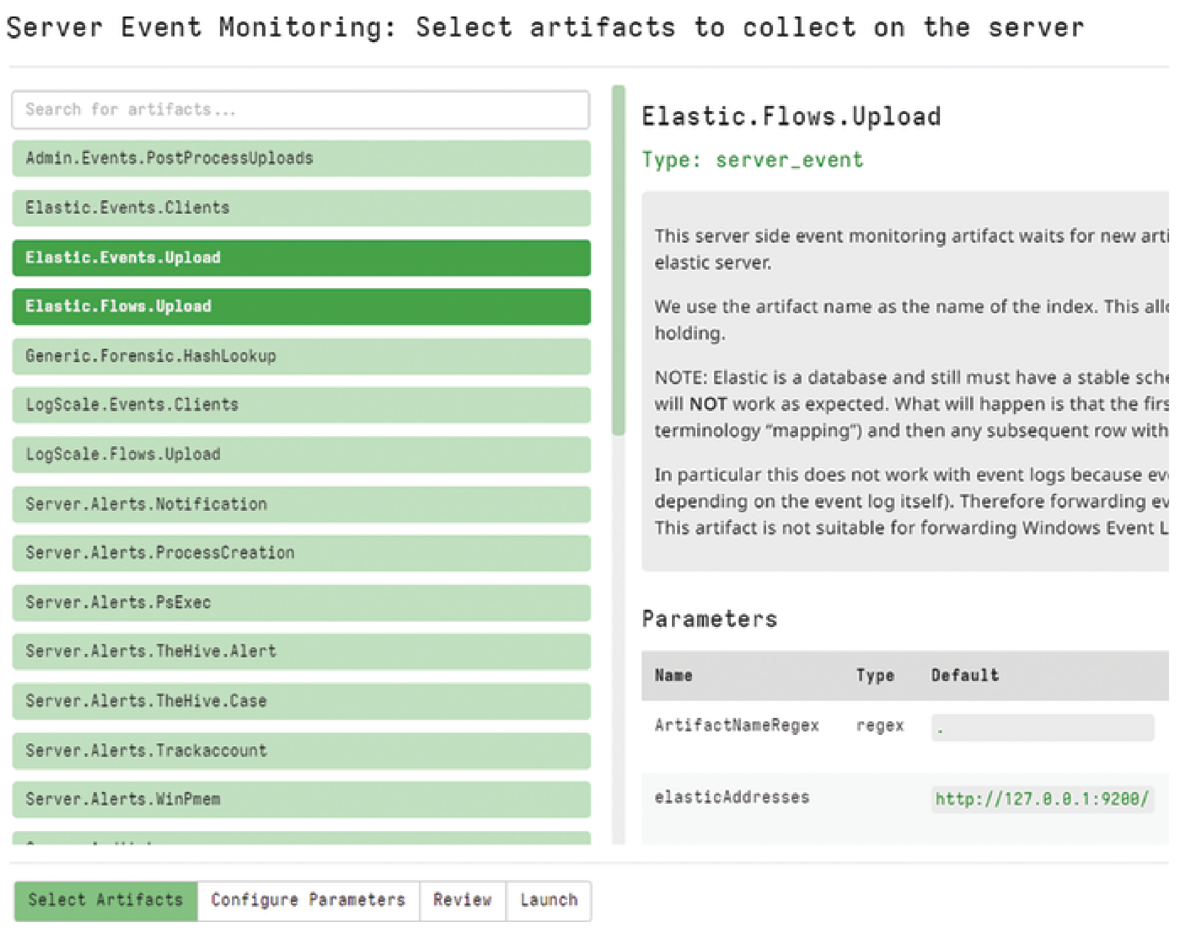


Figure 9.4 **Create server event.**

We now need to create two VQL artefacts called Elastic.Flows.Upload and Elastic.Events.Upload, as shown in Figure 9.5, just click on them both to add them to the list. These two artefacts will send Flows data to Elasticsearch.


Server Event Monitoring: Select artifacts to collect on the server

Search for artifacts...

- Admin.Events.PostProcessUploads
- Elastic.Events.Clients
- Elastic.Events.Upload
- Elastic.Flows.Upload
- Generic.Forensic.HashLookup
- LogScale.Events.Clients
- LogScale.Flows.Upload
- Server.Alerts.Notification
- Server.Alerts.ProcessCreation
- Server.Alerts.PsExec
- Server.Alerts.TheHive.Alert
- Server.Alerts.TheHive.Case
- Server.Alerts.Trackaccount
- Server.Alerts.WinPmem

Elastic.Flows.Upload

Type: server_event

This server side event monitoring artifact waits for new arti
elastic server.

We use the artifact name as the name of the index. This all
holding.

NOTE: Elastic is a database and still must have a stable sch
will **NOT** work as expected. What will happen is that the firs
terminology "mapping") and then any subsequent row with

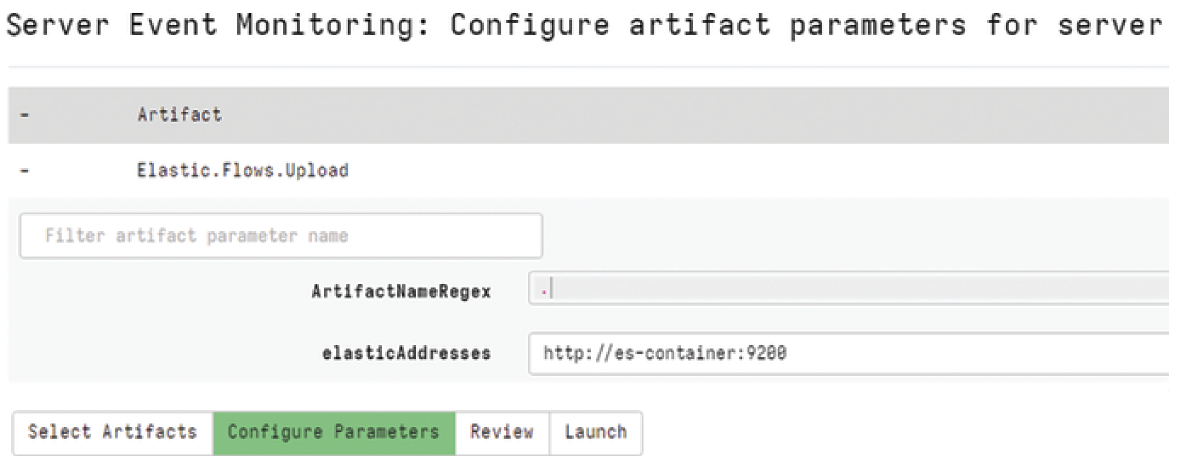
In particular this does not work with event logs because ev
depending on the event log itself). Therefore forwarding ev
This artifact is not suitable for forwarding Windows Event L

Parameters

Name	Type	Default
ArtifactNameRegex	regex	.
elasticAddresses		http://127.0.0.1:9200/

Figure 9.5 Server event monitoring.

Click on 'configure parameters', then click on Elastic.Flows.Upload. Here you need to change the 'ElasticAddresses' value to http://es-container:9200, as shown in Figure 9.6.


Server Event Monitoring: Configure artifact parameters for server

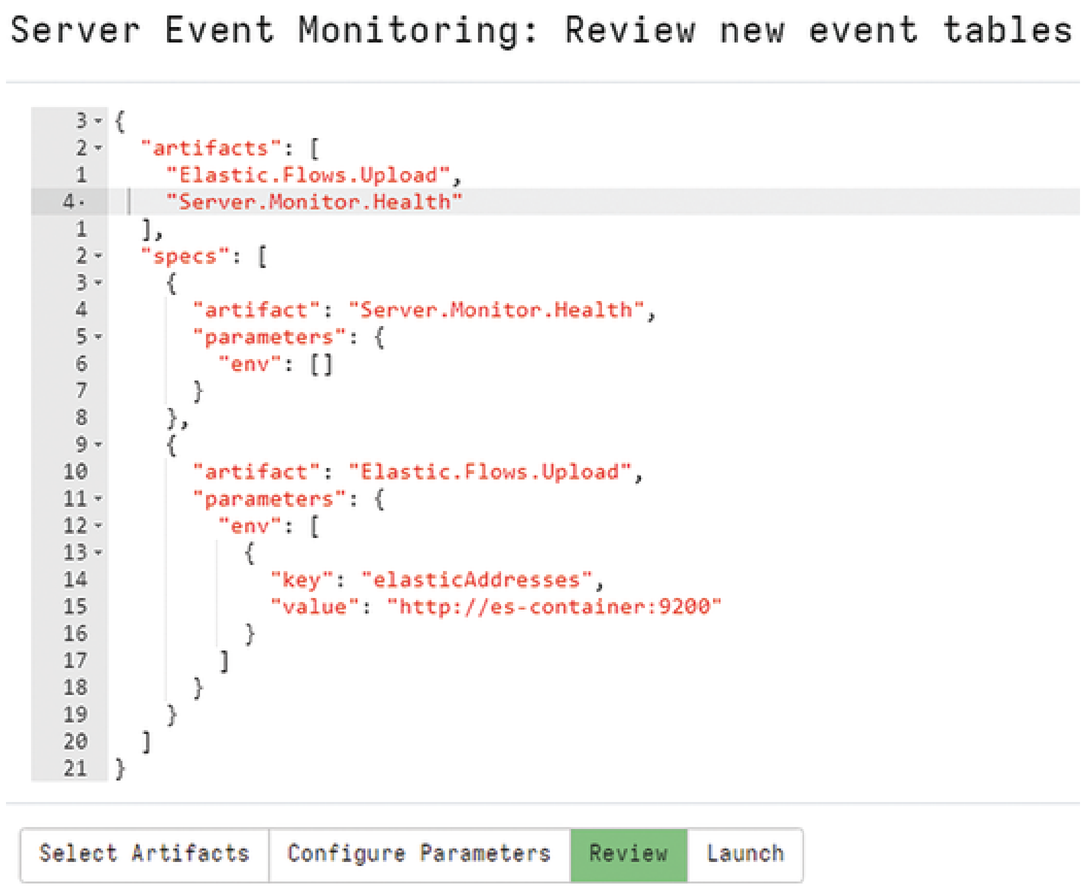
-	Artifact
-	Elastic.Flows.Upload

Filter artifact parameter name

ArtifactNameRegex	.
elasticAddresses	http://es-container:9200

Figure 9.6 Server event configuration.

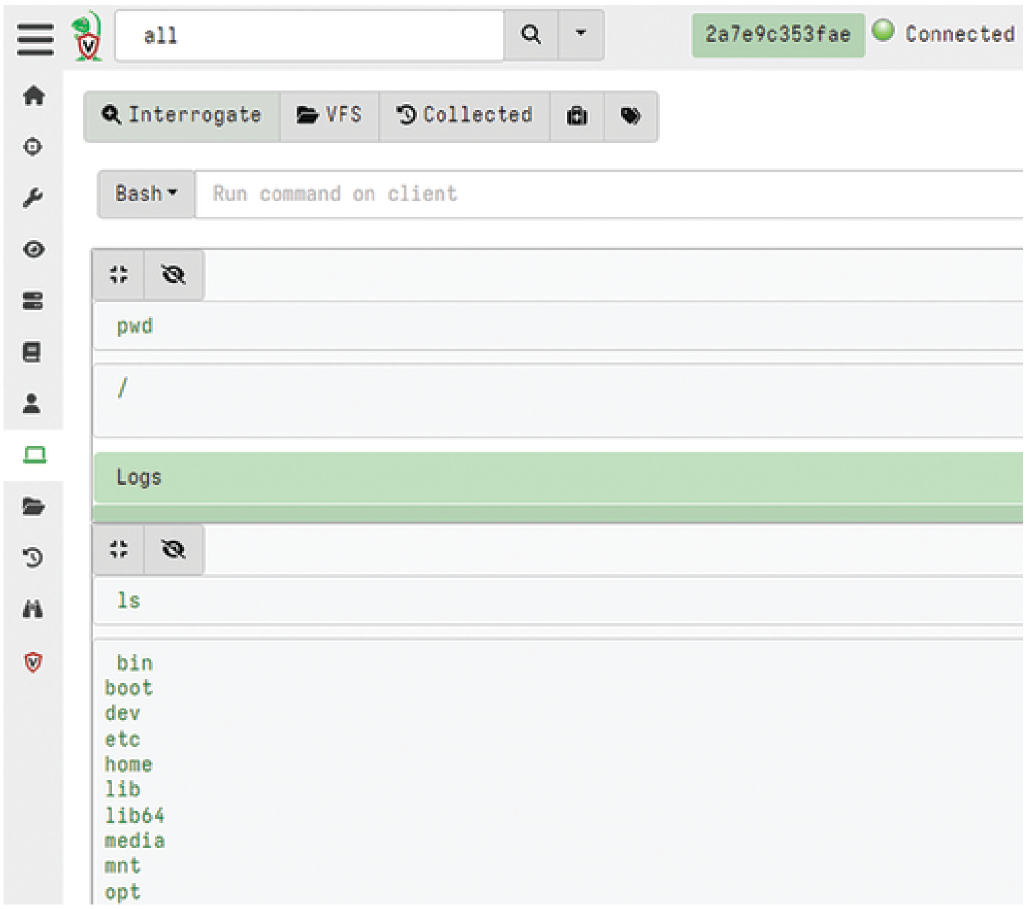
Press review and verify that the setup is correct. It should look something like in Figure 9.7.


Server Event Monitoring: Review new event tables

```
{
  "artifacts": [
    "Elastic.Flows.Upload",
    "Server.Monitor.Health"
  ],
  "specs": [
    {
      "artifact": "Server.Monitor.Health",
      "parameters": {
        "env": []
      }
    },
    {
      "artifact": "Elastic.Flows.Upload",
      "parameters": {
        "env": [
          {
            "key": "elasticAddresses",
            "value": "http://es-container:9200"
          }
        ]
      }
    }
  ]
}
```

Figure 9.7 Server event review.

Once you have verified the setup is correct, you can click on Launch. We are not quite done yet. We also need to have some data to send to Elasticsearch. One way to achieve this is by executing commands on a client using Velociraptor, like we did in the previous chapter. In the search bar, type ‘all’ and press Enter. From here, click on one of the clients. Then, in the upper right corner, click on Shell and execute a few commands, as shown in Figure 9.8.



```
pwd
/
```

```
ls
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
```

Figure 9.8 **Velociraptor client command.**

The data from Velociraptor should be sent to Elasticsearch. Let's head over to Kibana at localhost:5601 and see what data has been sent to Elasticsearch.

9.10 Analyse Logs with Kibana

This will not be an in-depth demonstration of how to use Kibana. We will just be going over the basics of creating a dashboard and performing some basic queries. If you are interested in a more detailed explanation about Kibana, please go to the Kibana documentation.[10] Start by opening your browser and navigating to the Kibana website http://localhost:5601. This should direct you to the Kibana webpage, as shown in Figure 9.9. From here, click on the menu button and then Discover.

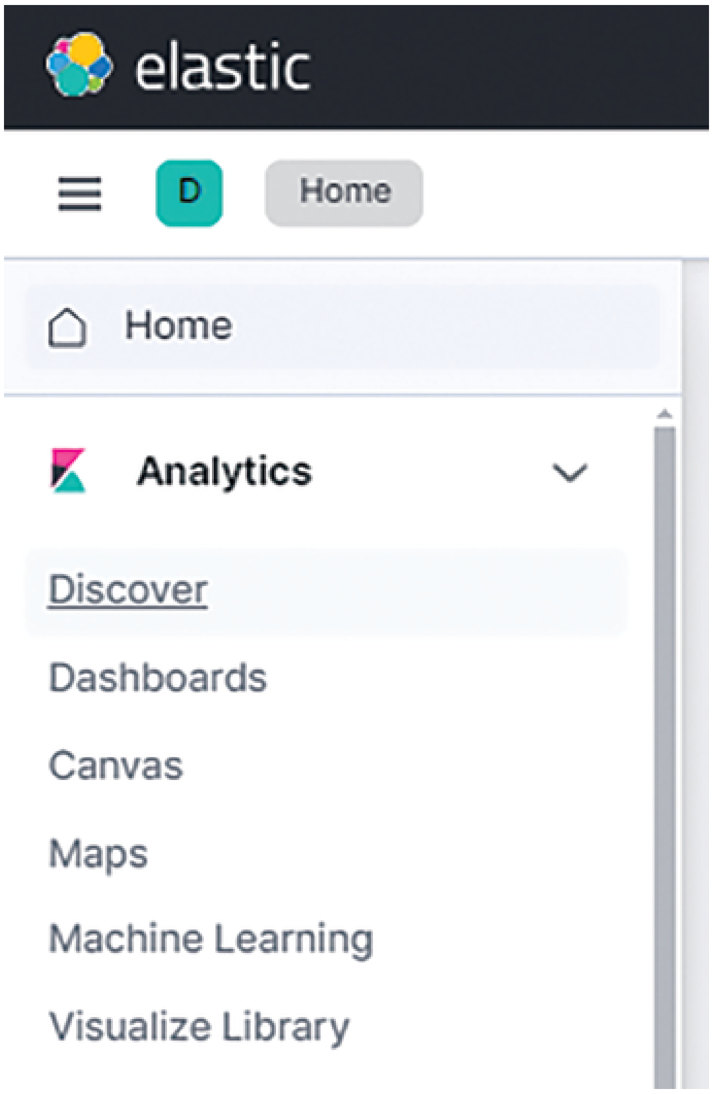
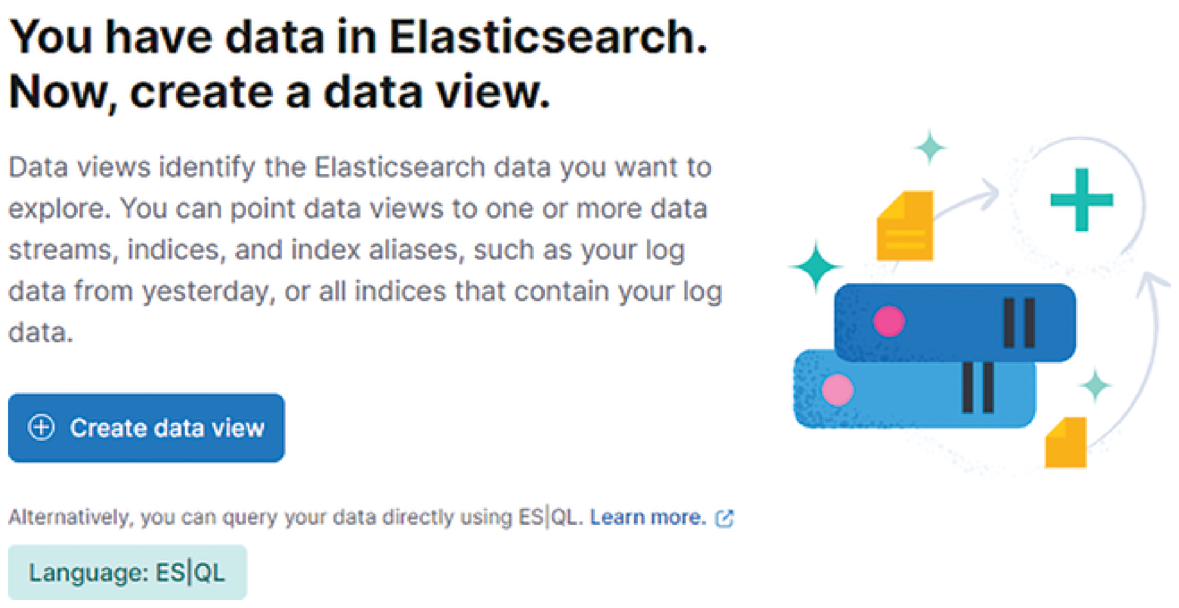


Figure 9.9 Kibana menu.

Here, it should require you to create a dashboard like in Figure 9.10.

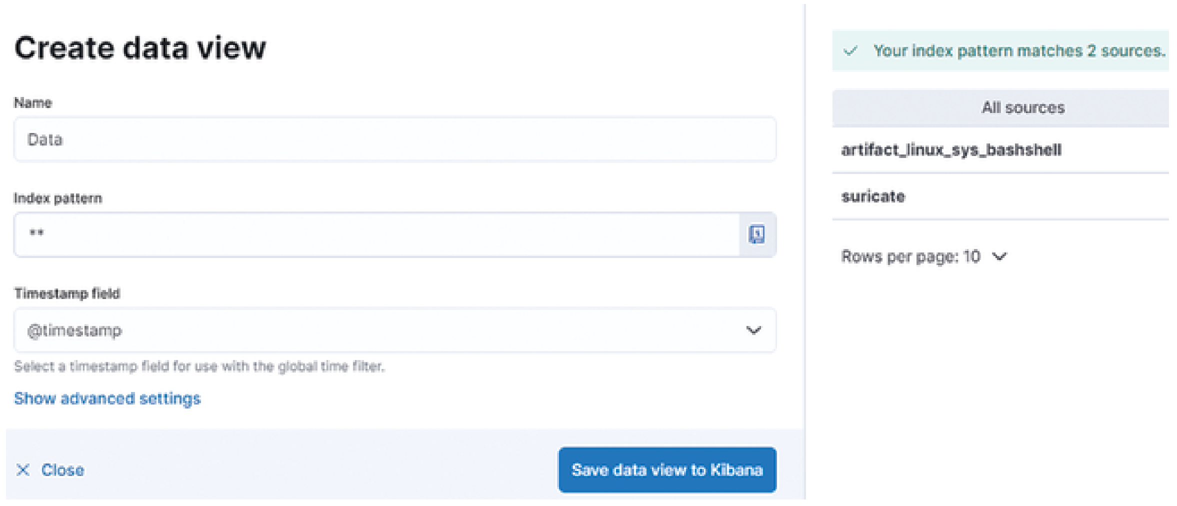

You have data in Elasticsearch. Now, create a data view.

Data views identify the Elasticsearch data you want to explore. You can point data views to one or more data streams, indices, and index aliases, such as your log data from yesterday, or all indices that contain your log data.

Create data view
Alternatively, you can query your data directly using ES|QL. Learn more.
Language: ES|QL

Figure 9.10 **Create data view.**

Click on the ‘Create data view’ button. This will start the process of creating a new data view. First, you must give it a name and then select which tables you want to include in the new view. For my case, I will be naming the view ‘data’. When it comes to selecting which table to include in the data view, it is by their name. Lucky for us, Kibana supports wildcards in the Index pattern type in ** and it should match to all the tables we have in Elasticsearch. In this scenario, we have two tables: Suricata and artifact_linux_sys_bashshell. At the end, it should look like what is shown in Figure 9.11.


Create data view

Name: Data

Index pattern: **

Timestamp field: @timestamp

Select a timestamp field for use with the global time filter.

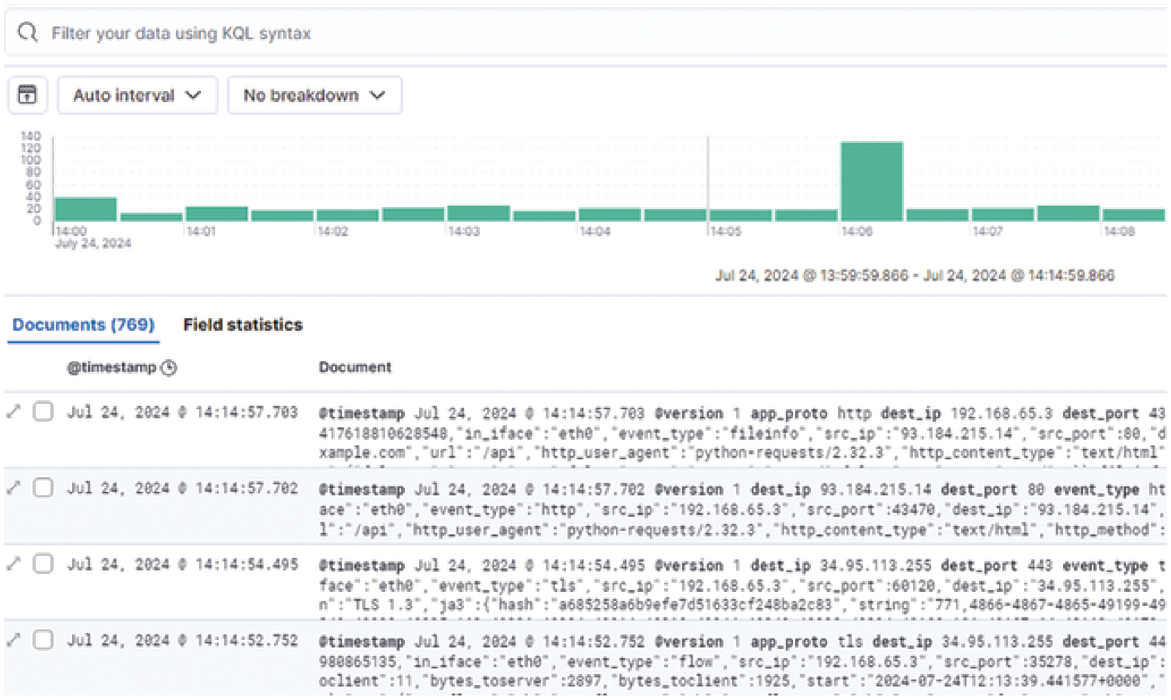
✓ Your index pattern matches 2 sources.

All sources
artifact_linux_sys_bashshell
suricate

Rows per page: 10

Figure 9.11 **Data view creation window.**

Once you click on Save, the system will present you with the newly created data view, similar to the one shown in Figure 9.12. On the left side, you should see what fields are available for searching. The data view appears in the centre. At the top, you have the search bar. Here you can search for data that exists in any of the tables, including the data view. For searching for data, Kibana uses 'Kibana query language' (KQL). You can read more about it on their site.[11] For our little demonstration, we will search for the commands you executed on the client using Velociraptor.



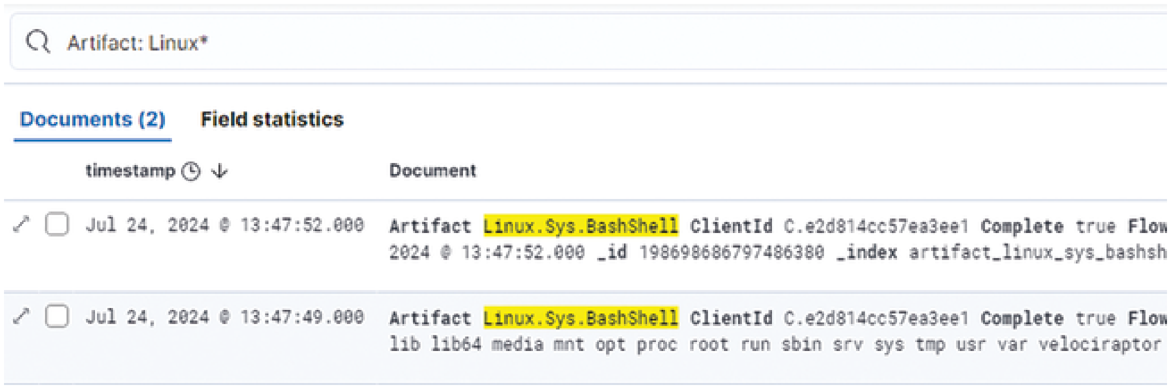
```
Jul 24, 2024 @ 14:14:57.703   @timestamp Jul 24, 2024 @ 14:14:57.703 @version 1 app_proto http dest_ip 192.168.65.3 dest_port 43
                              417618810628548,"in_iface":"eth0","event_type":"fileinfo","src_ip":"93.184.215.14","src_port":80,"d
                              xample.com","url":"/api","http_user_agent":"python-requests/2.32.3","http_content_type":"text/html"
Jul 24, 2024 @ 14:14:57.702   @timestamp Jul 24, 2024 @ 14:14:57.702 @version 1 dest_ip 93.184.215.14 dest_port 80 event_type ht
                              ace":"eth0","event_type":"http","src_ip":"192.168.65.3","src_port":43470,"dest_ip":"93.184.215.14",
                              l":"/api","http_user_agent":"python-requests/2.32.3","http_content_type":"text/html","http_method":
Jul 24, 2024 @ 14:14:54.495   @timestamp Jul 24, 2024 @ 14:14:54.495 @version 1 dest_ip 34.95.113.255 dest_port 443 event_type t
                              face":"eth0","event_type":"tls","src_ip":"192.168.65.3","src_port":60120,"dest_ip":"34.95.113.255",
                              n":"TLS 1.3","ja3":{"hash":"a685258a6b9efe7d51633cf248ba2c83","string":"771,4866-4867-4865-49199-49
Jul 24, 2024 @ 14:14:52.752   @timestamp Jul 24, 2024 @ 14:14:52.752 @version 1 app_proto tls dest_ip 34.95.113.255 dest_port 44
                              980865135,"in_iface":"eth0","event_type":"flow","src_ip":"192.168.65.3","src_port":35278,"dest_ip":
                              oclient":11,"bytes_toserver":2897,"bytes_toclient":1925,"start":"2024-07-24T12:13:39.441577+0000","
```

Figure 9.12 **Data view.**

To search for the data that came from Velociraptor, type in the following:

```
Artifact: Linux*
```

The results should look something like those in Figure 9.13, instead, it should return with the commands you ran inside Velociraptor.



```
Jul 24, 2024 @ 13:47:52.000  Artifact Linux.Sys.BashShell ClientId C.e2d814cc57ea3ee1 Complete true Flow
                             2024 @ 13:47:52.000 _id 198698686797486380 _index artifact_linux_sys_bashsh
Jul 24, 2024 @ 13:47:49.000  Artifact Linux.Sys.BashShell ClientId C.e2d814cc57ea3ee1 Complete true Flow
                             lib lib64 media mnt opt proc root run sbin srv sys tmp usr var velociraptor
```

Figure 9.13 Kibana search results.

It is also possible to search for Suricata logs. An example is a search for packages going to specific destinations using the search the following query in Kibana. Remember to change the IP address to the one that you want to search for.

```
dest_ip:{ip}
```

This is just a quick introduction to Kibana, if you are interested in more details, please refer to their documentation.[12]

9.11 Incident Response

There are two states in cybersecurity: either we are already breached or are going to be breached in the future. That means knowing how to respond when a cyber incident occurs is critical. We will use the NIST SP 800-61 framework,[13] as it is a well-defined framework used by many professionals. NIST has taken the incident response and broken it into six steps: preparation, detection and analysis, containment, eradication, recovery and post-incident activity. Shown in Figure 9.14.

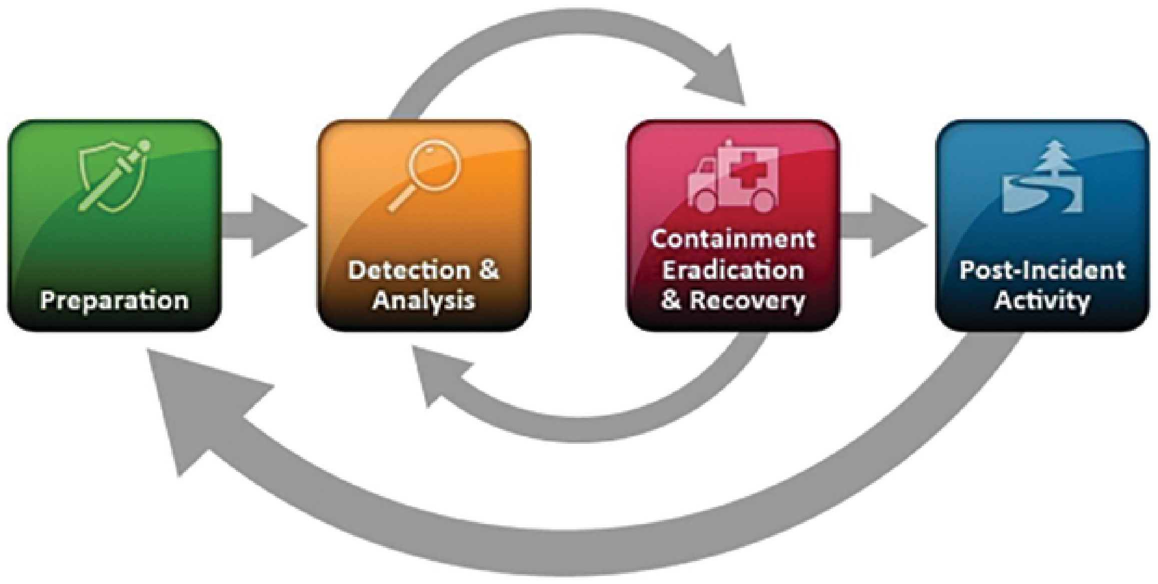


Figure 9.14 NIST SP 800-61 incident response lifecycle.

***Source:* P Cichonski et al. (2012)/NIST/Public domain.**

Let’s go over each of the steps in more detail.

1. Preparation is the first step in the incident response process. This step involves establishing the capability that allows one to respond to an incident. Much like how physical safety practices, fire drills, respond. We in cybersecurity also need to practise our responses in case of a cyber incident to ensure that we know what procedure to take, and if they are working as intended. It also allows us to make sure everyone knows what they are responsible for in the different stages of incident response. This step includes the following:
    - Developing an incident response policy and plan.
    - Establishing an incident response team.
    - Training staff and conducting regular incident response exercises.
    - Implementing necessary tools and technologies for incident detection and response.
2. Detection and Analysis: This step involves identifying and understanding any incident. The goal is to determine the response to incidents. Example of an incident: When a computer gets compromised, all trust in the computer is lost. There are two general

response options. Here we have to decide whether to remove the computer from the network, either by powering it off or isolating it, using the Endpoint Detection and Response. Another response, if you are on the bolder side, is to monitor the compromised computer to see what other devices it may be communicating with, allowing you to identify any additional compromised devices. The following processes are included in the step:

   - Continuous monitoring of network and system activities.
   - Analysing alerts from monitoring systems.
   - Identifying the scope and impact of the incident.

3. Containment is the process of limiting the impact of an incident. This includes isolating infected devices from the network. It is important to consider whether the containment action will impact the preservation of any evidence. An example of this is what would happen to the evidence if the system is taken down? These are just some of the considerations that have to keep in mind.
4. Eradication is the removal of the root cause of the incident. This should also include mitigating any vulnerabilities that the threat actors use to get access to the system, otherwise the threat actor has the possibility of returning to the system.
5. Recovery is the restoration of the system to operational functionality. The goal is to bring the affected system back to a controlled and secure state, ensuring that they are free from any compromise.
6. Post-Incident Activity: This stage focuses on learning from incidents, with the goal of improving both the response and detection capabilities of future incidents. It is important here that we do not blame anyone, as it will undermine the whole process and people will become too afraid to tell what really happened. The only goal is to understand what happened and what can be done in the future to reduce the probability of it happening again. This step should include documenting the incident, lessons learned, and implementing improvements in policies, procedures and technologies.

   Key activities in this phase include:

   - Conducting a detailed post-incident review meeting.

- Documenting the incident, response actions and lessons learned.
- Identifying gaps and weaknesses in the incident response process.
- Implementing improvements to enhance the organisation's incident response capability.

9.12 Detection Tuning

> It wasn't long after all the security tools was integrated into the SIEM, then one of the security analysts came up to you. "We have a big problem, we are getting far too many alerts a day to go through them all. And most of them are false positives. If we continue like this, we won't be able to identify if a real attacked have occurred." You take a moment to think "We have to invest more time tuning our SIEM to get a point that the SIEM is actually useful. You should see every false positive as a tuning opportunity."

We don't have a detection problem, but a response problem. The whole detection model is designed around providing as much detection as possible, logging everything, but it is up to someone else to figure out whether the alert matters. What gets in the way is the noise from all the false positives. The way we handle this is by tuning the alert sensitivity to a manageable level so that the Security Operations Center (SOC) team can handle it. I know this will be controversial, but in some cases it is better to miss true positives, if it means that the SOC team can deal with all the alerts being generated, otherwise you will end up with alert fatigue and alerts that are never responded to. The goal should eliminate 99% of all the false positives, then whatever is left is much easier to manage.

We know it takes several minutes for even the best-trained analysts to review an alert and decide how to respond. With the volume of attacks that are happening, we want two things to happen. The first is the reduction of the response time down to seconds. We know the attackers have automated their attacks, so the only way to keep up is to have a machine respond in real time. This is often done with playbooks. It allows us to create an automated response for any known bad things out there.

It’s the unknown malicious activity we need to manually respond to. You might be asking yourself ‘What happens if we accidentally shut down an important service with an automated rule?’. I completely understand your concern. What you are doing is make exceptions for business critical services, and in these cases the security analyst will have to manually respond to the alerts. It’s risky to be proactive, but it’s even riskier to be reactive.

9.13 Sigma

A question you should ask yourself is, if you were to switch SIEM solutions, then what about all the detection rules that you have created? Or another scenario could be an organisation want to share their detection rules with you, but is using a different SIEM. It would most likely not be possible to transfer the detection rules from one system to another, without having to translate them first. What about instead of writing detection rules in a system-specific language? We instead did everything in a universal language or schema that could be translated into any security tools-specific language. The good news for you is that it is already possible using a tool called Sigma.[14] It is an open-source project which provides a common language for writing detection rules. It allows security engineers to write detection rules in Sigma format and then convert them into a rule for a specific SIEM product. With Sigma, we have a shareable detection format, allowing you to take the rules you have written for your environment and hand them to someone and have Sigma translate the rules over to the system that they utilise like shown in Figure 9.15. The entire process is very simple and can be broken into two steps:

- Write a detection rule in Sigma format.
- Use sigma to convert the detection rule from sigma into a SIEM-specific language.

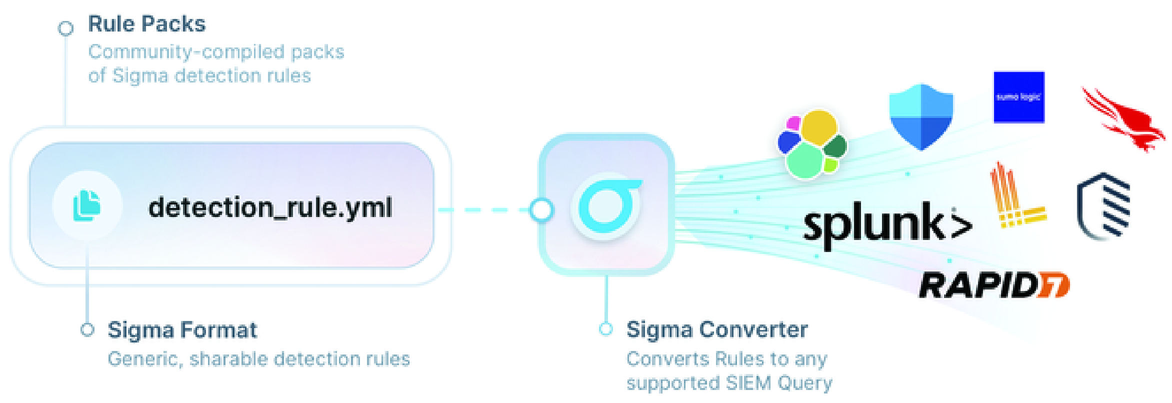

Rule Packs
Community-compiled packs of Sigma detection rules
detection_rule.yml
Sigma Format
Generic, sharable detection rules
Sigma Converter
Converts Rules to any supported SIEM Query
splunk>
RAPID7

Figure 9.15 Sigma flow.

***Source*: SigmaHQ by Alex.**

This requires that Sigma already has support for the target SIEM format, otherwise you have to write your own mapping script from Sigma to the target format. To allow for this process to happen, there are three components that make up the Sigma ecosystem.

- Sigma schema
- Sigma tool
- Sigma rule collection

Together, these components work to create a foundation for detection rules that support the creation, management and sharing of detection rules across different platforms and use cases.

These Sigma detection rules are written in Yet Another Markup Language format, which means they can be written using any text editor, and do not require any special tools to read or write. Sigma rules can be broken down into pieces.

1. Metadata: The first part of the rules is the metadata fields, which define the name and purpose of the rule.
2. Log source: Defines what log source that the detection rules is applied against.
3. Detection: Defines the search queries that need to be performed.
4. Condition: The conditions that are required for the rule to trigger.

Below is an example of a Sigma rule.

```
title: Authentication Occurring Outside Normal Business
Hours
id: 160f24f3-e6cc-496d-8a3d-f5d06e4ad526
status: test
description: Detects user signs-in outside of normal
business hours.

references:
    - https://docs.microsoft.com/en-us/azure/active-
directory/fundamentals/security-operations-user-
accounts#monitoring-for-failed-unusual-sign-ins
author: Mark Morowczynski '@markmorow', MikeDuddington,
'@dudders1'
date: 2022-08-11
modified: 2023-12-15
tags:
  - attack.persistence
  - attack.t1078
logsource:
  product: azure
  service: signinlogs
detection:
    selection:
      Status: Sucess
      # Countries you DO operate out of e,g GB, use
list for mulitple
      Location|expand: '%LegitCountries%'
      # outside normal working hours
      Date|expand: '%ClosingTime%'
    condition: selection
falsepositives:
  - User doing actual work outside of normal business
hours.
level: low
```

The rule above looks for user sign-ins outside of normal business hours on Azure. The first part of the rule is just metadata, or information about the rule itself. What we are interested in is the detection section. It is the most important component of any Sigma rule. It specifies exactly what the rule is looking for. Before we get into the detection section, you also have to understand the log source, which defines the data we are looking at. For the example above, we are looking at the sign-in logs from Azure.

```
logsource:
    product: azure
    service: signinlogs
```

For the detection part, it consists of multiple sections, each defines a filter that the rule is looking for.

```
detection:
  selection:
      Status: Success
      # Countries you DO operate out of e,g GB, use 
list for multiple
      Location|expand: '%LegitCountries%'
      # outside normal working hours
      Date|expand: '%ClosingTime%'
  condition: selection
```

The filter syntax is a little weird. Each line is an 'AND' operation. So, using the example above, this rule is looking of sign-in logs for a field called 'Status' that have the value of 'success', 'And' the location is equal to '%LegitCountries%'. By using the 'expand', it allows the author to embed the detection rule with logic, in this example it uses an environment variable to define the value. The final 'And' is that the date is outside of the normal working hours. When it comes to using 'or' as filters requires that filters are based on the same field in the log, e.g.,

```
field_name:
  - this
  - or
  - that
```

Sigma rules allow for very flexible detection. The detection section also allows you to define one or more selection groups, and it could look something like this.

```
detection:
  selection_connection:
      Initiated: 'True'
      DestinationPort: 3389
  selection_hosts:
      Computer|expand: '%domain_controller_hostnames%'
  filter_optional_defender_identity:
      Image|endswith: '\Microsoft.Tri.Sensor.exe' #
Microsoft
          Defender for Identity service makes port 3389
connections to hosts
  condition: all of selection_* and not 1 of filter_*
```

Each selection group defined must be present in the condition field. In our conditional example, all selection groups must be true and the filter must not be true. A more complex condition could look something like 'condition: (selection_one or selection_two) and not filter' here selection_one or selection_two must be true, and the condition of filter must not be true. This was a very fast overview of Sigma rules. There is still a lot to learn about how to create Sigma rules, and all the way to create detection rules. To learn more about how, check out the official documentation.[15] With the Sigma rules in place, we have the ability to write generic rules and translate them into something useful for our SIEM to use. This is where Sigma tools come into the picture. Let's go over how to set up Sigma and convert its rules into different SIEM formats. Sigma has several tools, and we will be focusing on sigma-cli, which can convert Sigma rules into SIEM queries that your environment can use. The easiest way to install sigma-cli is using Python package manager pip.

```
pip3 install sigma-cli
```

With sigma-cli installed, we need to add sigma-cli backend plugin that allows us to translate the rules into a specific query format. To see what plugins are available, use the command:

```
sigma plugin list
```

To install a plugin, use the ‘sigma plugin install <plugin_name>’ command. With sigma-cli installed, including the plugin, we can start translating sigma into queries.

```
sigma convert \
--target splunk \
--pipeline splunk_windows \
./rules
```

The ‘–target’ flag instructs Sigma to convert the Sigma files under the ./rules/ directory to specified SIEM format, to see the full list of supported SIEM, take a look at their documentation.[16] The ‘–pipeline’ flag tells Sigma to use a specific field- and source-mapping pipeline. Pipelines (or ‘processing pipelines’) should be seen as a subcategory of the target SIEM format. It provides a more nuanced and fine-tuned control over the way Sigma rules get converted into the SIEM-specific query format. The pipeline defines a sequence of transformations that are applied to a Sigma rule before it is converted into the target query language. These transformations can be field mappings, adding suffixes to field names or any of the others listed below. This ensures that the fields used within Sigma are mapped correctly to the fields and log sources used in SIEM. Each Sigma target provides pre-defined pipelines that Sigma command-line interface (CLI) makes available during conversion, to see the list of available pipelines for the plugins you have downloaded subcategory. To use the following command:

```
sigma list pipelines
```

For Elasticsearch, the pipelines that are available can be seen in Table 9.2.

TABLE 9.2

Elasticsearch pipelines for sigma.

Identifier	Priority	Process pipeline
ecs_windows	20	Elastic Common Schema (ECS) Windows log mappings from Winlogbeat from version 7
ecs_windows_old	20	ECS Windows log mappings from Winlogbeat up to version 6
ecs_zeek_beats	20	ECS for Zeek using filebeat 7.6.1
ecs_zeek_corelight	20	ECS mapping from Corelight
ecs_kubernetes	30	ECS Kubernetes audit log mappings

The last aspect that makes Sigma ecosystem so powerful is the sharing of knowledge among security professionals, to help the community detect threat actors. The largest collection of rules is from the Sigma repository,[17] hosted on GitHub. It contains numerous rules that can be downloaded and immediately converted and deployed into most SIEM solutions. All of this with just a few command lines, you have the knowledge of hundreds of security engineers ready to be injected into your SIEM with the goal of detecting to malicious activities.

9.14 Security Orchestration, Automation, and Response

This chapter would be incomplete without discussing security orchestration, automation, and response (SOAR), a crucial tool in modern cybersecurity operations. SOAR systems work by integrating and orchestrating the security stack for the organisation. This integration allows for a more holistic and efficient approach to security management. It allows the security team to automate responses to security alerts, streamline workflows and significantly improve the efficiency of security operations. For example, if an SOC analyst handles the same intrusion detection system alert a thousand times during their shift, the SOAR tool can facilitate the automation of that process. By implementing SOAR, it is possible for organisations to reduce the time and effort required to respond to incidents. This also enhances the overall effectiveness of their security posture, by standardising how they respond to incidents, regardless of who is on duty. To have a successful SOAR implementation, the following features should be included:

1. **Orchestration**: SOAR platforms integrate various security tools and systems, allowing them to work together seamlessly. This orchestration enables a coordinated response across multiple security layers.
2. **Automation**: By automating routine tasks and responses, SOAR reduces the workload on security analysts and minimises human error. This can include automating threat intelligence gathering, incident triage and initial response actions.
3. **Response**: SOAR facilitates rapid and consistent responses to security incidents. It can execute predefined playbooks or workflows based on the nature of the detected threat.

While SOAR offers significant benefits, its implementation comes with challenges:

- **Initial Setup Complexity**: Configuring SOAR to work effectively with existing systems can be complex and time-consuming.
- **Ongoing Maintenance**: SOAR playbooks and integrations need regular updates to remain effective against evolving threats.
- **Skill Requirements**: Effective use of SOAR requires skilled personnel who understand both security operations and automation technologies.
- **Balancing Automation and Human Oversight**: While automation is beneficial, it’s crucial to maintain human oversight for critical decisions and complex scenarios.

Another thing to be aware of when acquiring a SOAR: Many of the SOAR vendors require that you buy their entire application suite for the SOAR to be effective, as they cannot guarantee proper integration with any third-party tools.

9.15 Managed Security Service Provider

One of the first questions when it comes to building a SOC is whether it should be internal or external? Many organisations do not have or want the internal resources for monitoring the security systems. For this reason, many organisations choose to outsource this to a third-party Managed Security Service Provider (MSSP), especially for small to medium-sized organisations. It makes a lot of sense. That way, you do not have to worry about training and retaining people in a very competitive market of trained professionals. Depending on the MSSP, they can provide great value to mature the internal cybersecurity program. The advantages they have over an internal team are that they watch over multiple organisations, and there through gain more experience, in terms of how to detect and react to security incidents.

This is, of course, not without its downsides, as MSSPs work with clients in many industries, each using different security solutions. This means that they will not get a deep understanding of your organisation. The best way to mitigate this is through tailored playbooks, which tell the MSSP how to respond to security alerts detected within your organisation. Other organisations keep the incident response internal, for minor cyberattacks internal, so it is not the job of the MSSP to respond to the alerts, only to inform the organisation that an attack has happened. When it comes to selecting an MSSP, it is critical to understand the quality of their analyst. I always like to ask what kind of training they receive and how long an average analyst stays at the company. This allows me to get an idea of how they are treated. What you do not want is an untrained and burn-out personal watch over the security of your organisation, as they will most likely not be able to understand the alerts being generated. Remember, you are putting the security of the organisation into their hands.

9.16 Summary

Let’s review the key elements we have implemented in this chapter.

1. ELK Stack Setup: We have implemented the Elasticsearch, Logstash and Kibana (ELK) stack in our environment. Elasticsearch stores and

indexes all logs injected into the database. Kibana is used for visualising and analysing stored data. Logstash acts as the data pipeline, collecting logs from Suricata and sending them to Elasticsearch.

2. Logstash Configuration: We configured Logstash to collect logs from Suricata, parse them and forward the processed data to Elasticsearch.
3. Velociraptor-Elasticsearch Integration: We set up Velociraptor to send its logs directly to Elasticsearch, further centralising our log management.
4. Visualisation and Analysis: We utilised Kibana to create dashboards and visualisations of the collected data, with the goal of providing insights into the security of our environment.

Through these implementations, we have achieved centralised log management using a Security Information and Event Management (SIEM) solution built around Elasticsearch. Having an SIEM is critical for any modern cybersecurity strategy. It enhances visibility across the network, improves incident response efficiency and ensures compliance with regulatory requirements. By integrating logs from various sources into Elasticsearch, we can leverage powerful analysis and visualisation tools to effectively detect, investigate and mitigate security threats.

Notes

1 https://www.fluentd.org/

2 https://www.elastic.co/products/Logstash

3 https://regexlearn.com/

4 https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon

5 https://github.com/SwiftOnSecurity/sysmon-config/blob/master/sysmonconfig-export.xml

6 https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon\#configuration-files

7 https://attack.mitre.org/

8 https://mitre-attack.github.io/attack-navigator/

9 https://docs.velociraptor.app/blog/2019/2019-12-08-velociraptor-to-elasticsearch-3a9fc02c6568/

10 https://www.elastic.co/guide/en/kibana/current/index.html

11 https://www.elastic.co/guide/en/kibana/current/kuery-query.html

12 https://www.elastic.co/guide/en/kibana/current/index.html

13 https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-61r2.pdf

14 https://sigmahq.io/

15 https://sigmahq.io/docs/basics/rules.html

16 https://sigmahq.io/docs/digging-deeper/backends.html

17 https://github.com/SigmaHQ/sigma

Chapter 10
Vulnerability Management

In today’s information technology (IT) landscape, an organisation’s network comprises a multitude of systems and devices. Vulnerabilities can affect each of these systems, which attackers can exploit to gain access to the network. For this reason, having a vulnerability management process is crucial for safeguarding an organisation’s systems against vulnerabilities. A Zero Trust approach emphasises vulnerability management, which entails confirming that the systems and software functioning within the environment maintain a secure state. I want you to think of vulnerability management as a process for identifying and prioritising which systems need patching or reconfiguration to move them from an insecure to a secure state.

You might wonder why vulnerability management is necessary when a patch management system is already in place. The inability to automatically update some systems necessitates manual updates, delaying the process and increasing the potential for oversights. Not all vulnerabilities are just about patching; vulnerabilities often also arise from misconfiguration of the system. Vulnerability management is the process by which we identify these discrepancies or oversights in our environment. The process does this by identifying, evaluating, treating and reporting on vulnerabilities found within the environment. Giving a little more detail about the step, the process should look something like this:

1. Determine if any vulnerabilities expose the organisation. The vulnerability scanner should automatically do this.
2. Forecast the probability that a threat actor will leverage the vulnerabilities. For internet-facing systems, I like to set the probability to 1, as someone is definitely going to exploit it.
3. Determine what impact exploiting the vulnerability will have on the system.

4. Identify if there is a patch or other workaround that can help mitigate the vulnerability.
5. Implement the actions identified in the previous step.
6. Re-scan the system to verify the mitigation was successful.

Remember that not all vulnerabilities can have a permanent solution. In these cases, the implementation of temporary mitigations can help reduce the risk. This can include configuration changes, network segmentation and the application of compensating controls to protect against exploitation. The reason these changes are temporary is that they might change the nature of the system, like disabling needed features. This can create conflict, the security is responsible for the security of the environment, while the system owner is responsible for the availability of the system. That is why vulnerability management requires the involvement of the entire organisation. All vulnerabilities within an application should be the responsibility of the system owner, and with the guidance of the security team to identify viable treatment for the vulnerability. This also includes the prioritised remediation efforts and implementation of effective mitigations. By adopting a vulnerability management process, organisations can protect their assets, comply with regulatory requirements and enhance their overall security posture. Investing in vulnerability management is not just a recommendation; it is a critical necessity in today's threat landscape.

In this book, we will focus on the identification part of the process, which is the technical part of the process. The rest of the process is driven by organisational processes that are outside the scope of this book. To identify vulnerabilities, vulnerability scanners are used. These are specialised tools, such as Nessus, Qualys or OpenVAS. These tools scan the environment for any known vulnerabilities and misconfigurations. These tools compare the configurations and versions of the software against a database of known vulnerabilities, like the CVE database.[1] Any vulnerability management tool of any worth its price will also provide detailed information on how to mitigate any vulnerabilities that are identified.

10.1 Vulnerability Scanner

Vulnerability scanners are essential tools in modern security programs. They identify, assess and report security vulnerabilities across systems, applications and networks. They play a critical role in modern cybersecurity by proactively detecting weaknesses that attackers could exploit. These scanners use a vulnerability database, such as the Common Vulnerabilities and Exposures (CVEs) system, which serves as a centralised repository for known vulnerabilities. MITRE's Steve Christey and David Mann originally developed the CVE database as a community-driven solution for tracking vulnerabilities. Their goal was to create a universally accessible catalogue that the entire security community could use to refer to specific vulnerabilities consistently. Over time, the CVE system has become the de facto standard for communicating about vulnerabilities across security tools, reports and advisory systems, enabling organisations to stay informed and mitigate risks efficiently. This works together with standardised metrics like the Common Vulnerability Scoring System (CVSS), where experts assess each vulnerability for its potential impact and likelihood of exploitation.

Vulnerability scanners rely on this vulnerability database to identify issues in real-time, comparing system configurations and software versions against known vulnerabilities in the CVE database. The CVSS assigned to each vulnerability helps security teams prioritise remediation efforts by understanding the potential risk and the severity of each flaw. There are two main types of vulnerability scanning approaches: agent-based and agentless scanning, each having its own set of advantages and disadvantages.

10.1.1 Agent-based Scanning

Agent-based scanning involves the installation of a software agent on each target device, with the purpose of scanning for vulnerabilities. This approach provides several key benefits. One of the main advantages is the insights it offers by having an agent directly on the device, providing deeper visibility into the system, including real-time monitoring and configuration assessments. This is particularly useful for identifying vulnerabilities that might not be visible through network scanning alone. Agent-based scanning is effective for scanning remote devices that are not always connected to the network, such as laptops used by remote workers.

However, there are also disadvantages to this approach. The primary challenge is the deployment effort required. Installing and maintaining agents on all target devices can be labour-intensive, especially in large environments with numerous devices. Extensive planning is necessary to ensure all devices are included, and the process can be time-consuming. Agents can consume system resources, potentially affecting the performance of the target devices. This resource consumption needs to be managed to avoid disrupting normal operations.

- Advantages:
  - Detailed Insights: Provides deep visibility into the system, including real-time monitoring and configuration assessments.
  - Remote Devices: Effective for scanning devices that are not always connected to the network (e.g. laptops).
- Disadvantages:
  - Deployment Effort: Requires installation and maintenance of agents on all target devices.
  - Resource Consumption: Agents can consume system resources and potentially impact performance.

10.1.2 Agentless Scanning

Agentless scanning scans devices over the network without requiring software installation on the target devices. This approach is simpler to implement because it does not involve installing or managing agents. The ease of deployment is a significant advantage, making it quicker to set up and reducing administrative overhead. Another benefit of agentless scanning is that it is non-intrusive, as it consumes fewer resources on the target devices. This means the scanning process doesn't impact target device performance.

However, agentless scanning has its own limitation, which is the limited visibility it provides compared to agent-based scanning. Without an agent on the device, the scanner may not gather necessary information, potentially missing vulnerabilities. Agentless scanning depends heavily on reliable network connectivity. Network problems can cause incomplete or inaccurate scan results.

- Advantages:
  - Ease of Deployment: No need to install or manage agents, making it simpler to implement.
  - Non-intrusive: Does not consume resources on the target devices.
- Disadvantages:
  - Limited Visibility: May not provide as detailed information as agent-based scanning.
  - Network Dependency: Requires access to the entire network to scan every device, making the scanner very attractive as you will get access to the entire network.

Both agent-based and agentless scanning methods have their place in a comprehensive vulnerability management strategy. The choice between them depends on the specific needs and constraints of the environment being secured. By understanding the strengths and weaknesses of each approach, organisations can make informed decisions to protect their systems effectively.

10.2 Zero Trust Vulnerability Management

Vulnerability management is part of the monitoring principle of Zero Trust. We should ideally architect, create, and configure systems securely from the start. However, this is rarely the reality in practice. We live in a world where security is not a top priority. I really like the phrase ‘Safety third’, it reminds me that security is not and never will be a top priority for the organisation. Getting things into a working state is always more important for the organisation. Even if the system is currently secure, people discover new exploits daily. A new vulnerability can change a secure system into an insecure one overnight. And that is what vulnerability management is about: detecting the current state of the systems on the network. It removes the trust we have in our environment that everything is secure and will stay secure.

For the sharp reader, you can quickly see that for the vulnerability scanner to detect and analyse the current state of the environment, it requires it to have access to every system on the network. The question quickly changes to: what about the trust we put in the vulnerability scanner, a system that requires access to every system for it to be effective? It is not a surprise that the vulnerability scanner is often the target of threat actors, as it acts both as a great pivot point to the network and the potential of stored credentials in the system. Let's add the vulnerability scanner to the network diagram to show how it is connected to the network.

In the network diagram shown in Figure 10.1, we can see that it has a connection to each zone or segment. To get another perspective, let's also add it to the transaction map to see what system it interacts with. Table 10.1 shows the transaction map. Notice that the vulnerability scanner also needs internet access to stay updated with the latest vulnerabilities.

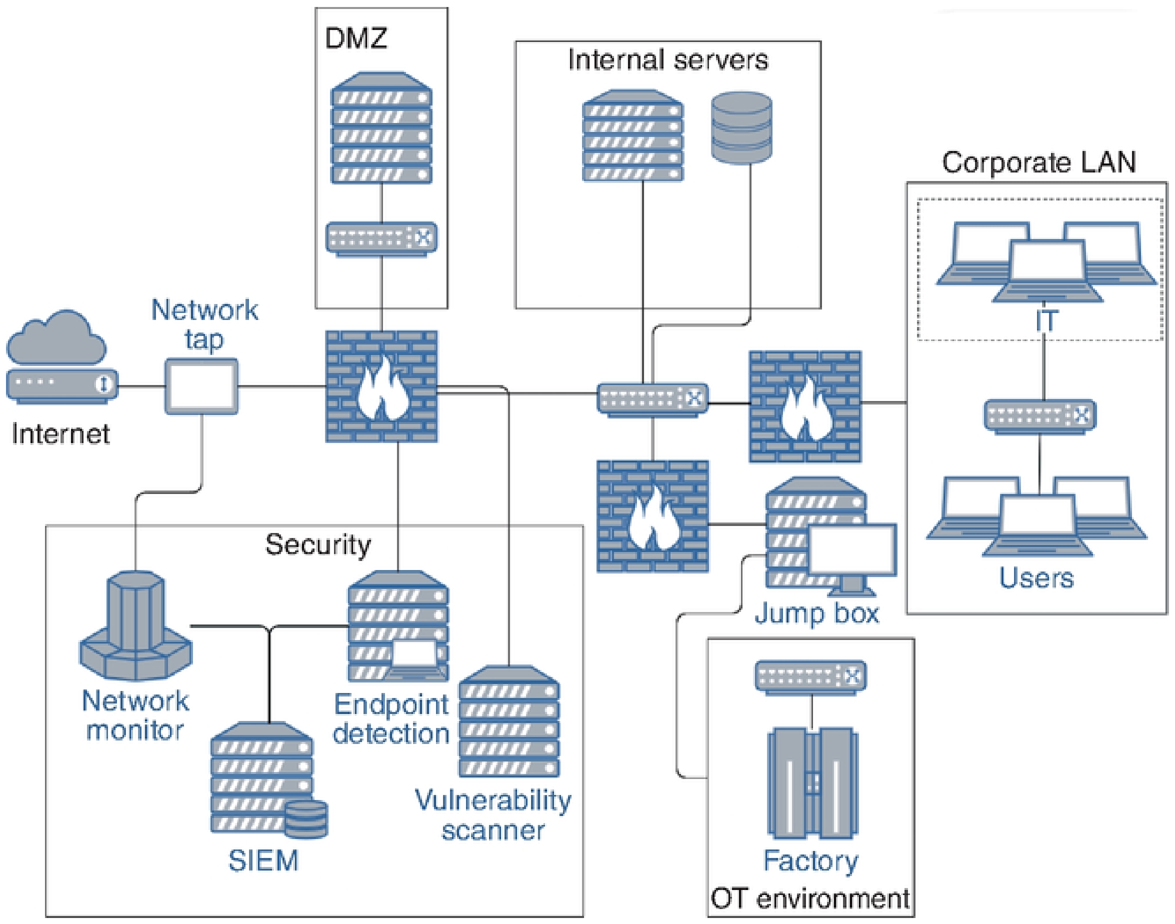
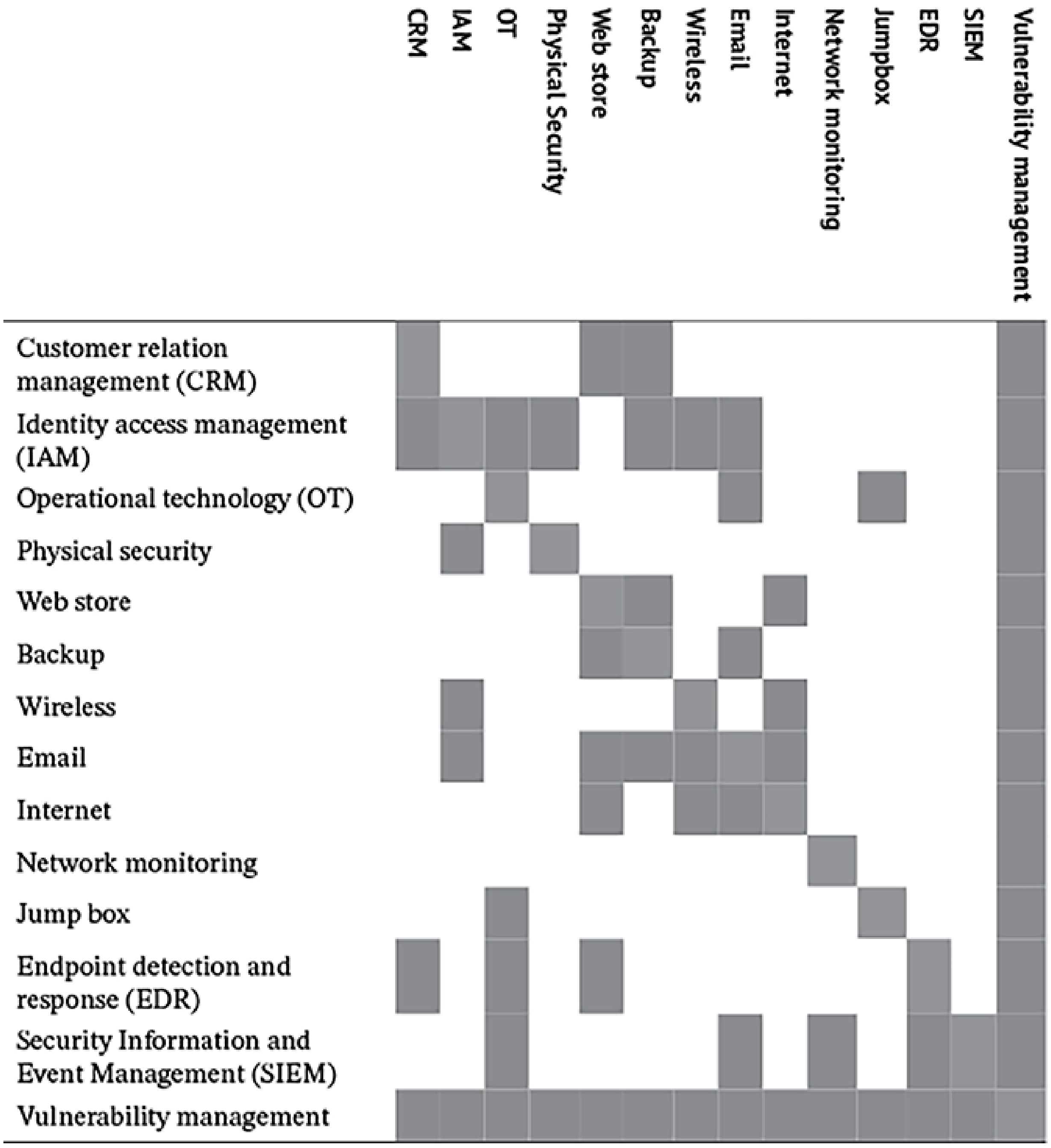


Figure 10.1 **Network diagram.**

TABLE 10.1

Transaction map.



	CRM	IAM	OT	Physical Security	Web store	Backup	Wireless	Email	Internet	Network monitoring	Jumpbox	EDR	SIEM	Vulnerability management
Customer relation management (CRM)	■				■	■								■
Identity access management (IAM)	■	■	■	■		■	■	■						■
Operational technology (OT)			■					■			■			■
Physical security		■		■										■
Web store					■	■			■					■
Backup					■	■		■						■
Wireless		■					■		■					■
Email		■			■	■	■	■	■					■
Internet					■		■	■	■					■
Network monitoring										■				■
Jump box			■								■			■
Endpoint detection and response (EDR)	■		■		■							■		■
Security Information and Event Management (SIEM)			■					■		■		■	■	■
Vulnerability management	■	■	■	■	■	■	■	■	■	■	■	■	■	■

What then can be done to remove the trust we have in the vulnerability scanner? With a system that needs access to every system for it to function. The thing is that it is not always scanning, so should it always have access? What you can do is limit the network access to only when the system is scanning. Vulnerability scanners often have administrative or high-level credentials stored within them to perform scans across network. Attackers can extract these credentials and use them to access other systems within the network, facilitating lateral movement. Remember, just because they are security tools, it does not mean they are secure themselves. The worst that can happen for a security team is that the tools they use for security are used to compromise the environment. For identifying new areas of removing trust, try viewing them from different perspectives with an adversary mindset. This allows you to see how they can use the system to their advantage. Once you have identified these holes, the next step is implementing controls to mitigate them.

10.3 Deployment of Nessus Within Our Docker Environment

For our project, we will use Nessus in an agentless configuration. Nessus is a popular vulnerability scanner developed by Tenable. It is one of the key players in the vulnerability scanner market, used by many organisations around the world. Nessus is a paid product. Luckily for us, they also provide a free version of Nessus. This free version provides the essential features for performing vulnerability scanning, making it an excellent choice for our project. Deploying Nessus using Docker is straightforward. All we have to do is add the Nessus images to our Docker Compose file:

```
nessus:
  image: tenable/nessus:latest-ubuntu
  container_name: nessus
  restart: always
  ports:
    - 8834:8834
  networks:
    - corporation
    - Production
    - dmz
```

With the Nessus container added to the Docker compose and configured to have access to all the networks. This ensures that it can scan all the containers that we have in our environment. The first time you start up Nessus, it will start the initialisation process. This will take some time (approximately 30 minutes), so some patience is required.

We are going to use Nessus in an agentless configuration, leverage the ease of deployment and non-intrusive nature of agentless scanning, which is very suitable for our needs to perform a quick scan of the entire environment. This way, we can identify if there are any known vulnerabilities that exist in our environment. Using Nessus in this configuration, we can efficiently monitor our Docker network without the overhead of installing and managing agents in each container. Once Nessus has started, it is time to begin the setup process. The first step is to head over to localhost:8834. There you should see that Nessus is in the process of initialisation, as displayed in Figure 10.2. This will take a few minutes to complete.

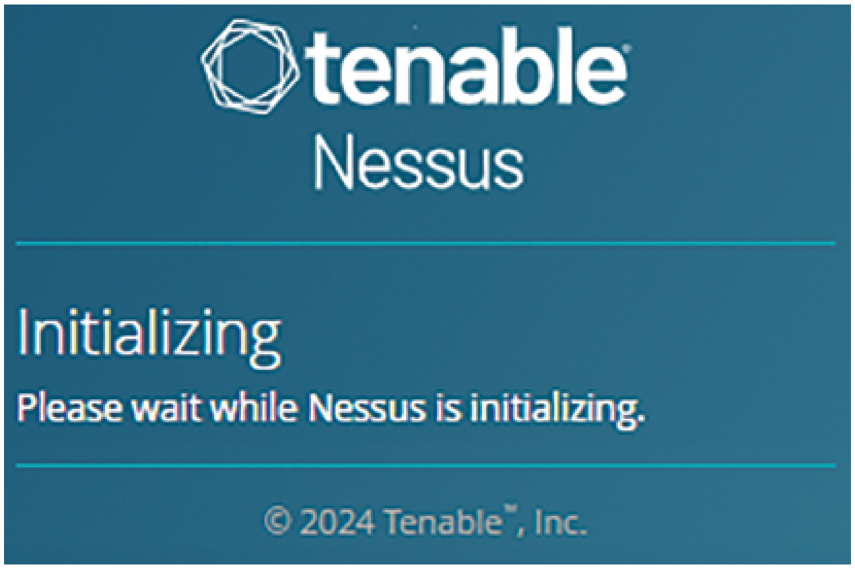
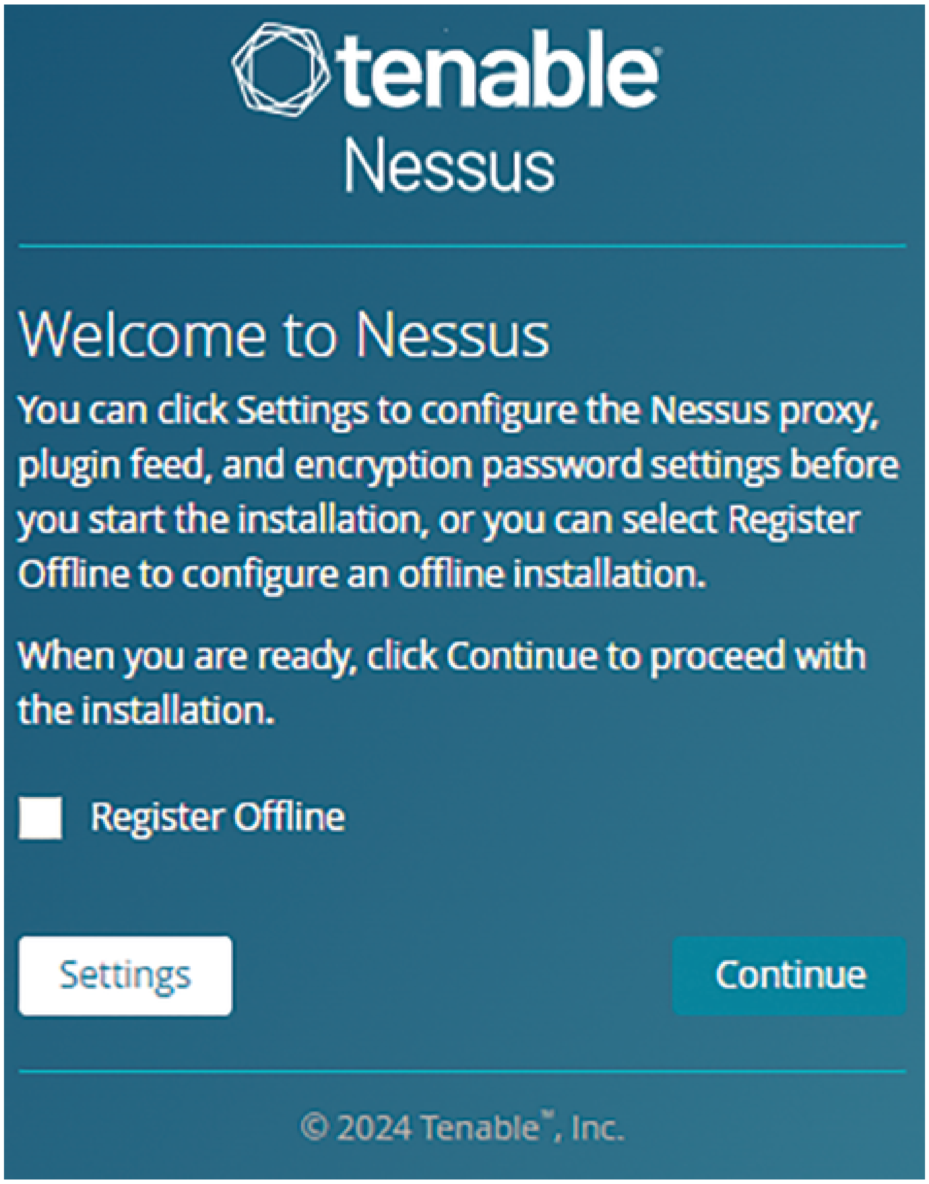


Figure 10.2 Initiation.

Once the initialisation is complete, Nessus will ask you to register, as shown in Figure 10.3.

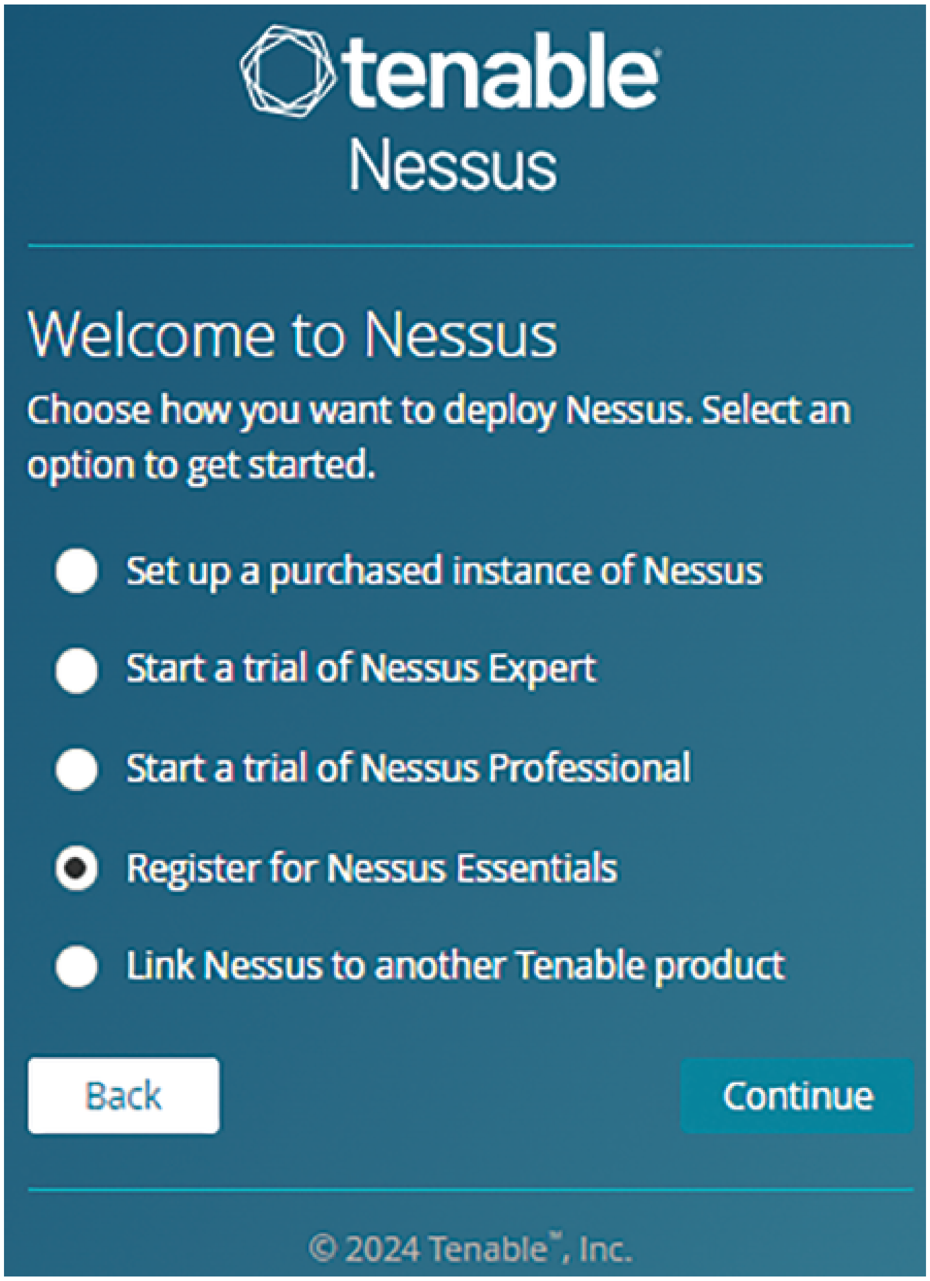

Welcome to Nessus

You can click Settings to configure the Nessus proxy, plugin feed, and encryption password settings before you start the installation, or you can select Register Offline to configure an offline installation.

When you are ready, click Continue to proceed with the installation.

Figure 10.3 Nessus registration.

Just like in Figure 10.4, select the Nessus essential product.


Nessus

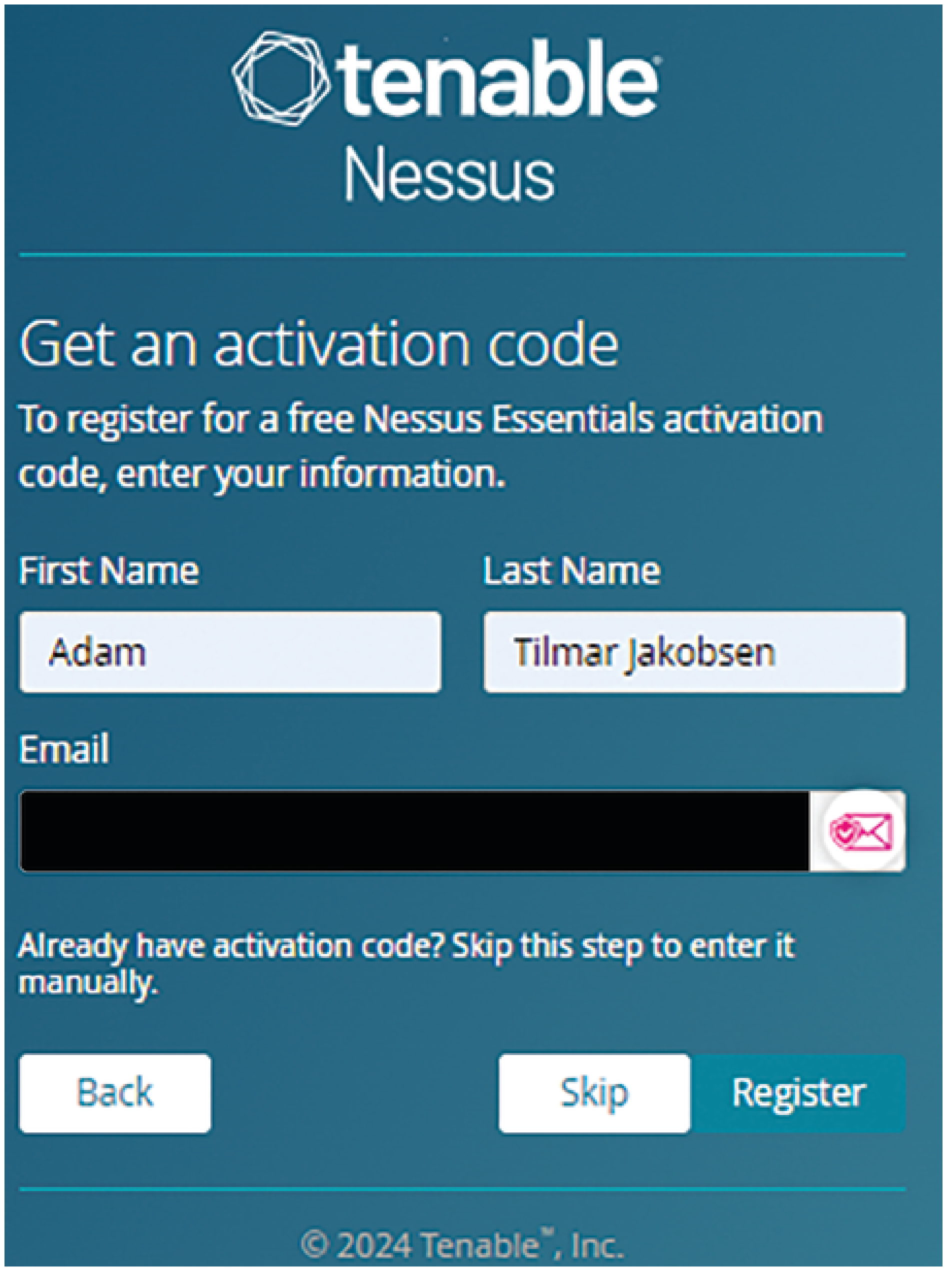
Welcome to Nessus

Choose how you want to deploy Nessus. Select an option to get started.

- ( ) Set up a purchased instance of Nessus
- ( ) Start a trial of Nessus Expert
- ( ) Start a trial of Nessus Professional
- [✓ Register for Nessus Essentials]
- ( ) Link Nessus to another Tenable product

Figure 10.4 Nessus product.

Similar to Figure 10.5, fill out your information and click next.


tenable
Nessus

Get an activation code

To register for a free Nessus Essentials activation code, enter your information.

First Name: Adam

Last Name: Tilmar Jakobsen

Email: [redacted]

Already have activation code? Skip this step to enter it manually.

© 2024 Tenable™, Inc.

Figure 10.5 Nessus get activation code.

You should now have activated your account, and be seeing a message as displayed by the registration code, as shown in Figure 10.6.

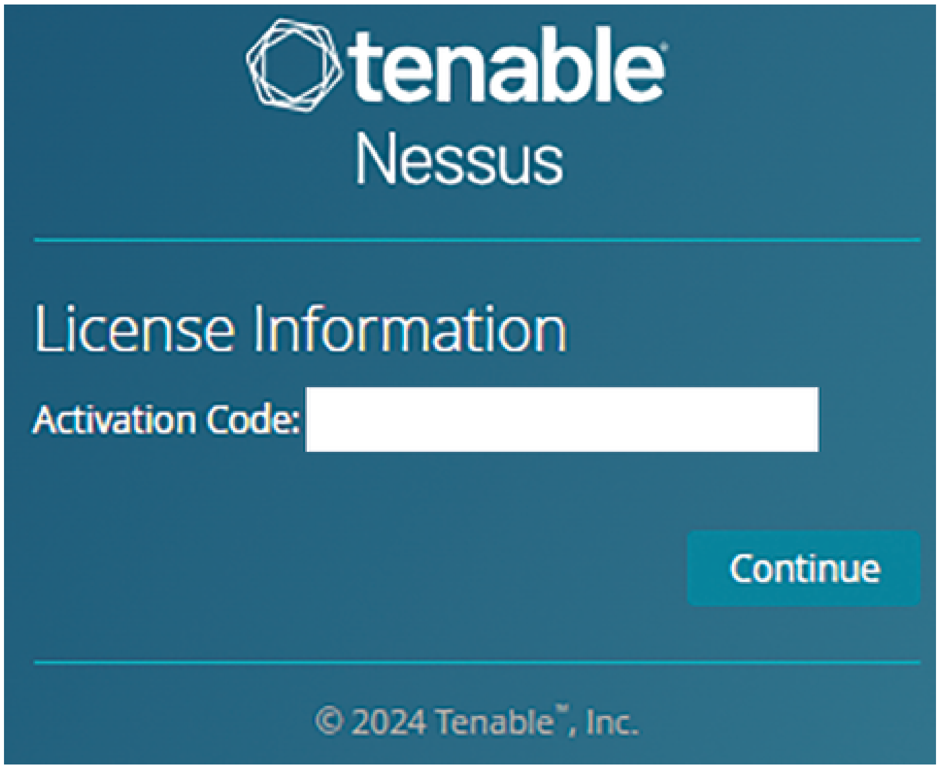
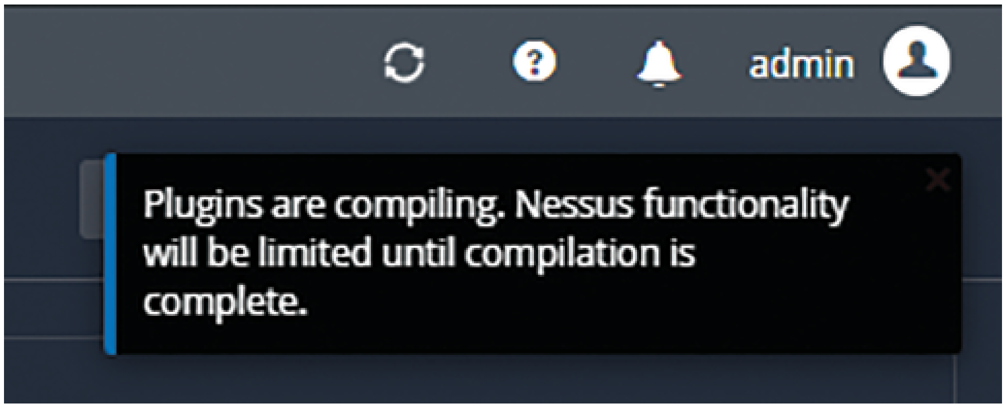


Figure 10.6 Nessus activation code.

Once you have completed the registration process, it will start downloading the plugins. In the upper right corner, you should see a spinning icon; when you hover over it, it informs you that it is downloading. Figure 10.7 shows this.



Plugins are compiling. Nessus functionality will be limited until compilation is complete.

Figure 10.7 Nessus download plugins.

Once complete, we should now be able to perform a vulnerability scan of our network. Before you can scan anything, you need to identify the IP range of the network you want to scan. For this demonstration we will scan the corporation network, as we know there are vulnerable systems there. You can find the corporate network IP range by using the Docker network command.

```
docker network inspect corporation
```

The output of the command should look something like below, just with a different IP range.

```
PS C:\> docker network inspect corporation
"Name": "corporation",
"Id": "2c2493...",
"Created": "2024-07-20T14:02:45.542276217Z",
"Scope": "local",
"Driver": "bridge",
"EnableIPv6": false,
"IPAM": {
    "Driver": "default",
    "Options": null,
    "Config": [
        {
          "Subnet": "172.21.0.0/16",
          "Gateway": "172.21.0.1"
        }
    ]
},
```

Now that we have identified the IP range, we would like to scan. Head back over to Nessus at localhost:8834 and click on the scan button, as shown in Figure 10.8.

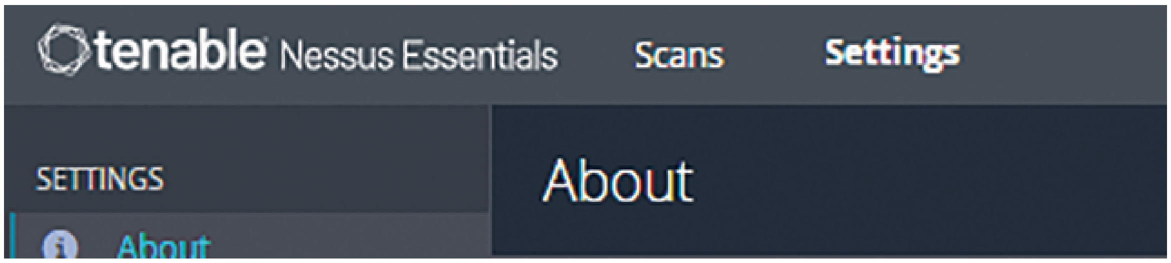


Figure 10.8 Nessus scan menu.

Click on ‘new scan’ as shown in Figure 10.9.

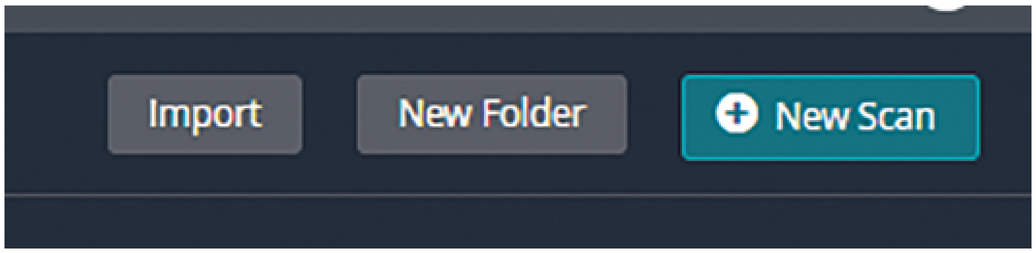


Figure 10.9 **Nessus new scan button.**

We want to perform a network scan, so click on the 'basic network' scan, shown in Figure 10.10.

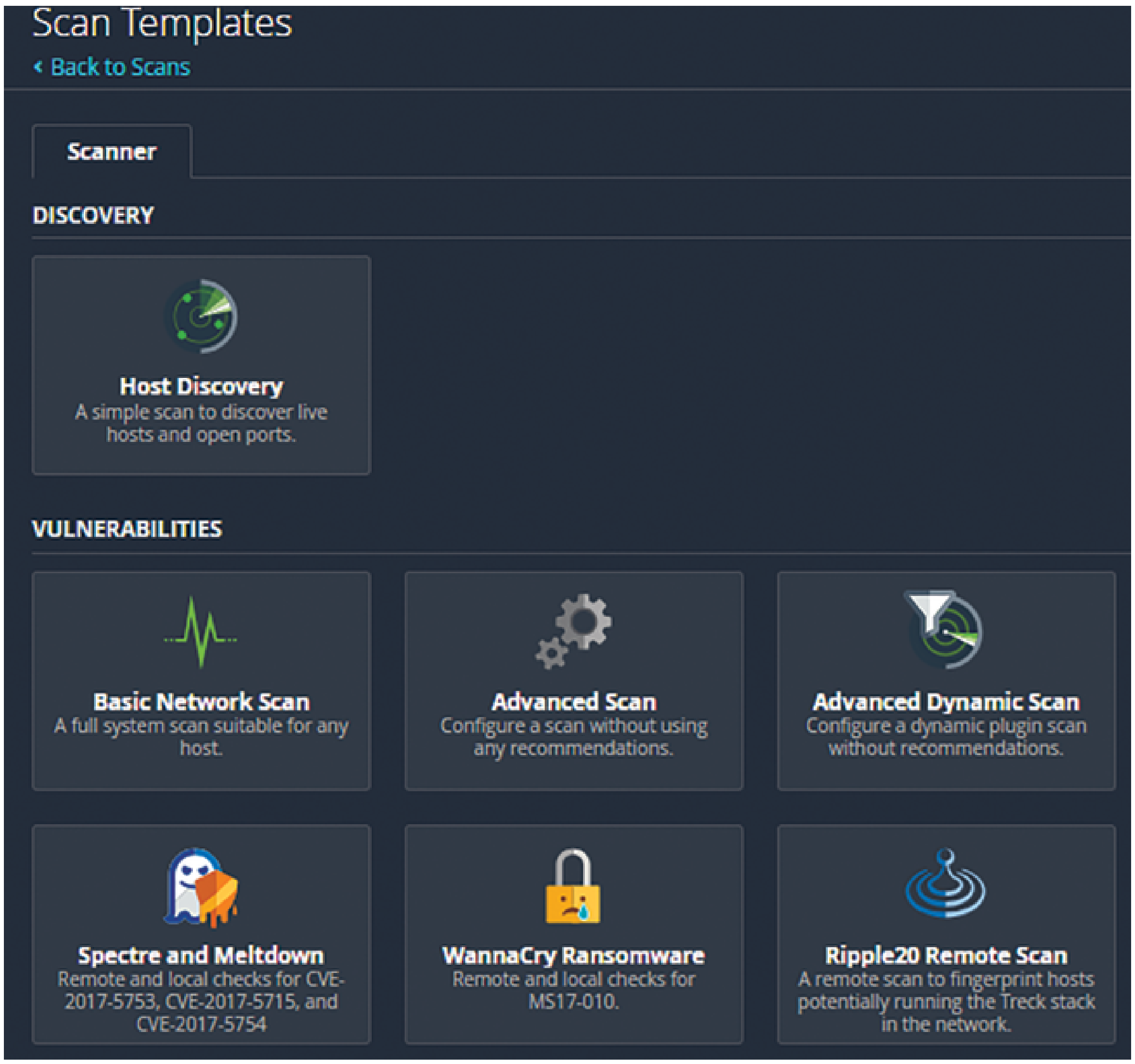
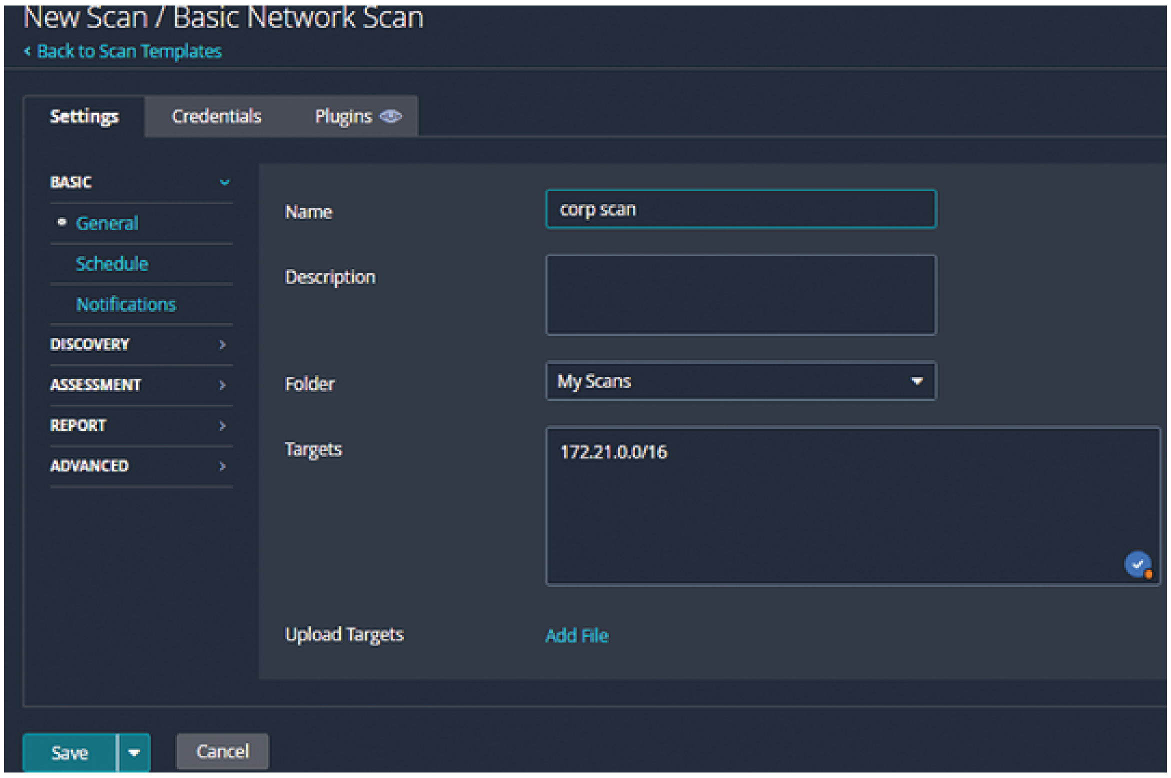

Scan Templates
‹ Back to Scans
Scanner
DISCOVERY
Host Discovery
A simple scan to discover live hosts and open ports.
VULNERABILITIES
Basic Network Scan
A full system scan suitable for any host.
Advanced Scan
Configure a scan without using any recommendations.
Advanced Dynamic Scan
Configure a dynamic plugin scan without recommendations.
Spectre and Meltdown
Remote and local checks for CVE-2017-5753, CVE-2017-5715, and CVE-2017-5754
WannaCry Ransomware
Remote and local checks for MS17-010.
Ripple20 Remote Scan
A remote scan to fingerprint hosts potentially running the Treck stack in the network.

Figure 10.10 **Basic scan.**

It should now ask you to fill out the subnet you would like to scan. We will use the one identified above, as shown in Figure 10.11.


New Scan / Basic Network Scan

Name: corp scan

Description:

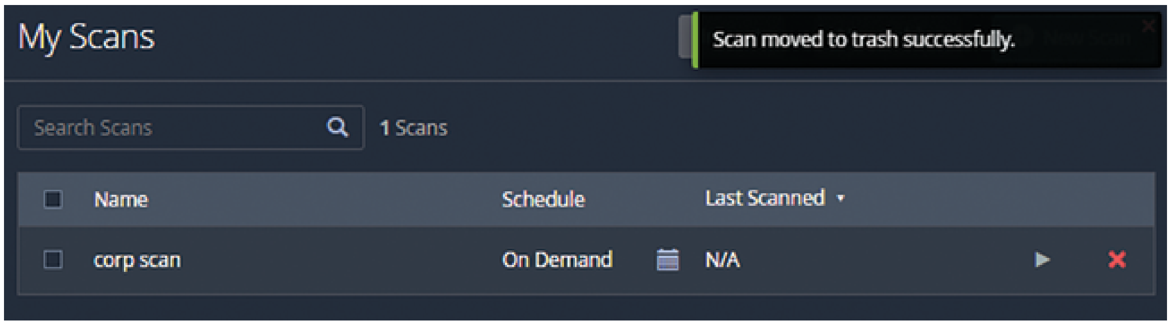
Folder: My Scans

Targets: 172.21.0.0/16

Upload Targets: Add File

Figure 10.11 **Create new scan.**

Click on save, and it should now have saved the scan, and bring you the scan overview, similar to Figure 10.12. From here, press the play button on the scan we have just created to start the scanning process.


My Scans

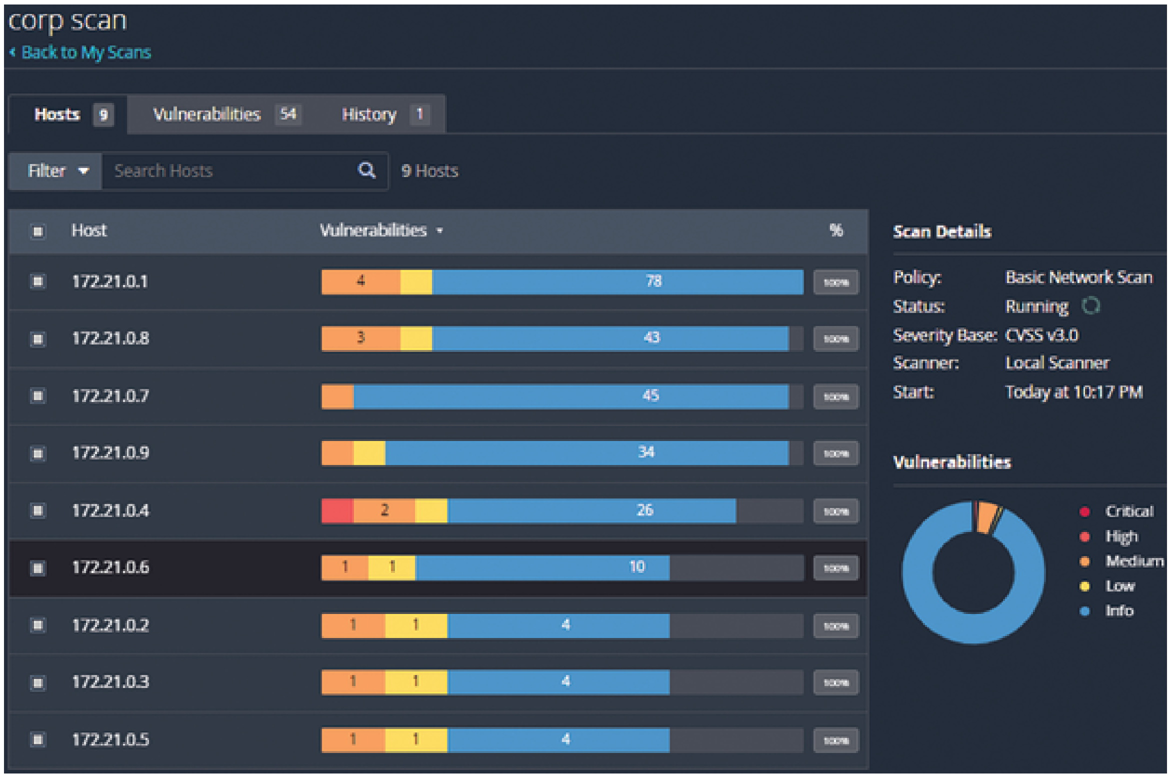
Scan moved to trash successfully.

1 Scans

	Name	Schedule	Last Scanned
☐	corp scan	On Demand	N/A

Figure 10.12 **Nessus scan overview.**

It's going to take a bit of time for it to complete the scan. After the scan finishes, you can click on it to see an overview of the results (Figure 10.13).



Host	Vulnerabilities	%
172.21.0.1	4 78	100%
172.21.0.8	3 43	100%
172.21.0.7	45	100%
172.21.0.9	34	100%
172.21.0.4	2 26	100%
172.21.0.6	1 1 10	100%
172.21.0.2	1 1 4	100%
172.21.0.3	1 1 4	100%
172.21.0.5	1 1 4	100%

Scan Details

Policy:	Basic Network Scan
Status:	Running
Severity Base:	CVSS v3.0
Scanner:	Local Scanner
Start:	Today at 10:17 PM

[Figure 10.13](#) **Scan results.**

Here, you can click on each individual asset to see which vulnerabilities each asset has. This walkthrough is just a quick overview of how to perform a vulnerability scan in an agentless configuration. The other features are outside the essential license provided by Nessus for free. The next step would be to go through the vulnerabilities and identify the impact if they were exploited, and finally, what actions can be taken to mitigate the risk of the vulnerability.

10.4 Summary

We have now implemented Nessus in our environment, which gives us the ability to perform basic scanning of our environment for vulnerabilities. This leads to finding multiple vulnerabilities affecting our environment that a threat actor can exploit. The benefits of vulnerability management include:

1. Visibility into Security Posture

- Regular Scanning: By conducting regular vulnerability scans, organisations gain valuable insights into their security environment. These scans help identify and catalogue vulnerabilities across all systems and applications, providing a clear picture of potential weaknesses.
- Awareness of Vulnerabilities: Understanding what vulnerabilities exist is the first step in getting visibility into the environment.

Vulnerability management helps in assessing the severity and potential impact of each vulnerability, allowing organisations to prioritise their remediation efforts effectively. By addressing critical vulnerabilities first, vulnerability management minimises the risk of exploitation. With a clear understanding of vulnerabilities, organisations can allocate resources more efficiently, focusing on areas that present the highest risk and require immediate attention. By conducting regular vulnerability scans, organisations gain valuable insights into the security of their environment. These scans help identify and catalogue vulnerabilities across all systems and applications, providing a clear picture of potential weaknesses. Understanding what vulnerabilities exist is the first step in securing an environment. This awareness allows organisations to take the steps needed to secure their environment, rather than react to incidents after they occur.

Note

1 https://cve.mitre.org/

Chapter 11
DevSecOps and Web Protection

We have now implemented multiple controls to help mitigate the probability of a material cyber incident. In this chapter, we will examine the trust we place in the code that our internal development team has created and how DevSecOps practices, which emphasise the importance of creating secure code and integrating security into the development process, can help. We will also discuss the DevOps framework called CALMS, static and dynamic code analysis, and introduce open-source tools to help with code analysis. Finally, we will deploy a Web Application Firewall (WAF) as a protective layer in front of our web shop.

Organisations with in-house development teams often fall into a pattern of inherently trusting their code's security. However, this trust is rarely justified. Even large companies with substantial development teams, such as Meta, Apple, and Microsoft, have discovered vulnerabilities in their code that attackers have successfully exploited. However, Zero Trust does not lead to the writing of more secure code. The removal of trust in the development process can go a long way towards improving security. Simple things, such as removing hard-coded secrets from the code, such as IP addresses, application programming interface (API) keys and passwords, can go a long way. This is where DevSecOps comes into the picture. It is a cultural and technical movement that integrates security practices into the DevOps process. By embedding security into every phase of the software development lifecycle, DevSecOps' goal is to help developers create more secure code and reduce the likelihood of vulnerabilities making it into production. Another DevSecOps principle is rapid deployment, which allows developers to quickly fix any code issues, thereby reducing the time of exposure. All this requires that we give the responsibility for the security of the code and the environment to the developers themselves. Of course, this process is guided and audited by the security team. This requires a major cultural change in the organisation. Most developers are rewarded for how quickly they can produce code, with security not even being an afterthought at most. What needs to change is a culture that prioritises security higher. This requires leadership that prioritises security, and accepting that development might slow down a little bit. If they are not on board, the initiation will fail. Therefore, as a security professional, one of the greatest skills to have is the ability to communicate the value of security to the rest of the organisation.

11.1 The CALMS Framework

For this to succeed, security needs to understand the core value that drives DevOps. For that, we will look at a DevOps framework called CALMS. That should help us understand the mindset of developers. The CALMS framework comprises five principles: Culture, Automation, Lean, Measurement and Sharing.

- The first principle, Culture, focuses on fostering a collaborative environment where developers, operations and security teams work together seamlessly. DevOps stresses the importance of eliminating silos and advocating a unified security approach, ensuring all team members share the goal of maintaining robust security practices throughout the development lifecycle.
- Automation is the second principle, and it involves integrating security tools and processes into automated Continuous Integration/Continuous Deployment (CI/CD) pipelines. This integration ensures that security practices are consistent and repeatable, minimising the risk of human error and allowing for more efficient detection and remediation of security issues. By embedding security into the automation process, teams can achieve faster delivery times without compromising on security standards.
- The third principle, Lean, highlights the importance of efficiency and minimising waste. In the context of DevSecOps, this means reducing the time spent on manual security checks and focusing on streamlining processes to enhance productivity. Lean practices advocate for the continuous improvement of workflows and the elimination of unnecessary steps, ultimately leading to a more agile and responsive security posture.

- Measurement, the fourth principle, underscores the use of metrics to track the effectiveness of security practices. By systematically measuring and analysing data, teams can gain insights into the performance of their security measures and identify areas that require improvement. This data-driven approach enables organisations to make informed decisions and continuously refine their security strategies to address emerging threats.
- The final principle, Sharing, encourages knowledge sharing and communication across teams. Sharing information and good practices helps organisations improve everyone's security awareness. This collaborative approach not only enhances the overall security posture but also fosters a culture of continuous learning and improvement, empowering teams to stay ahead of potential security challenges.

The idea of CALMS is that it is supposed to be a guide for directing the development processes with the goal of creating an optimal development environment.

11.2 OWASP Top 10

Having a culture of openness and collaboration is not enough. We also need to be informed about the most common vulnerabilities that are seen in applications. For us to have any chance of discovering vulnerabilities in our application and fixing them. To help with this, we will look at Open Web Application Security Project (OWASP), which is a non-profit foundation dedicated to improving software security. Founded in 2001, OWASP operates as an open community, freely providing articles, methodologies, documentation, tools and technologies for application security. One of OWASP's most well-known and widely used projects is the OWASP Top 10. This list represents a broad consensus about the most common and critical security vulnerabilities in web applications. Updated periodically, the OWASP Top 10 is developed by a team of security experts from around the world. It serves the purpose of:

1. **Education**: It raises awareness of the most important web application security risks.

2. **Guidance**: It provides developers and security professionals with a prioritised list of areas to focus on.
3. **Standardisation**: It offers a common language for discussing web application security risks.
4. **Benchmarking**: Many organisations use the OWASP Top 10 as a baseline for their security standards and practices.

In the context of Zero Trust Architecture, understanding and addressing the OWASP Top 10 web application security risks is crucial. These risks represent the most critical security flaws in web applications, and mitigating them aligns closely with Zero Trust principles. Let's examine each of these risks and their relevance to Zero Trust.

11.2.1 OWASP Top 10 Risks and Their Zero Trust Relevance

1. **Broken Access Control**
   - *Risk*: Improper implementation of restrictions on what authenticated users are allowed to do.
   - *Zero Trust Relevance*: Emphasises the need for granular, context-aware access controls in web applications.
2. **Cryptographic Failures**
   - *Risk*: Failures related to cryptography in web applications, often leading to exposure of sensitive data.
   - *Zero Trust Relevance*: Highlights the importance of encrypting data both in transit and at rest, even within web applications.
3. **Injection**
   - *Risk*: User-supplied data is not validated, filtered, or sanitised by the web application.
   - *Zero Trust Relevance*: Underscores the need to treat all input to web applications as potentially malicious.
4. **Insecure Design**

- *Risk*: Flaws in the design and architecture of the web application.
- *Zero Trust Relevance*: Emphasises the importance of security-first design principles in web application development.

5. **Security Misconfiguration**
    - *Risk*: Improper configuration of web application stacks, frameworks, and platforms.
    - *Zero Trust Relevance*: Highlights the need for secure configurations and regular audits of web application environments.
6. **Vulnerable and Outdated Components**
    - *Risk*: Using components with known vulnerabilities in web applications.
    - *Zero Trust Relevance*: Emphasises the importance of continuous monitoring and updating of web application components.
7. **Identification and Authentication Failures**
    - *Risk*: Incorrect implementation of authentication and session management in web applications.
    - *Zero Trust Relevance*: Underscores the critical nature of robust authentication in a Zero Trust model, particularly for web-based services.
8. **Software and Data Integrity Failures**
    - *Risk*: Web application code and infrastructure that does not protect against integrity violations.
    - *Zero Trust Relevance*: Highlights the need for verifying the integrity of all software and data in web applications.
9. **Security Logging and Monitoring Failures**
    - *Risk*: Insufficient logging and monitoring of web application activities.
    - *Zero Trust Relevance*: Emphasises the importance of comprehensive logging and real-time monitoring of web application events.

10. **Server-Side Request Forgery (SSRF)**
    - *Risk*: The web application server makes requests to an unintended location.
    - *Zero Trust Relevance*: Highlights the need for strict control over all network communications initiated by web applications.

Only through awareness of these risks are we better equipped to help the development team mitigate these risks to our internally developed systems and verify that any third party has done their due diligence in securing their application. By integrating OWASP's Top 10 mitigations into a Zero Trust Architecture, organisations can create a more robust and resilient security posture for their applications. This approach addresses security at multiple layers, from web application design and development to network architecture and access controls.

11.3 Applying Zero Trust to the Development Process: Static and Dynamic Code Analysis

The question we need to answer is how to remove trust from the development processes. We do that by first throwing out the idea that the code developed by the internal team is secure by default. Instead, we have to implement a process that will test and verify the security of the code being written. A good development practice is to have code review processes, where multiple people will review any code committed to the code base. Thus reducing the probability of problematic code entering the application. From a security perspective, the problem is that insufficient training prevents developers from identifying vulnerable code. Most code review practices are about verifying that code is functional and fulfils the change request while also making sure that it follows the code practices laid out by the team. For code review to work from a security perspective, we have to put in the effort of teaching the developers what it means to write secure code. This allows developers to catch potential flaws as the code gets written. It can even start earlier during the design phase. The idea is to shift when security gets involved as far left in the process as possible. This concept is called secure by design, which you can read more about over at Cybersecurity Information Sharing Act (CISA).[1] Using a security-by-design approach allows developers to incorporate security from the initial stages of application development. Studies[2] have shown that training developers on how to create secure technologies can reduce the number of vulnerabilities by 47–53%.

Unit testing is another approach to verifying the code. Unit testing involves writing logical tests into the code base. Each build of the project runs these tests, verifying the code's functionality. This process ensures that no unforeseen changes have affected the code's functionality. We can also extend this idea to include security tests. We intentionally send malicious code to the application, and see if it reacts as expected. Here's an example of how one might test for a SQL-injection vulnerability.

```
@Test
public void testSqlInjectionVulnerability() {
  String maliciousInput = "1'; DROP TABLE users; --";
  assertFalse(isSqlInjectionSafe(maliciousInput));
}
```

The injection attempts to delete the user table, and testing verifies that the table remains. Having a good coding standard is not enough. What we need is a way to automate the process of analysis of code. There currently does not exist a solution that will make any code 100% secure, nor do I think it will ever be possible. There are currently two approaches for performing automated code analysis, the first being static code analysis. These processes involve automatically examining the source code without executing it. Most of the static analysis tools are programming language-specific, such as pylint, which is a static code analysis for Python only. When it comes to selecting a code analysis tool, it is important to verify that the tool is compatible with the programming language being used by the development team and can find security vulnerabilities that matter to you. Below is an example of code analysis tools that exist:

- SonarQube: Analyses code quality and security for multiple languages.
- Bandit: Focuses on security issues in Python code.
- ESLint: Lints JavaScript and ensures code adheres to predefined rules.

The other approach is Dynamic Code Analysis, which tests the application during runtime. This is done by having the analysis tool send malicious and bogus data to the application and detect how it responds. An example for a web application would be interacting with the front end. This method can provide insights into how the applications behave under various conditions during runtime. This can uncover vulnerabilities that static analysis might miss, which arise from the interaction between different components, like race conditions. When selecting a dynamic code analysis tool, there are some considerations that have to be thought about, whether the tool has the ability to interact with the application during runtime, and is it able to find the type of vulnerabilities that you are worried about. The overhead introduced by this type of tool is another factor to consider. While thorough analysis is important, excessive overhead can slow down development and testing processes. Forcing developers to wait for a response from the security tools, before they can either commit the code or deploy the application. It all comes down to balancing speed and security. To summarise, here are the main benefits of code analysis:

- Early Detection: Identifies vulnerabilities early, reducing the cost and effort required to fix them.
- Continuous Improvement: Regular code analysis helps maintain high code quality and security.
- Compliance: Ensures that code meets regulatory and industry standards.

11.4 Web Application Firewall

Now that we understand not to trust the code being developed. We can extend the idea of not trusting the code to what's already in production. What we can do is remove trust from request coming to the web application, whether it is internal or external. This requires that we constantly monitoring the request to the application for any maliciousness, and automate the response to these malicious requests by blocking them.

This is where a (WAF) comes into the picture. WAFs represent a perfect example of Zero Trust principles in action. Instead of trusting web traffic, a WAF examines each request with equal scrutiny, no matter its origin. This aligns perfectly with the Zero Trust mindset of 'never trust, always verify'.

WAFs are web applications-aware firewalls that sit in front of the web applications they protect. It works by continuous verification happens through several mechanisms. The WAF inspects every request, looking for patterns that might indicate malicious intent. This allows it to detect malicious actions such as SQL injection and cross-site scripting attempts. Upon detection, the system blocks these threats before they reach the application. Figure 11.1 shows the WAF added to the network diagram in front of the web application, creating an additional layer of protection.

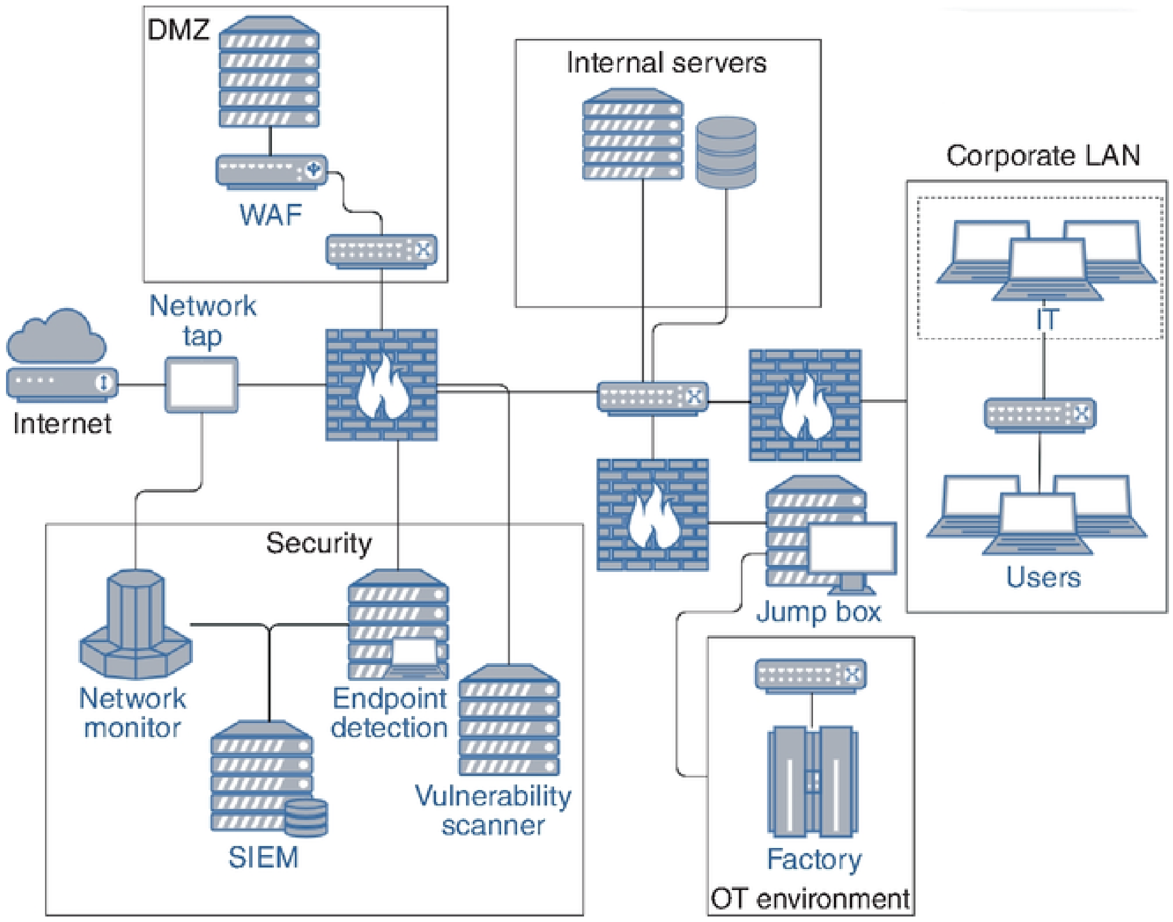


Figure 11.1 Network diagram.

Remember that a WAF should only be considered a band-aid for the application, as it does not remove the application's vulnerabilities. They only make it harder for a threat actor to exploit the existing vulnerabilities. As with anything in cybersecurity, a skilled threat actor can bypass WAF protection and execute payload code on the system. WAF should never be the sole protector of any applications. It can never replace the practices of good DevSecOps, like continuously monitor the code base for vulnerabilities and fixing them in a timely manner as they are discovered.

11.4.1 WAF Deployment in Our Environment

Juice Factory web application called Juice Shop is used for e-commerce. This web application is created with the purpose of being intentionally vulnerable to most attacks. The idea of the Juice Shop project is about teaching both developers and penetration testers about the vulnerabilities that can exist in web applications. We will use OWASP ModSecurity WAF to create a layer of protection between the web application and the users. Here's the Docker configuration for the WAF setup.

```
waf:
  image: owasp/modsecurity:3.0.7-alpine
  container_name: waf
  environment:
    - paranoia=1
    - BACKEND=http://juice:3000
  ports:
    - 9090:80
  networks:
    - dmz
juice:
  image: bkimminich/juice-shop
  container_name: webapp
  expose:
    - 3000
  networks:
    - dmz
```

In the setup above, the WAF proxies the Juice Shop web application, shielding it from common web threats. The WAF that we are deploying for this setup is OWASP ModSecurity, commonly referred to as ModSecurity, it is an open-source WAF engine designed to enhance the security of web applications, and is a part of the OWASP. ModSecurity aims to protect applications from a wide range of attacks, including SQL injection, cross-site scripting (XSS), and other common web vulnerabilities. A unique feature of OWASP ModSecurity is the Core Rule Set (CRS) which introduces the concept of 'paranoia levels', there are currently four level paranoia. Allow administrators to adjust the strictness of the security rules applied by ModSecurity, balancing between security and usability. The security levels range from 1 to 4. Each level adds additional security rules and stricter filtering:

1. Level (Default):
    - Provides basic protection against common attacks
    - Minimal false positives
    - Suitable for production environments
2. Level:
    - Adds tighter security rules
    - Includes additional pattern matching
    - May generate some false positives
    - Recommended for most business applications
3. Level:
    - Implements strict protocol enforcement
    - Adds complex pattern matching
    - Higher likelihood of false positives
    - Suitable for high-security applications
4. Level (Maximum):
    - Enables all security rules
    - Implements most stringent filtering
    - High probability of false positives
    - Reserved for extremely sensitive applications

Each increase in paranoia level adds protection but also increases the likelihood of legitimate traffic being blocked. Organisations must balance security requirements against application usability when selecting an appropriate level. For this demonstration, we will use the Paranoia Level 4. This enables all the rules, offering maximum protection. This might cause false positives, which could block legit users.

Before you deploy the WAF to the environment, I would like to show how the system responds to attacks before and after deploying the WAF. To show the effectiveness of ModSecurity, let's try to exploit the application using SQL injection to log in as the administrator account. The first step is to head over to the Juice Shop over at http://localhost:9090. Then go to the login page of the Juice Shop, by pressing on Account, shown in Figure 11.2.

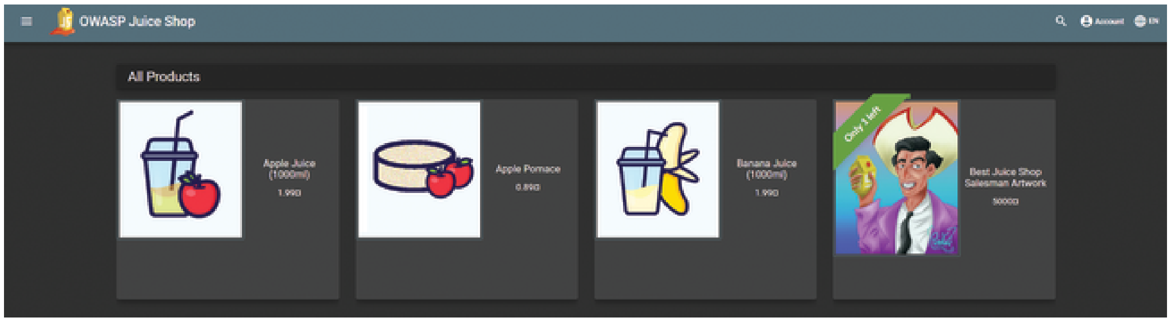


Figure 11.2 Juice Shop frontpage.

At the login screen, type the following as the email:

```
' or 1=1;--
```

For the password field, anything can be inserted. When both are filled out, it should resemble Figure 11.3.

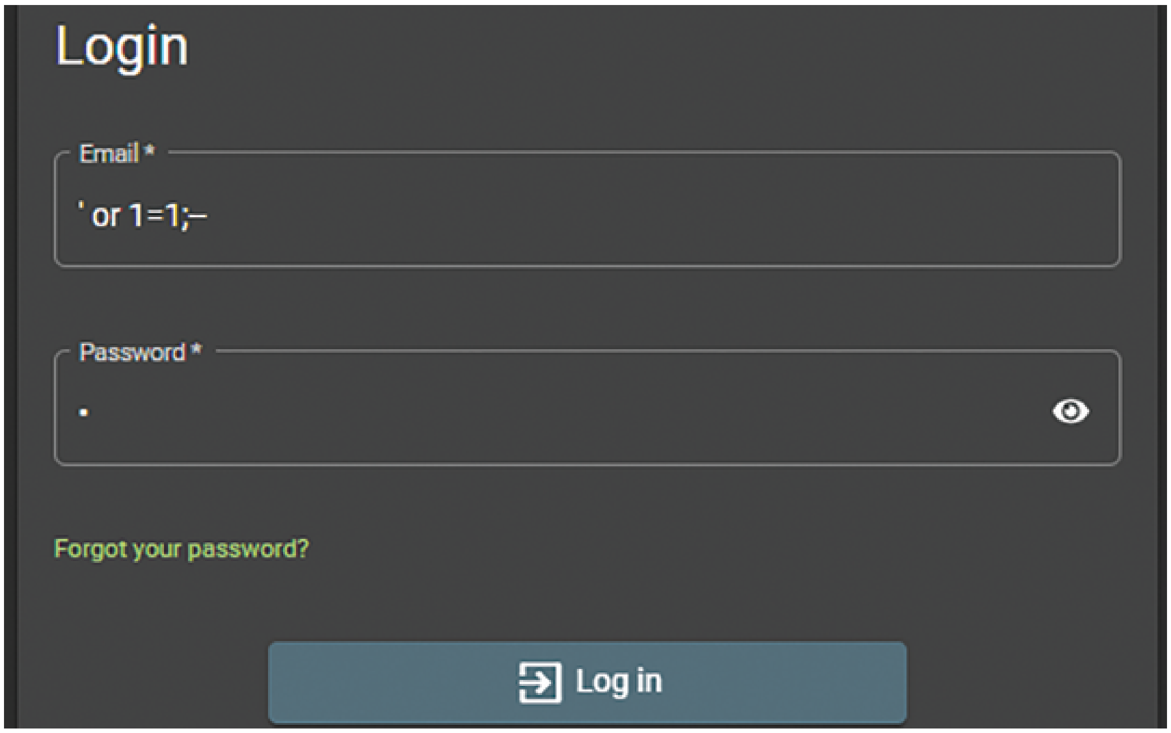
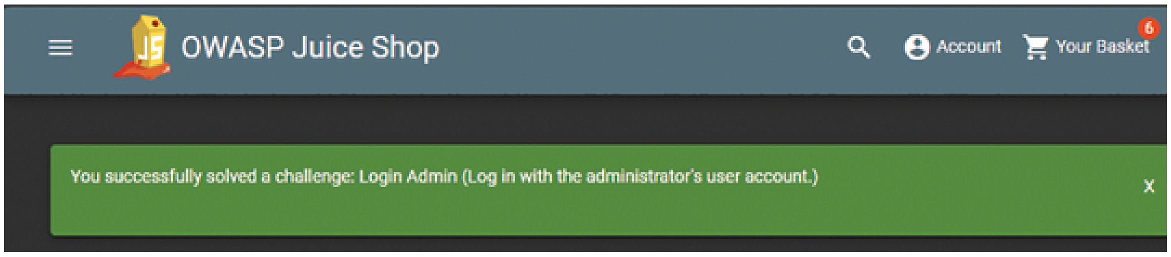


Figure 11.3 Juice Shop login page injection.

Press login, and you should see a pop-up message congratulating you for successfully logging in as the admin, similar to Figure 11.4.



You successfully solved a challenge: Login Admin (Log in with the administrator's user account.)

Figure 11.4 Juice Shop admin login.

Similar to what is shown in Figure 11.5, pressing the account button shows that you are now logged in as the admin.

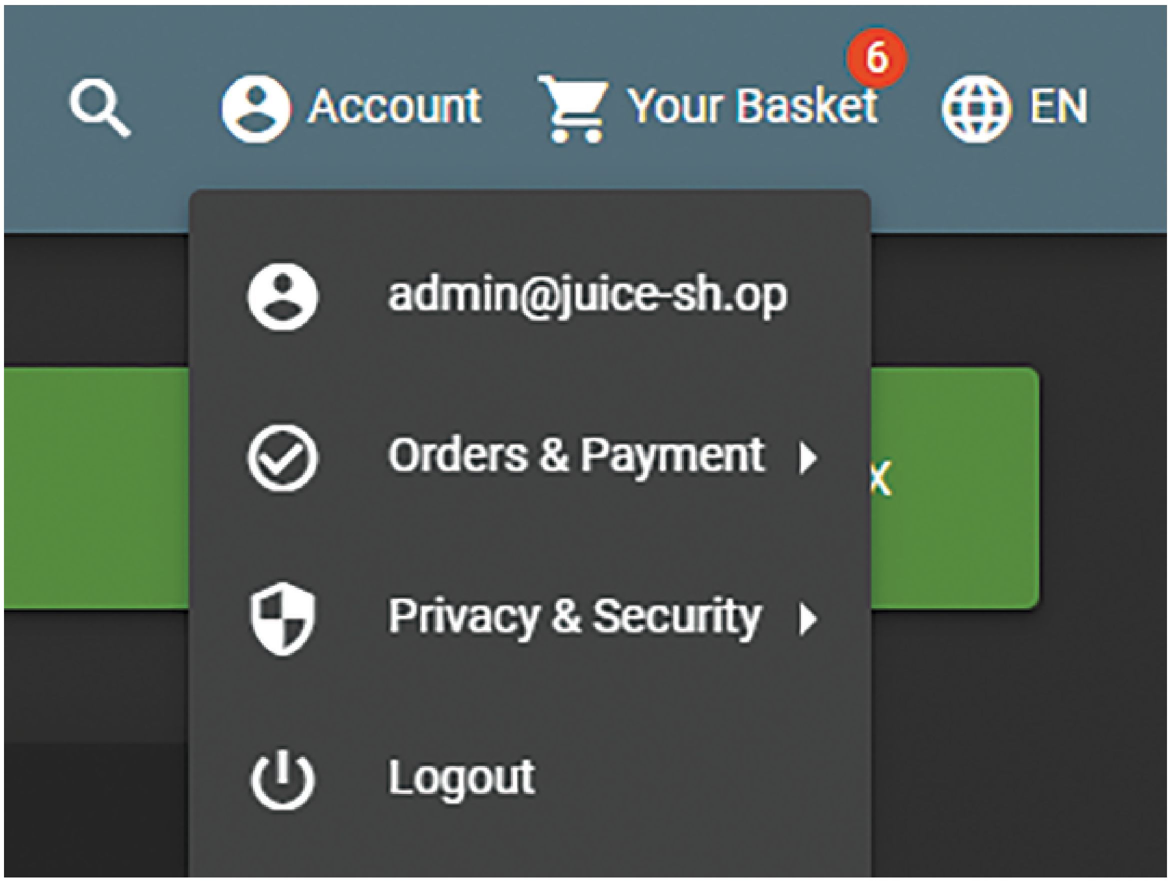


Figure 11.5 Juice Shop admin profile.

This type of attack works because the application is vulnerable to injection attacks, as we talked about in the OWASP section. SQL injection remains one of the most critical web application vulnerabilities. To understand its impact, let's examine how it works in an example. Here is how a SQL query for checking the account validity (simplified) could look like:

```
SELECT * FROM users WHERE email = '[user_input]' AND
password =
    '[password_input]'
```

When we input: ' or 1=1;–, the query becomes:

```
SELECT * FROM users WHERE email = '' or 1=1;--' AND
password =
    'anything'
```

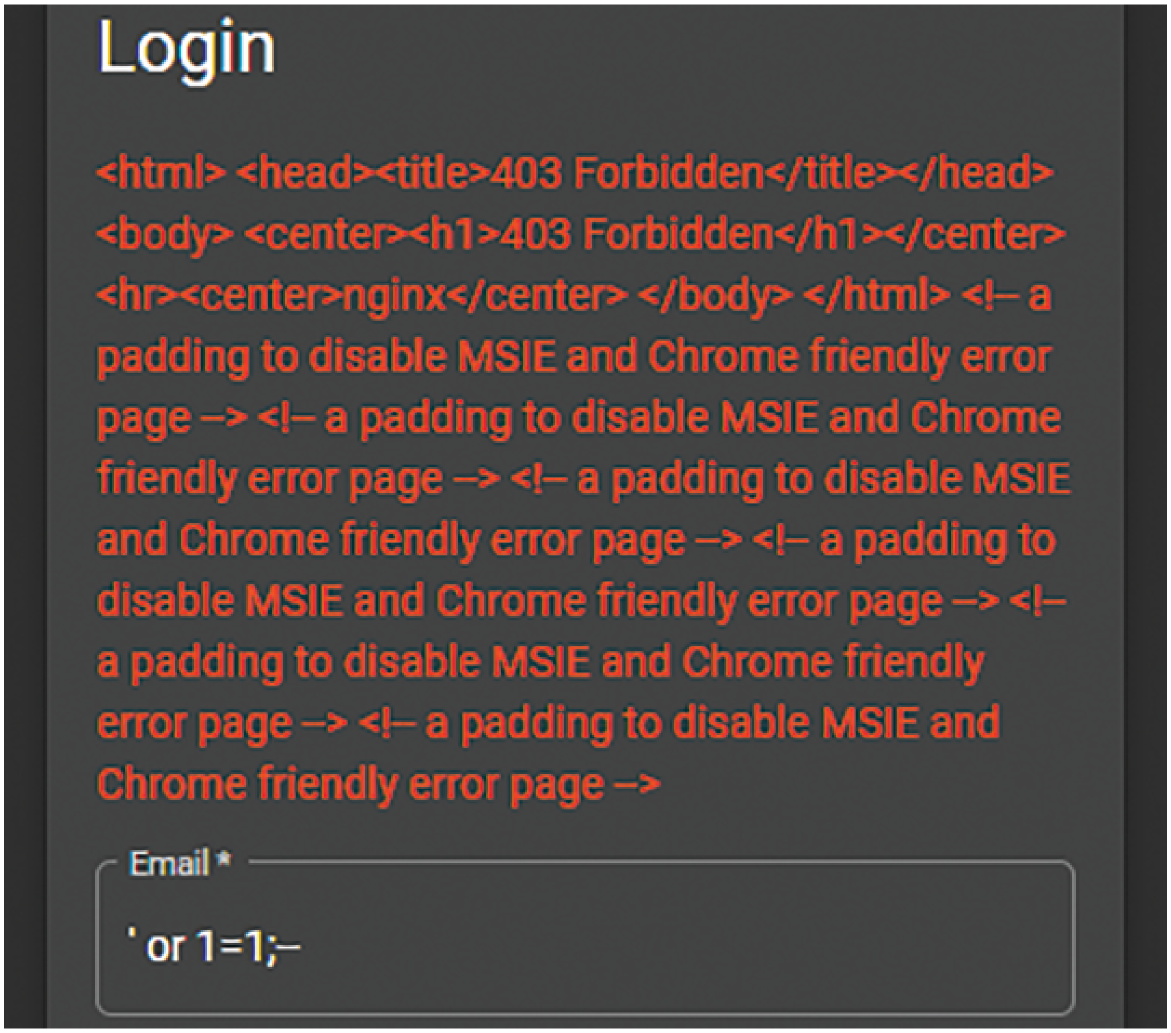
This injection works by:

- Closing the email string with a single quote (’)
- Adding an OR condition that’s always true (1=1)
- Commenting out the rest of the query (–)

The modified query returns all true because the ‘OR’ condition is always true. Since the administrator account is typically first in the database, we gain administrative access. This example shows why input validation and parametrised queries are crucial security controls. Let’s try the same again, this time with the web application protecting with the WAF. Start up docker-compose from Chapter 12 and wait for the web application and WAF to get up and running. This can take a few minutes. Once ready, head over to:

```
http://localhost:9090/#/
```

Now try the same exploit once again. Instead of getting access to the admin account, see an error message pop up instead (Figure 11.6).



Login

```
<html> <head><title>403 Forbidden</title></head> <body> <center><h1>403 Forbidden</h1></center> <hr><center>nginx</center> </body> </html> <!– a padding to disable MSIE and Chrome friendly error page –> <!– a padding to disable MSIE and Chrome friendly error page –> <!– a padding to disable MSIE and Chrome friendly error page –> <!– a padding to disable MSIE and Chrome friendly error page –> <!– a padding to disable MSIE and Chrome friendly error page –> <!– a padding to disable MSIE and Chrome friendly error page –>
```

' or 1=1;–

Figure 11.6 **WAF protection.**

The Juice Shop has many other exploitable vulnerabilities. I recommend anyone that has not tried their hand at exploiting the Juice Shop. To try getting some experience with web exploitation, as there is a lot to be learned about the different vulnerabilities that can exist. It is important to say again that a WAF does not protect against all attacks, especially the more advanced ones. Consider it merely a temporary fix; the proper solution is better input validation of all user input. They get a better idea of the extent to which ModSecurity protects the application. You could try to go through the Juice Shop with the WAF protecting it and see what vulnerabilities you can still exploit.

11.5 Software Bill of Material

I was unsure whether to include a software bill of material or SBOM in this book. Until I tried scanning my code repositories for vulnerable packages. To my surprise, I found that many of the packages I used had vulnerabilities. After this realisation, I knew SBOM was a necessity that had to be included in the book.

You might wonder, well, what is SBOM? SBOM became popular after the Log4j vulnerability, as organisation scrambled to identify what software actually used Log4j, and which version is used. The SBOM, a record of all software components, promotes transparency between users and developers. It functions similarly to a food product's index, informing customers about the product's ingredients to help them make better choices. The same idea applies to software with SBOM. When a business wants to buy a piece of software, the SBOM will list all the components used to build the application, including all the subcomponents used in those components.

The goal of SBOM is to remove the trust in the software that we are running in our organisation, whether it is open source software or closed source. SBOM makes it possible to verify that the components used have no vulnerabilities present. Giving the customer a more transparent view of the risk involved in running the application. The problem is not that simple to fix. Remember that SBOM is not only for developers but also for the security team, the end user, and anyone else interested in the application. Giving them the ability to view the 'ingredients' of the application, so we need a standard. OWASP has already developed a standard called CycloneDX[3] that specifies how to create SBOMs. It is currently the most adopted standard with a wide range of vendors already adopting the CycloneDX standard. The full list can be viewed on the site.[4] I will not go into details about how it works. I believe that is something you have to find out for yourself. The implementation processes depend not only on what programming language you are using, but also on how they fit into your CI/CD pipeline. One thing is sure that it should be automated into the DevOps process, and notify and stop the deployment whenever there are any known vulnerable package in the build. I therefore recommend anyone who is writing their own code to look into SBOM to help create more transparency.

Systems are not just made up of code; there are also hardware components that the code needs to run on. That means we can extend the idea of SBOM to hardware called HBOM or hardware bill of material. The best current example of bad HBOM control is Lebanon pager explosions, where Israel managed to either intercept the delivery of the pages or control a middleman from the production to the delivery of the pages. In the moment of control, they implanted explosives into the device with no one knowing. Of course, having an HBOM is not enough. A validation step is also required that verifies that the device contains only the hardware listed in the HBOM. That requires hardware experts who can actually verify that HBOM statement. Which can be an expensive process, and should therefore be limited to governance institutions and major business. You can read more about HBOM over at CISA.[5]

11.6 Applying DevSecOps to Security

Applying the DevSecOps idea to just development practices is not enough. We should also apply it to our own practices. What does this mean for security? In the SIEM chapter, we talked about how we could use a SOAR platform to automate processes of security, where we transform the procedures for handling incidents into playbooks, which are instructions the machine can follow, allowing security to automate the part of the process, which is one of the core principles of DevSecOps. That is just one way of implementing DevSecOps into security. We are not just limited to the security operation with DevSecOps. Another element of security that can benefit from this process is compliance. Having to go through the entire environment to check if it’s compliant with a standard like ISO 27001 can be very time-consuming. So why not automate it? This is, of course, easier said than done. I can with high confidence say that your environment comprises a suite of different devices and systems, and added to the complexity, there is also the cloud you have to consider. Here, it would be exceptional if I could just point to a single tool that would solve this problem. Sadly that is not the case. With endpoint, this includes Windows, Mac and Linux. I would say the best and easiest tool I have found is FleetDM,[6] which is an open-source orchestrated tool that uses OSQuery,[7] for device monitoring, Fleet also provides limited support for the major cloud providers. When it comes to supporting compliance standards, then out of the box fleet supports CIS benchmarks,[8] if you want to audit against other standards or regulations, that’s something you would have to set up yourself. You can do this using query commands. The example query below checks if the SMB1 protocol is disabled on Windows machines.

```
SELECT 1 FROM windows_optional_features WHERE name =
'SMB1Protocol-Client' AND state != 1;
```

Having auditing procedures transformed into code instead of manual processes ensures consistency, that it is done the same way each time and allows for version control. The fleet will continuously monitor every device for any deviation from the standard and alert the administrators when that is the case. For more dedicated cloud auditing tool, most of them have built-in auditing capabilities that come at a financial cost. Luckily for us, there are exceptional people on the internet that have made open-source tools that can provide automated auditing capabilities. The tool is called Prowler.[9] It is an open source security tool for performing security best practices assessments and auditing on AWS, Azure, Google Cloud and Kubernetes. It can audit an entire environment using a single command. Here is an example of auditing Azure against CIS 2.1 controls.

```
prowler azure --compliance cis_2.1_azure
```

Prowler uses JavaScript Object Notation (JSON) files to define what to audit and which checks to perform. Going to detail how to create your own JSON file, to do auditing it outside the scope of this book, but you can read more about how to create your own in the documentation.[10]

You might ask yourself, why just automate the auditing of the things you have made? Why not transform the entire infrastructure into code and automate the whole creation of the system instead? That way, it is possible to do code analysis for any potential vulnerability and for compliance. That kind of infrastructure we called infrastructure as code (IaC). It is very similar to our Docker environment, where you have a configuration file that defines how the environment should look, and then the engine, in our case, Docker, ensures that the environments get spun up according to the specification. As of this writing, one of the most popular multi-cloud solutions is Terraform.[11] The general idea of

Terraform is very similar to Docker Compose, of using configuration to define the environment. One thing of note about infrastructure as code is that the cost of the environment can quickly skyrocket, as everything is just code, it is hard to get an idea of the cost of things. I suggest you limit the amount of money and resources your Terraform learning account can use. Going full Infrastructure as code might be too big of a step to take. A middle ground is using a hardened image or hardened script. This ensures that all systems start from the same baseline. From there, you adjust the image to better fit your needs. The Center for Internet Security (CIS) has created hardened images,[12] based on their benchmarks,[13] that have been developed with the help of a global community of cybersecurity experts. Ensuring that the base image you're using has been set up to be secure by default.

By aligning security practices with DevSecOps, organisations can create processes that are automated, resilient, adaptable, and aligned with the Zero Trust philosophy of 'never trust, always verify'.

11.7 Summary

In this chapter, we discuss the importance of integrating DevSecOps practices, which are essential for developing secure software. Such as utilising the CALMS framework, developers can create a collaborative and efficient environment that emphasises security. This chapter also discusses static and dynamic code analysis that can further enhance code quality and security. And finally, how deploying a WAF can create an additional layer of protection. Following the practices talked about will help create a more secure development process.

Notes

1 https://www.cisa.gov/securebydesign

2 https://cyberscoop.com/wp-content/uploads/sites/3/2024/10/Developer-Readiness-Analysis-Secure-by-Design-10.15.24.pdf

3 https://cyclonedx.org/

4 https://cyclonedx.org/tool-center/

5 https://www.cisa.gov/resources-tools/resources/hardware-bill-materials-hbom-framework-supply-chain-risk-management

6 https://github.com/fleetdm/fleet

7 https://github.com/osquery/osquery

8 https://www.cisecurity.org/cis-benchmarks

9 https://github.com/prowler-cloud/prowler

10 https://docs.prowler.com/projects/prowler-open-source/en/latest/developer-guide/security-compliance-framework/

11 https://www.terraform.io/

12 https://www.cisecurity.org/cis-hardened-images

13 https://www.cisecurity.org/cis-benchmarks-overview

Chapter 12
What About People?

> What about people? They are everywhere, like ants messing around with our systems! Can we really say that we have implemented zero trust if we have not talked about the people that make up the organisation?

People, like any other system or protective surface, have a life cycle, from when they are first interviewed to when they leave or retire from the organisation. We have to protect the entire cycle of our people. Traditionally, security professionals have viewed people as ‘the weakest link’. This mindset, while common, is misaligned with Zero Trust. Instead of treating people as a vulnerability to be contained, we must recognise them as active participants in our security architecture. They represent both our largest attack surface and our most dynamic defence mechanism. A common mindset to have about people is ‘if we really want to protect the organisation, we need to put in technical controls, as nothing else will work, as we can not trust people not to click on links’. The problem with this mindset is twofold. First, it requires implementing so many controls that employees can’t effectively do their jobs. Second, it creates an adversarial relationship between security and employees. The other assumption is just to let people get compromised and instead have effective control in place in response to such an event. It is kind of like having the mentality of why have speed limits on the road, when we have seatbelts and airbags in cars to protect us.

With the belief that people are the ‘weakest link’, we are then setting ourselves up for failure, not only because it is wrong, but because it changes the way we treat them. A more accurate saying would be ‘People are the largest attack surface in our organisations’. They are also the only link in the chain that binds everything together. Making them the most targeted protected surface we have in our organisation.

Just to head it home, why do you need to not only believe but also trust that people can learn about security? In the 1960s, Robert Rosenthal created an experiment with the goal of identifying how one's perception of other people can develop into a sort of self-fulfilling prophecy, how one's conscious and unconscious treatment of people can steer their development in a direction that fits that perception. He partnered with an elementary school, and together they told teachers at the school that the worst-performing students were the best and the best, and vice versa.[1] At the end of the year, they tested the students again, and the students whom the teachers believed were the 'best' had outperformed their classmates.

Robert Rosenthal's 1968 study showed how believing in people's ability to learn and grow makes a big difference in education. Known as the 'Pygmalion Effect' or 'Rosenthal Effect', this research demonstrated how expectations can become self-fulfilling prophecies. In the study, teachers were told that certain students (randomly selected) showed exceptional potential for intellectual growth, despite these students being chosen at random. By the end of the year, they showed significantly higher academic achievement. The teachers' positive expectations had unconsciously influenced their behaviour towards these students, providing more encouragement, feedback and learning opportunities.

This phenomenon has direct implications for security awareness programs. When we label people as 'the weakest link' or treat them as security liabilities, we may unconsciously limit their potential for growth. Instead, by approaching employees as capable partners in security and setting positive expectations, we can create an environment where security awareness flourishes. Likewise, if we continue with the belief that people are the weakest link, then that belief will become the reality because of how we treat them. There is, of course, the caveat that the teachers have far more interaction with the students, then we have with the employees of the organisation, and therefore, the effect should not be to the same degree.

12.1 Culture

Don’t tell me not to click, when clicking is my job!

What defines your organisation’s culture is its expectations, processes, behaviours and rituals. We can influence these through training and reinforce them through policies. The problem with culture is the expectations we have of security awareness, and the metrics we are using to measure its success. The goal should never be to reduce the click rate to zero, instead it is about teaching people how to recognise and report potential attacks. Even if they are the victim of an attack, they should not be afraid of reporting the incident, because of the consequences or being talked down to by the security team. Many times, I have confronted people who had been a victim of a phishing, and they just flat out denied it. For security to succeed, a level of trust and human connections between the security team and the rest of the organisation are required.

Maybe the videos we have forcing people to watch are just bad, and a stupid way of training people. If it really was the most effective way, then why wouldn’t we do it everywhere? They simply say it is because it does not work. The reason for its widespread usage is that it is cheap and easy to buy generic material from a third-party vendor. It could simply not be any easier, and still fulfilling the compliance requirements. We should try to tailor the training messages based on the specific roles people have in the organisation. Of course this is difficult to do, and can quickly become expensive. For that reason, a combination of the two might be the best approach: having the mandatory training everyone should go through, but supplement it with more personal training, to the most targeted or risky users, such as information technology (IT) admin and support, finance and executives.

Another thing we have to understand is how people make decisions. Psychologist Daniel Kahneman’s research identifies two distinct modes of thinking: Systems 1 and 2.[2] System 1 thinking is fast, intuitive and automatic – the kind we use for routine tasks and habit-based decisions. This is how most people handle their daily email, click on links, or respond to requests. System 2 thinking, in contrast, is slow, analytical and requires conscious effort – the kind we want people to use when evaluating potential security threats.

The challenge for security awareness programs is that we cannot expect people to operate constantly in System 2 mode. It's mentally exhausting and impractical. Instead, we need to develop security habits that work with System 1 thinking, making secure behaviour the automatic natural response. This means:

- Creating simple, clear security procedures that can become automatic.
- Building muscle memory through regular practice.
- Establishing environmental cues that trigger security awareness.
- Developing organisational habits that support secure behaviour.

Cybersecurity is often scary for individuals. It is technically challenging, and there can be real consequences for not getting it right. We need to help everyone build an identity with the belief that they can play a role in security. Awareness training is an important component of security culture, but it's not the only component of the equation. Another is skills. An example of this is: does the developer know what it means to create a secure application? Do the IT administrators know how to configure the systems to be secure? This is just to highlight that it is equally important to equip people with practical skills to actually do their job securely.

12.2 Tabletop Exercise

In the same way that corporate buildings conduct yearly fire drills, organisations should regularly practise their cybersecurity incident response plans in simulated environments. These exercises, known as tabletop exercises, serve as crucial preparation for real-world cyber incidents. Tabletop exercises give participants a safe space to practise their procedures, roles and responsibilities, improving everyone's understanding of the incident's phases and individual expectations. These exercises also create valuable opportunities to address hard questions with senior management. For instance, participants can discuss paying ransoms in ransomware attacks, considering the circumstances that might justify such payments and assessing whether their organisation has the necessary cryptocurrency funds to do so. Through these discussions and simulations, organisations can evaluate the effectiveness of their incident response plans in various scenarios and assess whether employees can follow them effectively.

12.2.1 Conducting Tabletop Exercises

While there are many ways to conduct a tabletop exercise, a basic format involves gathering people from across the organisation to discuss how to respond to different scenarios. A moderator typically guides participants through a simulated event, prompting group discussion on various response strategies. In these tabletop exercises, it is beneficial to involve both business partners and IT professionals. This inclusiveness helps business units understand the challenges the organisation will face during an incident. It also allows the business to identify options for how to continue the operations of the business, even when IT systems are unavailable. Perhaps most importantly, it builds connections between departments and fosters trusting relationships between IT and business units, preparing everyone to work together effectively when a real disaster strikes. Having a good communication strategy is critical for combatting the fog of war and creating a situational picture of what is going on.

These exercises clarify the roles people need to play and test whether existing procedures work as intended. Often, teams discover crucial insights during these simulations. For example, they might realise they aren't receiving the necessary logs for investigations or that their understanding of incident response tools is limited. These discoveries are invaluable, because a more effective IT team can contain a breach faster and recover more quickly.

12.2.2 Conducting a Tabletop Exercise: A Sample Scenario

To illustrate how to conduct a tabletop exercise, let’s consider a sample scenario. The first step is to define the learning objectives of the exercise. For instance, you might choose to focus on Cloud. With this focus in mind, you would create a series of events related to this topic. For each event, prepare possible responses, but keep these hidden from participants to simulate a real incident, where the participant does not know exactly what happened. As you guide participants through the events, encourage open discussion and decision-making at each step. The goal is to create a challenging but educational experience that prepares your organisation for real-world cybersecurity incidents. Remember, it’s perfectly fine for people to make mistakes during these exercises. In fact, it’s preferable to learn from these mistakes in a controlled environment where there are no real consequences, rather than during an actual cyber event. By regularly conducting these exercises, organisations can better understand their roles, improve their incident response capabilities and build a more resilient cybersecurity posture. It’s not just about preparing for the worst; it’s about fostering a culture of security awareness and collaboration across the entire organisation.

Let’s create our own scenario, where we focus on identity access management (IAM). In Table 12.1, you can see an example of events related to identity. You then guide the participants through each event and have them discuss the best way to respond. For this scenario, the whole thing starts off with the user being a victim of a social engineering attack as the initial compromise. Which is the stage the threat actor tries to get access to the environment. Once you have presented the scenario, it is time for the participants to respond. The discussion should include both how to detect the attack and the best method of responding.

TABLE 12.1

Table scenario.

Stage	Initial compromise	Pivot and escalate	C2 and Exfil	Persistence
Event	Social engineering	Credential stuffing	HTTP as Exfil	New users added
Description	Actor tricks a user into running malware	Credentials have been discovered on open shares and files within the environment.	HTTP traffic as exfil and C2	Threat actor adds a new user to the environment
Detection	EDR, network monitoring, User awareness	SIEM, UEBA, cyber deception	Network traffic monitoring	AD log, EDR

The next stage is the pivot and escalate. The initial compromise is often not enough for the threat actor to achieve their objectives. For that reason, they need to escalate their privileges or get access to new resources. Here, the threat actor uses credentials that were available in one of the files on the network share. A good discussion point for this event could be to identify a method of detecting files that contain credentials.

The third stage is C2 and Exfil. In this stage, the threat actors need to find a way for their command-and-control (C2) centre to communicate with the organisational environment continuously. This requires a network connection from the threat actor environment to the organisational environment. In this scenario, the threat actor uses HTTP as their method for communication with their C2.

The final stage is persistence. This is how the attack maintains access to the environment. In our scenario, the threat actor creates a new user either on a local machine or in the Active Directory, depending upon how you want the tabletop exercise to play out. A good discussion point for this event could be if the security team has access to the logs and if they have set up any alerts based on user creation, which is not inherently malicious, and how do they go about verifying that it is a legitimate user?

Throughout the exercise, there will also be injections to create twists in the scenario. They are smaller events that change the flow of the tabletop, like removing a participant because they dominate the discussion and you want the other participants to step up. Another option is the red herrings. During an incident, we have a limited view of what is going on. This is also called the fog of war, where we receive conflicting information. The most effective way to deal with the fog of war is through communication. Ask questions, be transparent, and above all, revise your conclusions as you receive new information.

NIST has, of course, also created a special publication on a tabletop exercise called NIST SP 800-84.[3] This standard defines several key considerations when building a tabletop exercise.

- Define the objectives of the exercise.
- Start building the scenario so it matches the desired outcomes.
- Ensure that the right people are in the room to achieve the desired learning outcomes.

You might think to yourself that you're not very good at coming up with ideas for the tabletop scenario. NIST 800-84 talks about creating a Master Scenario Events List (MSEL) that the moderator uses as a guide to keep the tabletop exercise on track. NIST SP 800-84 provides a sample MSEL in Appendix B-5 of their document. The MSEL outlines a timeline of events, along with a corresponding response.

Another resource is Black Hills Security's Backdoors & Breaches card game.[4] You can find their game and instructions for how to play over on their website. They have even been so kind to create a free online version of the tabletop game over at https://play.backdoorsandbreaches.com/, here you can try your hand at the game. If you find it helpful, I recommend you buy a copy of the game to help support the development of future expansions.

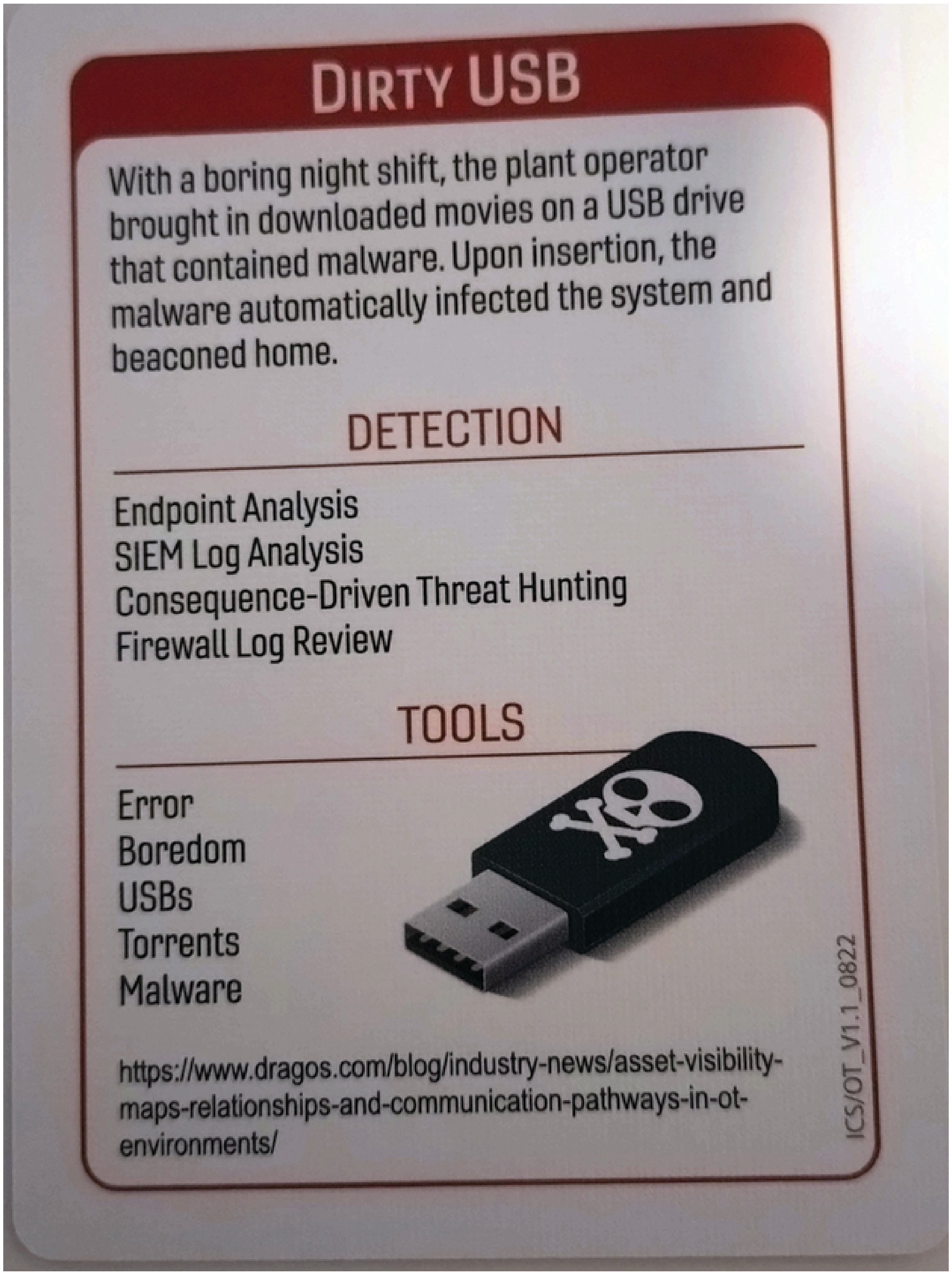
Backdoors & Breaches is a cooperative, turn-based threat emulation game in which participants work together to uncover the attack method used to hack into their environment. The game takes the concept of traditional tabletop exercises, combines the structure of a card game with the flair of classic role-playing games to help make learning about the tactics, methods, and tools used in cybersecurity more interesting. Backdoors & Breaches is a team-based game where players either succeed or fail together. The game begins with a moderator designing a scenario using four attack cards, one of each type, to create a fictional security breach. The moderator then guides players through this scenario.

The game comprises three types of cards.

- Attack – INITIAL COMPROMISE (red), PIVOT and ESCALATE (yellow), C2 and EXFIL (brown), and PERSISTENCE (purple) cards
- Procedure – Blue cards
- INJECT – Grey cards

Each type has a specific purpose. Threat actors use attack cards to achieve their objective. Figure 12.1 shows an example. Procedure cards are how the participants are going to respond to the attacks, and inject cards are for the moderator to alter the flow of the game to help guide it in a certain direction. In addition to the card game, you also need.

- A crew of two or more; the ideal number of players is five to seven.
- D20 (20-sided die) or a virtual dice-rolling app.



DIRTY USB

With a boring night shift, the plant operator brought in downloaded movies on a USB drive that contained malware. Upon insertion, the malware automatically infected the system and beaconed home.

DETECTION

Endpoint Analysis
SIEM Log Analysis
Consequence-Driven Threat Hunting
Firewall Log Review

TOOLS

Error
Boredom
USBs
Torrents
Malware

https://www.dragos.com/blog/industry-news/asset-visibility-maps-relationships-and-communication-pathways-in-ot-environments/

ICS/OT_V1.1_0822

Figure 12.1 Dirty USB attack card from Backdoors & Breaches.

Players work together to uncover the attackers' actions before time runs out, utilising critical thinking skills and procedure cards. The game ends in 10 rounds or when the players have revealed all four attack cards. In each round, players discuss and play a procedure card relevant to the scenario, then roll a die to determine success (1–10 fails, 11–20 succeeds).

If a procedure fails, players must explain why, considering factors, such as finances, politics, personnel or technology. The game offers expansions that introduce additional elements to the game. For a more comprehensive explanation of the rules, look at the visual guide available on the Black Hills Information Security website.[5]

12.3 Summary

Creating a security-aware culture is both the most important and the most difficult task of any security professional job. Basically, 'your job is to make people want to wear their bicycle helmet when they go out for a ride'. The worst that can happen to the security department is being seen as the department of no, where the rest of the organisation is deliberately making sure that security is not aware of the projects going on within the organisation. It is the duty of security to integrate with the culture of the organisation. This requires them to connect with the existing culture of the organisation. The goal is to have them understand we are here to make sure that they and the organisation can operate securely. The best way to do this is through transparency by talking with the business about the challenges faced by security. This way, they feel like they are part of the discussion, creating a sense of ownership of the security program instead of upper management imposing policies on them. Another important factor is to never underestimate the user's knowledge of their own system in which they operate daily. They most likely have knowledge of the system that you are not aware of and never will. For anyone interested in learning about a new way of looking at safety, I can recommend the book *Do Safety Differently*.[6] This book introduces a new way of looking at safety, which is more open to the input of people within the organisation. So, stop hiding behind your screens and start connecting with the rest of the organisation.

Notes

1 Rosenthal & Jacobson (1968) Pygmalion in the Classroom, Urban Review 3(1): 16–20.

2 Thinking fast and slow, by Daniel Kahneman.

3 https://csrc.nist.gov/pubs/sp/800/84/final

4 https://www.blackhillsinfosec.com/projects/backdoorsandbreaches/

5 https://www.blackhillsinfosec.com/wp-content/uploads/2024/03/BnB_VisualGuide_v2_03052024.pdf

6 Book: Do Safety Differently by Sidney Dekker, Todd E Conklin, 2022.

Chapter 13
Journey from Flat Network to Zero Trust

Through this book, we've embarked on a journey to transform our network architecture from a flat, open network into one that is more secure and resilient – a Zero Trust Network. Let's recap how we achieved this transformation and the principles we followed along the way. They start by first laying out the design principles of Zero Trust that we have followed in this book. They are the guiding principles of how we identify and evaluate the security controls that we need to implement in our organisation.

The design principles of Zero Trust:

1. Define Business Outcomes: Define what the business is trying to achieve, and identify how security initiatives will affect the business. The fictive company we are working with, Juice Factory, is a manufacturing business that produces juice products for its customers.
2. Design from the Inside Out: Starting with the things that are most important for the organisation, the crown jewels. For us, it is the factory. From there, try to understand why and how the crown jewels interact with the rest of the organisation and external systems. Allowing us to determine which controls to implement to reduce the likelihood of a cyber event.
3. Determine Who Needs Access: This all comes down to allow-listing for both users and systems. Based on the information from the previous stage. By following the principle of need to know, we determine who needs access based on their job role and who does not. Reducing the trust we put into our environment.
4. Inspect and Log All Traffic: Monitoring is your eyes and ears on what is going on within the environment. Only when we know what is going on do we have the possibility of responding to malicious activities. This requires creating and deploying a monitoring strategy and setting up an alerting plan for any unauthorised activities. The

effectiveness of these hinges on how well we know our environment. The more familiar we are with the environment, the more efficient we can be with the coverage of our monitoring, and the ability to create well-tuned alerting rules that are precisely created to fit their intended environment.

13.1 Cloud

Cloud computing represents a critical component of modern information technology (IT) infrastructure, and you may wonder why we omitted it from our hands-on examples. While cloud services are ubiquitous in today's business environment, the practical constraints of requiring personal credit cards for cloud resources make it impractical for our demonstration purposes. However, the Zero Trust principles we've discussed apply equally to cloud environments, with some additional considerations we'll explore in this section. I would still like to talk about some of the security concerns of the cloud.

While cloud services offer unprecedented ease of deployment, this accessibility creates new security challenges. One significant concern is shadow IT: the unauthorised procurement and use of cloud services by employees with purchasing authority. This creates several risks:

1. Unmanageable Data: Information that is not managed by the IT department
2. Security Gaps: Services that don't meet organisational security standards
3. Compliance Issues: Potential violations of regulatory requirements
4. Cost Control: Untracked and potentially redundant service purchases

This refers to any IT systems or services that aren't managed by the IT department. The problem with shadow IT is that it does not follow the same policy while still containing business data. Creating a potential risk of losing business data, which can be very hard to catch. One approach to catching shadow IT is to ask the finance department for receipts for everything bought using the company card, and see if you can identify anything that is IT-related and not managed by IT. This approach is unachievable for larger organisations, with thousands of people having corporate credit cards. Here, a potential solution could deploy cloud access security broker (CASB). It can help provide visibility into cloud usage. In some cases, it can even provide detailed information on how the application is being used. Of course, no CASB can catch all cloud services, but it is a good starting point for getting more control over what is happening in the cloud.

Remember that the cloud is not just a single surface that needs to be protected. The main thing you have to get right with the cloud is identity access management (IAM), as I discussed in the identity and access management chapter. The dilemma of the cloud is that it's accessible everywhere and at any time. This requires that you have strong authentication and identify-protect the environment, using multi-factor authentication, separation of duties and conditional control. Another challenge with the cloud is that the monitoring of cloud applications can prove difficult as most do not provide the same level of visibility that organisations have when running services on-premises. In the cloud, we are at the mercy of the cloud provider, and what they will share.

For those of you interested in delving deeper into cloud security, I recommend heading over to the Cloud security alliance,[1] if you would like documentation that is more specific to the different cloud providers. I recommend looking at their Well-Architected Framework. Here is an example of the Azure Well-Architected Framework.[2] These resources should be enough to get you started with implementing zero trust in your cloud environment.

13.2 All That We Have Achieved So Far

Let's go over the changes we have made to our environment. At the outset, the network architecture was a flat and open network. This type of network lacks segmentation, meaning that once an attacker breaches the perimeter, they can move laterally with little resistance. Such networks are vulnerable to a wide range of attacks. To align with Zero Trust principles, we implemented several key security measures throughout the book:

1. Network Segmentation: We segmented the network into different zones, such as the demilitarized zone (DMZ), Production, Corporation and Security networks. This segmentation helped contain potential breaches and limited the lateral movement of attackers.
2. Jump Box and Separation of Duties: We incorporated Jump Boxes and separation of duties to ensure that only authorised users could access sensitive resources.
3. Continuous Monitoring and Incident Response: Tools like Suricata and Velociraptor for intrusion detection, and Logstash, Elasticsearch and Kibana for centralised logging and monitoring helped us maintain visibility into network activities and respond promptly to security incidents.

   In Vulnerability Management, we utilised Nessus to scan for, identify, and mitigate potential vulnerabilities before potential exploitation.
4. Web Application Protection: By deploying a Web Application Firewall (WAF), we added a layer of defence against common web threats, though we emphasised that WAFs are not a replacement for secure coding practices.

For the last time, let's look at the Docker environment and what we have implemented throughout this book.

13.2.1 Network Segmentation

We started by segmenting the network from a single flat network into three different zones based on business processes.

```
networks:
  dmz:
    name: dmz
  Production:
    name: Production
    internal: true
  corporation:
    name: corporation
  security:
    name: security
```

13.2.2 Network Monitoring

Setting up Suricata to monitor all network traffic happening on the host machine. This also includes the Docker environment.

```
suricata:
  image: jasonish/suricata:latest
  container_name: suricata
  network_mode: host
  cap_add:
    - NET_ADMIN
    - NET_RAW
    - SYS_NICE
  volumes:
    - ./suricata/logs:/var/log/suricata
    - ./suricata/rules:/var/lib/suricata/rules
    -
./suricata/suricata.yaml:/etc/suricata/suricata.yaml
  command: -i eth0
  tty: true
  stdin_open: true
  restart: unless-stopped
```

13.2.3 Operational Technology Environment

Separated the operational technology (OT) from the internet and added Jump Box, allowing engineers to interact with the system in a secure way.

```
bottle-filling_plc:
  image: paradoxxs/virtuaplant:plc
  container_name: bottle-filling_plc
  ports:
    - 5020:5020
    - 3001:3000
  networks:
    - Production
bottle-filling_hmi:
  image: paradoxxs/virtuaplant:hmi
  container_name: bottle-filling_hmi
  environment:
    - PLC_SERVER_IP=bottle-filling_plc
    - PLC_SERVER_PORT=5020
  ports:
    - 3002:3000
  depends_on:
    - bottle-filling_plc
  networks:
    - Production
Engineer_jumpbox:
  image: lscr.io/linuxserver/chromium:latest
  container_name: Engineer_jumpbox
  environment:
    - PUID=1000
    - PGID=1000
    - TZ=Etc/UTC
    - TITLE=Jumpbox
    - CHROME_CLI=http://bottle-filling_hmi:3002
#optional
  ports:
    - 3003:3000 # http
    - 3004:3001 # https
  shm_size: "1gb"
  restart: unless-stopped
networks:
  - Production
  - corporation
```

13.2.4 Added Endpoint Detection and Response

Added endpoint detection and response (EDR) Velociraptor to the environment, including a client for monitoring.

```
velociraptor:
  container_name: velociraptor
  image: wlambert/velociraptor
  volumes:
    - ./velociraptor:/velociraptor/:rw
  environment:
    - VELOX_USER=${VELOX_USER}
    - VELOX_PASSWORD=${VELOX_PASSWORD}
    - VELOX_ROLE=${VELOX_ROLE}
    - VELOX_SERVER_URL=${VELOX_SERVER_URL}
    -
VELOX_FRONTEND_HOSTNAME=${VELOX_FRONTEND_HOSTNAME}
  ports:
    - "8000:8000"
    - "8001:8001"
    - "8889:8889"
  restart: unless-stopped
  networks:
    - corporation
    - dmz
    - Production
    - security
  healthcheck:
    test:
      [
        "CMD-SHELL",
        "curl -s -I http://localhost:8889",
      ]
    interval: 10s
    timeout: 10s
    retries: 120
velo_client:
  container_name: velo_client
  depends_on:
    velociraptor:
      condition: service_healthy
  image: ubuntu
  tty: true
  volumes:
    - ./velociraptor:/velociraptor/:ro
  networks:
    - corporation
  entrypoint:
```

```
["/velociraptor/clients/linux/velociraptor_client_repac
ked", "--config=/velociraptor/client.config.yaml",
"client", "-v"]
```

13.2.5 Added Security Information and Event Management Using Elasticsearch, Logstash, and Kibana Stack

Added security information and event management (SIEM) using Elasticsearch, Logstash, and Kibana (ELK) stack. In this setup, we used Logstash to take the logs from the Suricata and send them to Elasticsearch.

```
logstash:
  container_name: logstash
  depends_on:
    - elasticsearch
    - kibana
  image: logstash:8.14.1
  labels:
    co.elastic.logs/module: logstash
  user: root
  volumes:
    - "./suricata/logs:/var/log/suricata"
    -
"./logstash.conf:/usr/share/logstash/pipeline/logstash.
conf:ro"
  environment:
    - xpack.monitoring.enabled=false
    - ELASTIC_HOSTS=https://es-container:9200
  networks:
    - security
elasticsearch:
  container_name: es-container
  image:
docker.elastic.co/elasticsearch/elasticsearch:8.14.1
  environment:
    - xpack.security.enabled=false
    - "discovery.type=single-node"
  ports:
    - 9200:9200
  networks:
    - security
kibana:
  container_name: kb-container
  image: docker.elastic.co/kibana/kibana:8.14.1
  environment:
    - ELASTICSEARCH_HOSTS=http://es-container:9200
  depends_on:
    - elasticsearch
  ports:
    - 5601:5601
  networks:
    - security
```

13.2.6 Vulnerability Management

Added Nessus vulnerability scanner to identify potential vulnerabilities in our environment, while allowing it to monitor the entire environment.

```
nessus:
  image: tenable/nessus:latest-ubuntu
  container_name: nessus
  restart: always
  ports:
    - 8834:8834
  networks:
    - corporation
    - Production
    - dmz
```

Web Application Firewall: Added WAF proxy to help protect web application from attacks.

```
waf:
  image: owasp/modsecurity-crs:4.3.0-nginx-202406090906
  container_name: waf
  environment:
    - PARANOIA=4
    - BACKEND=http://juice:3000
  ports:
    - 9090:8080
  networks:
    - dmz
juice:
  image: bkimminich/juice-shop
  container_name: webapp
  ports:
    - 3000:3000
  networks:
    - dmz
```

Throughout our journey, we have implemented many additional security measures. This might seem very easy when working on such a simple network, but implementing these in a real-world environment can be quite challenging. I would also like to add that many of these controls require people to operate them. Something like SIEM, if that should be operated 24/7, would need between 10 and 12 people. With every new control, we also have to be thinking about who is going to be operating these systems.

The transition from a flat network to a Zero Trust Network is a significant step towards enhancing the cybersecurity posture. This transition requires a shift in mindset and the implementation of robust security measures. We must verify every access request, minimise privileges and operate under the assumption that breaches will occur. While we've made significant strides towards a Zero Trust Architecture, it's important to recognise that Zero Trust is a continuous journey of improvement. One key initiative we have yet to do is verifying the effectiveness of our security measures. It's insufficient to simply 'trust' that the project we have implemented will adequately protect the organisation. A third-party penetration testing company often performs this verification by testing the environment for potential weaknesses, similar to those a real threat actor might exploit. To finish this comparison, let's look at the current network diagram of Juice Factory, shown in Figure 13.1.

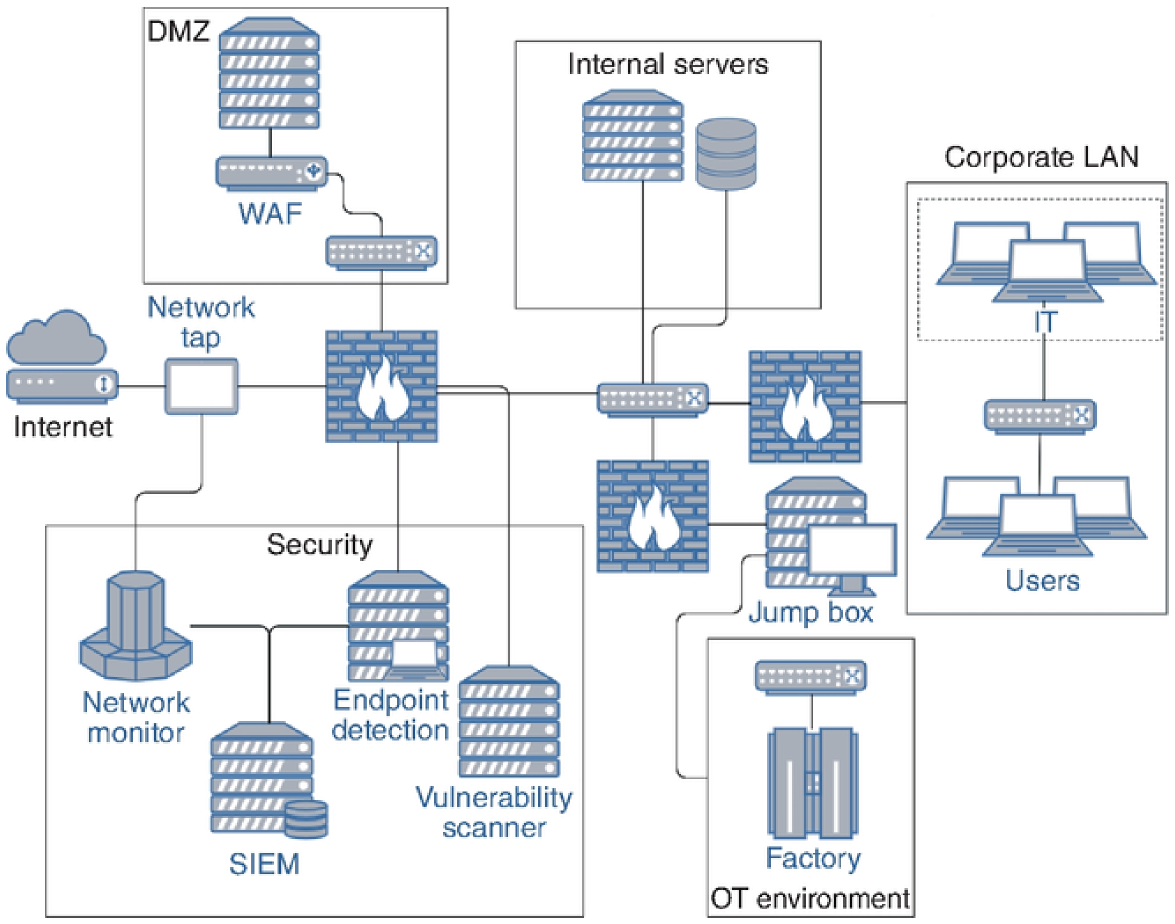


Figure 13.1 **Network diagram.**

As we added different components to the network.

The transaction map shown in Table 13.1 gives us a better idea of which services running in our environment are interacting with each other. Here are some other initiatives to consider:

- Regular Security Audits and Assessments: Continuously evaluate the effectiveness of security measures and identify areas for improvement through regular security audits and assessments.
- Advanced Threat Detection: Implemented advanced threat detection technologies, such as machine learning and artificial intelligence, to identify and respond to emerging threats more effectively.
- User and Entity Behaviour Analytics (UEBA): Utilise UEBA tools to detect anomalies in user and entity behaviour, which could indicate a potential security breach.

- Micro-segmentation: Further segment the network at a granular level to isolate workloads and limit the attack surface even more.
- Securing the OT Environment: One of the things that I did not talk about throughout this book is how to secure the OT environment, such as programmable logic controller (PLC) and HMI, and the challenges that it brings. This is something you have to pursue on your own.

TABLE 13.1

Transaction map.

	CRM	IAM	OT	Physical security	Web store	Backup	Wireless	Email	Internet	Network monitoring	Jumpbox	EDR	SIEM	Vulnerability management
Customer relation management (CRM)	■				■	■								■
IAM	■	■	■	■		■	■	■						■
OT			■					■			■			■
Physical security		■		■										■
Web store					■	■			■					■
Backup					■	■		■						■
Wireless		■					■		■					■
Email		■			■	■	■	■	■					■
Internet					■		■	■	■					■
Network monitoring										■				■
Jump Box			■								■			■
EDR	■		■		■							■		■
SIEM			■					■		■		■	■	■
Vulnerability management	■	■	■	■	■	■	■	■	■	■	■	■	■	■

Zero Trust is not a destination but a continual process of improvement. As threats evolve, so must our strategies and technologies. By adhering to Zero Trust principles and remaining vigilant, we can build a resilient security framework that protects our assets, data, and users from the ever-growing landscape of cyber threats. This book has hopefully provided you with the foundational knowledge and practical steps to embark on your own Zero Trust journey. Remember, the path to Zero Trust is ongoing, and the key to success lies in continual adaptation and improvement.

> Zero Trust is about tightening the security controls as much as possible, while still having the business processes running smoothly. Constantly look for new ways to remove trust. Remember that the most important part of any security program is the people and procedures.

Notes

1 https://cloudsecurityalliance.org/

2 https://learn.microsoft.com/en-us/azure/well-architected/

Glossary

Command and Control (C2)
Server used by threat actors to remotely control the devices they have compromised.

Containerisation
The process of packaging an application and its dependencies into a single, lightweight container that can run consistently across different computing environments, improving portability and scalability.

Cross-site Scripting (XSS)
A security vulnerability that allows attackers to inject malicious scripts into web pages viewed by other users, potentially stealing sensitive information or performing actions on behalf of the user.

Docker
A platform that enables developers to create, deploy and manage applications in lightweight, portable containers, allowing for consistent environments across different systems and simplifying application deployment.

Endpoint Detection and Response (EDR)
A security solution that monitors and responds to threats on endpoints, such as computers and servers, by detecting suspicious activities and providing tools for investigation and remediation.

Identity Access Management (IAM)
A framework of policies and technologies that ensures the right individuals have appropriate access to technology resources, managing user identities and their permissions within an organisation.

Jump Box
A secure intermediary server that provides controlled access to other servers or networks, often used to enhance security by limiting direct access to sensitive systems.

Network Segmentation

Dividing a network into smaller, isolated segments to improve security and performance, limiting the spread of threats and controlling access to resources.

Proxy

A server that acts as an intermediary between a client and another server, forwarding requests and responses while providing additional security, anonymity, or caching capabilities.

Security Information and Event Management (SIEM)

A system that collects, analyses and correlates security-related data from various sources to provide real-time visibility into an organisation's security posture and facilitate incident response.

SQL Injection

A code injection technique that exploits vulnerabilities in web applications by inserting malicious SQL queries into input fields, allowing attackers to manipulate databases and gain unauthorised access to data.

Threat Intelligence

Gathering and analysing information about potential threats to an organisation's security, including vulnerabilities, attack vectors and threat actors.

Vulnerability Scanner

A tool that automatically scans systems and applications for known vulnerabilities, helping organisations identify and remediate security weaknesses before they can be exploited by attackers.

Web Application Firewall (WAF)

A security solution that monitors and filters HTTP traffic to and from web applications, protecting them from common threats such as SQL injection, cross-site scripting (XSS) and other web-based attacks.

Zero Trust

A security model that assumes no trust by default, requiring strict verification for every user and device attempting to access resources, regardless of their location within or outside the network.

Index