CHAPTER 1
INTRODUCTION

Cybersecurity threats continue to evolve at an alarming rate, posing significant challenges to organizations worldwide. In today's digital landscape, the protection of sensitive data and critical infrastructure against malicious actors is of paramount importance. Real-time monitoring of endpoints for intrusion detection and response with threat intelligence has emerged as a crucial strategy in mitigating these threats. This section provides an introduction to the project, outlining its objectives and significance in addressing the pressing cybersecurity concerns faced by organizations today.

1.1 OVERVIEW

The project centers on the development and implementation of a robust system for real-time monitoring of endpoints to detect and respond to security threats effectively. In the modern digital landscape, where cybersecurity threats are constantly evolving, the need for proactive measures to safeguard sensitive data and critical infrastructure is paramount. This project addresses this imperative by leveraging advanced technologies and methodologies to create a comprehensive solution.

1.2 OBJECTIVES

- Develop a real-time monitoring system tailored for endpoint security.
- Enhance detection capabilities to identify potential security breaches promptly.
- Enable swift and automated responses to detected threats to minimize the impact of security incidents.
- Integrate external threat intelligence feeds to enhance the system's ability to detect emerging threats proactively.

1.3 SIGNIFICANCE

- Addressing critical cybersecurity challenges faced by organizations, including the detection and mitigation of cyber threats.
- Providing organizations with the necessary tools and capabilities to detect, respond to, and mitigate security incidents in real-time.
- Enhancing organizations' resilience against evolving cyber threats by leveraging advanced technologies and threat intelligence.
- Safeguarding sensitive data and critical infrastructure from malicious actors and unauthorized access.

1.4 SCOPE

The scope of the project encompasses the design, development, and evaluation of a real-time monitoring system for endpoint security. Key aspects include:

- Data collection from endpoints, network devices, and other sources.
- Analysis of collected data to detect anomalous behavior and potential security threats.
- Response mechanisms to mitigate detected threats swiftly and effectively.
- Integration of external threat intelligence feeds to enrich security event data and enhance threat detection capabilities.
- Considerations for scalability, performance, and usability to ensure the effectiveness and practicality of the solution.

CHAPTER 2
LITERATURE SURVEY

In the rapidly evolving landscape of cybersecurity, staying abreast of existing systems and technologies is essential for developing effective strategies to defend against threats. A comprehensive literature survey provides valuable insights into the state-of-the-art in real-time monitoring, endpoint change detection, and threat intelligence. By examining established systems and methodologies, as well as exploring emerging trends and innovations, organizations can gain a deeper understanding of the challenges and opportunities in cybersecurity.

This section delves into existing systems and technologies relevant to the project's objectives, with a focus on real-time monitoring, endpoint change detection, and threat intelligence. By synthesizing findings from previous research and industry practices, this literature survey aims to inform the design and implementation of the project's solution. Additionally, the survey identifies older technologies that were once prevalent but have since been surpassed by newer, more advanced solutions, providing valuable context for evaluating the project's technological choices.

2.1 EXISTING SYSTEMS IN REAL-TIME MONITORING

Existing systems for real-time monitoring encompass a wide range of technologies and approaches. Common solutions include Security Information and Event Management (SIEM) systems, Network Intrusion Detection Systems (NIDS), and Endpoint Detection and Response (EDR) platforms. These systems continuously monitor network traffic, system logs, and endpoint activities to identify suspicious behavior and potential security incidents.

2.2 EXISTING SYSTEMS IN ENDPOINT CHANGE DETECTION

Endpoint change detection systems focus on monitoring changes to system files, configurations, and registry settings. They help detect unauthorized modifications that may indicate a security breach or compromise. Common techniques employed include file integrity monitoring (FIM), registry monitoring, and configuration management solutions.

2.3 EXISTING SYSTEMS IN THREAT INTELLIGENCE

Threat intelligence platforms gather, analyze, and disseminate information about cybersecurity threats and vulnerabilities. They provide insights into the tactics, techniques, and procedures (TTPs) used by threat actors, enabling organizations to proactively defend against emerging threats. These platforms collect data from various sources, including open-source feeds, commercial threat intelligence providers, and internal security telemetry.

2.4 MAJOR DRAWBACKS

Despite their benefits, existing systems in real-time monitoring, endpoint change detection, and threat intelligence have several limitations. These may include high false positive rates, limited scalability, and inadequate integration capabilities. Additionally, many systems rely on manual analysis and lack automation, leading to delays in threat detection and response.

CHAPTER 3
SYSTEM ENVIRONMENT

3.1 SYSTEM ARCHITECTURE

The system architecture serves as the blueprint for the project's design and implementation, outlining the various components and their interactions. It encompasses the integration of the chosen technologies to create a cohesive and efficient real-time monitoring system for endpoint security. The system architecture of the proposed solution is built upon a distributed and scalable framework that leverages microservices architecture to ensure modularity, flexibility, and resilience. The architecture comprises interconnected components that work cohesively to collect, process, analyze, and respond to security events and incidents in real-time. Key architectural elements include:

- Centralized Management Server: Acts as the command center for the entire cybersecurity infrastructure, orchestrating the deployment and configuration of security components, managing security policies, and facilitating centralized logging and reporting.

- Distributed Sensor Nodes: Deployed across the network, sensor nodes collect security event data from various endpoints, network devices, and cloud environments. These lightweight agents transmit telemetry data to the central management server for analysis and correlation.

- Scalable Data Storage Layer: Utilizes distributed database technologies such as Cassandra to store and manage large volumes of security event data efficiently. The scalable data storage layer ensures high availability and reliability, enabling organizations to retain and query security logs over extended periods.

- Advanced Analytics Engine: Employs machine learning algorithms, behavioral analytics, and threat intelligence feeds to analyze security event data in real-time. The analytics engine correlates disparate data sources, identifies patterns indicative of malicious activity, and prioritizes security incidents for further investigation and response.

- Incident Response Orchestration Platform: Automates incident response workflows, enabling security teams to coordinate response efforts, execute remediation actions, and track the progress of incident resolution. The orchestration platform integrates with ticketing systems, collaboration tools, and external threat intelligence platforms to streamline incident management processes.

- Threat Intelligence Integration Layer: Facilitates the integration of external threat intelligence feeds from sources such as Misp, enabling organizations to enrich security event data with contextual information about known threats, vulnerabilities, and indicators of compromise (IOCs).

3.1.1 COMPONENT OVERVIEW

This subsection provides an overview of the key components of the system architecture, including Wazuh, Hive, Cortex, MISP, Cassandra, Minio, and Shuffle. Each component's role and functionality within the system are described, highlighting how they contribute to the overall effectiveness of the solution.

3.1.2 HIGH-LEVEL DIAGRAM

A high-level architectural diagram illustrates the relationships and dependencies between different system components. This diagram visually represents the flow of data and control within the system, helping stakeholders understand the system's structure and operation at a glance.

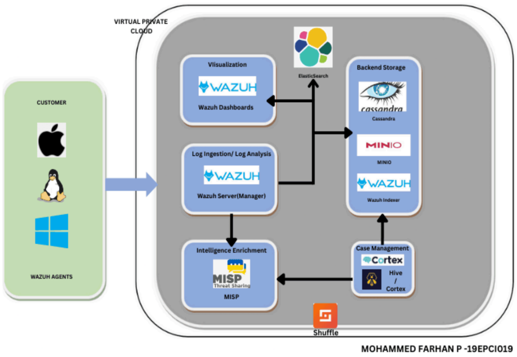


Figure 3.1: The Diagrammatic Representation of the Physical Infrastructure of the System

3.2 DATA COLLECTION

Data collection is a critical aspect of the system environment, involving the ingestion of log data from various sources to facilitate analysis and detection. This subsection explores the mechanisms and strategies employed for collecting data from endpoints, network devices, and other relevant sources.

Log ingestion mechanisms play a crucial role in the system environment, enabling the collection, normalization, and enrichment of security event data from diverse sources. The log ingestion process involves the following steps:

- Data Collection: Sensor nodes deployed throughout the network collect security event data from endpoints, servers, network devices, and cloud environments. Data collection agents may include host-based agents, network sensors, and log forwarders configured to capture relevant telemetry data.

- Normalization: Raw log data collected from disparate sources is normalized to a standardized format for consistency and ease of analysis. Normalization involves parsing log entries, extracting relevant fields, and mapping them to a common schema to facilitate correlation and analysis.

- Enrichment: Enrichment processes augment raw log data with additional contextual information from external sources such as threat intelligence feeds, vulnerability databases, and asset inventories. Enriched data provides valuable context for understanding the significance and severity of security events, enabling more informed decision-making and response prioritization.

3.2.1 ENDPOINT AGENTS

Wazuh agents are deployed on endpoints to capture and monitor security-relevant events, such as process executions, file modifications, and network connections. These agents collect and forward data to the centralized monitoring platform for analysis and correlation, providing real-time visibility into endpoint activities.

3.2.2 NETWORK SENSORS

Network sensors passively monitor network traffic for signs of suspicious activity, such as intrusion attempts or unauthorized access. These sensors analyze network packets in real-time, identifying anomalies and potential security threats that may require further investigation or response.

3.3 DATA STORAGE AND MANAGEMENT

Data storage and management are essential components of the system environment, ensuring the reliable storage and retrieval of security logs and telemetry data for analysis and reporting. This subsection delves into the technologies and strategies utilized for storing and managing large volumes of security data.

3.3.1 DISTRIBUTED DATABASE

Cassandra is employed as the distributed database for storing security event data at scale. Its distributed architecture and tunable consistency levels ensure high availability and fault tolerance, making it suitable for handling large volumes of security logs and telemetry data.

3.3.2 OBJECT STORAGE

Minio serves as the object storage solution for long-term retention of security data and logs. Its scalable and cost-effective storage infrastructure enables organizations to store and archive security data for compliance and forensic purposes, ensuring data integrity and accessibility over time.

3.4 DATA PROCESSING AND ANALYSIS

Data processing and analysis are integral components of the system environment, enabling organizations to derive actionable insights from security data and telemetry. This subsection explores the technologies and methodologies employed for processing and analyzing security events in real-time.

3.4.1 THREAT INTELLIGENCE INTEGRATION

MISP serves as the central repository for aggregating and correlating threat intelligence feeds and indicators of compromise (IOCs). By integrating external threat intelligence sources with internal security telemetry, organizations can enrich security events with contextual information, enabling proactive threat detection and response.

3.4.2 WORKFLOW ORCHESTRATION

Shuffle facilitates workflow orchestration and automation, enabling organizations to streamline incident response processes and threat hunting activities. Its intuitive interface and workflow management capabilities empower security analysts to collaborate effectively and respond rapidly to security incidents and alerts.

CHAPTER 4
OVERVIEW

4.1 INTRODUCTION

The overview section serves as a foundational introduction to the project, providing context and setting the stage for the subsequent sections of the report. It aims to articulate the project's objectives, scope, and significance, establishing a clear understanding of its relevance and importance in the realm of cybersecurity.

4.2 VIEWS

In considering multiple viewpoints, the project ensures a comprehensive understanding of stakeholders' needs and requirements. These viewpoints encompass perspectives from various roles within the organization, including system administrators, security analysts, and end-users. By soliciting input from diverse stakeholders, the project endeavors to align its design and implementation with the overarching goals and priorities of the organization.

- User Viewpoints: Understanding the needs and preferences of end-users, including security analysts, incident responders, and system administrators, to design user-friendly interfaces and intuitive workflows that enhance usability and productivity.

- System Administrators' Perspectives: Considering the operational requirements and challenges faced by system administrators in managing and maintaining the cybersecurity infrastructure, including scalability, performance, and compliance management.

4.3 SCOPE

The scope of the project delineates the boundaries and objectives within which the proposed solution operates. It encompasses the design, implementation, and evaluation of a real-time monitoring system for endpoint security, with a focus on intrusion detection and response with threat intelligence. The scope also includes the selection and configuration of appropriate technologies, the development of detection and response mechanisms, and the establishment of operational processes and procedures.

- Deliverables: The deliverables of the project include the development and implementation of the proposed solution, documentation of system architecture and functionalities, training materials for end-users and system administrators, and ongoing support and maintenance services.

- Limitations: While the proposed solution aims to address key cybersecurity challenges, it may have certain limitations in terms of resource constraints, technical feasibility, and organizational constraints. These limitations will be carefully considered and managed throughout the project lifecycle to ensure successful implementation and adoption.

4.4 SIGNIFICANCE OF THE PROJECT

The significance of the project lies in its potential to address critical cybersecurity challenges faced by organizations today. In an increasingly interconnected and digitized world, the threat landscape continues to evolve, with adversaries employing sophisticated tactics and techniques to exploit vulnerabilities and compromise systems. By implementing a robust real-time monitoring system for endpoint security, the project aims to mitigate these risks and enhance organizations' resilience to cyber threats. Through proactive detection, rapid response, and effective threat intelligence integration, the project seeks to empower organizations to safeguard their critical assets, protect sensitive data, and maintain the trust and confidence of their stakeholders.

- Strengthen Cyber Resilience: By enhancing threat detection and incident response capabilities, organizations can improve their ability to identify and mitigate cyber threats, reducing the risk of data breaches and system compromises.

- Enhance Compliance and Regulatory Compliance: The Phase 2 proposed solution helps organizations achieve compliance with industry regulations and standards, such as GDPR, HIPAA, and PCI DSS, by implementing robust security controls and ensuring data protection and privacy.

- Protect Critical Assets and Intellectual Property: By safeguarding critical assets, sensitive data, and intellectual property from unauthorized access and exploitation, organizations can minimize the risk of financial loss, reputational damage, and legal liabilities.

CHAPTER 5
COMPARISON OF EXISTING AND PROPOSED SOLUTION

5.1 OVERVIEW OF EXISTING SYSTEM

In the initial phase of the project, Suricata, Zeek, and OWLH were considered as potential components for the real-time monitoring system. However, after careful evaluation and analysis, these technologies were eliminated from the proposed solution.

5.1.1 SURICATA

Suricata, a widely used open-source Intrusion Detection System (IDS), offers powerful network traffic analysis capabilities. Despite its strengths, Suricata primarily focuses on network-level monitoring and lacks comprehensive visibility into endpoint activities. As a result, it was deemed unsuitable for fulfilling the project's requirement of holistic endpoint monitoring for intrusion detection and response.

5.1.2 ZEEK

Zeek, formerly known as Bro, is another popular open-source network analysis framework known for its protocol analysis and network traffic inspection capabilities. While Zeek provides valuable insights into network-level events and anomalies, it also lacks the granular visibility into endpoint activities required for comprehensive threat detection. Therefore, it was deemed inadequate for meeting the project's objectives of real-time monitoring and response at the endpoint level.

5.1.3 OWLH

OWLH is an open-source honeypot framework designed to detect and analyze attacks targeting wireless networks. Although OWLH offers capabilities for monitoring and analyzing wireless network traffic, it does not align with the project's focus on endpoint security and threat detection. As such, it was determined to be outside the scope of the project's requirements and was eliminated from consideration.

5.2 DISADVANTAGES OF THE EXISTING SYSTEM

The decision to eliminate Suricata and Zeek stemmed from their inherent limitations in addressing the evolving landscape of cybersecurity threats. While proficient in analyzing network traffic patterns for signs of intrusion, these solutions lacked the granular visibility into endpoint activities and changes crucial for detecting sophisticated attacks and insider threats. Moreover, their reliance primarily on network-centric data limited their effectiveness in detecting lateral movement and targeted attacks that exploit vulnerabilities at the endpoint level.

- Inadequate Threat Intelligence Integration

  While Suricata and Zeek excel in analyzing network traffic patterns and identifying potential security incidents, they have limited capabilities for integrating external threat intelligence feeds and contextual information. This limitation hinders their ability to correlate security events with known threats, vulnerabilities, and indicators of compromise (IOCs), resulting in higher false positive rates and reduced effectiveness in threat detection and response.

- Lack of Automation and Orchestration

  Suricata and Zeek rely primarily on manual analysis and configuration, requiring significant human intervention for incident detection, analysis, and response. This manual approach is time-consuming and resource-intensive, making it challenging for security teams to keep pace with the rapidly evolving threat landscape and respond effectively to security incidents in real-time. Additionally, the lack of automation and orchestration capabilities limits their scalability and agility in adapting to changing security requirements and operational needs.

- Scalability and Performance Constraints
  Suricata and Zeek may face scalability and performance challenges when deployed in large-scale enterprise environments with high volumes of network traffic and diverse infrastructure. Their architecture and processing methodologies may struggle to handle the sheer volume of data generated by modern networks, leading to performance degradation, resource contention, and operational inefficiencies. This scalability limitation hampers their ability to scale seamlessly and accommodate the growing demands of the organization without compromising performance or reliability.

- Complexity and Maintenance Overhead
  The complexity of deploying, configuring, and maintaining Suricata and Zeek in production environments can pose significant challenges for organizations with limited resources and expertise in cybersecurity. Managing rule sets, updating signatures, and fine-tuning detection policies require specialized knowledge and skills, increasing the maintenance overhead and operational burden on security teams. This complexity may deter organizations from adopting Suricata and Zeek as core components of their cybersecurity infrastructure, especially when more user-friendly and integrated solutions are available in the market.

5.3 OVERVIEW OF PROPOSED SYSTEM

The Phase 2 proposed solution introduces a curated tech stack comprising Wazuh, Hive, Cortex, MISP, Cassandra, Minio, and Shuffle, strategically chosen to overcome the shortcomings of Suricata and Zeek while enhancing the system's capabilities in real-time monitoring, threat detection, and incident response.

Wazuh stands out as a versatile endpoint detection and response (EDR) platform that offers comprehensive visibility into endpoint activities, including process execution, file integrity monitoring, and registry changes. Its agent-based approach ensures continuous monitoring and timely detection of suspicious behavior, empowering organizations to swiftly respond to potential security incidents.

Hive and Cortex complement Wazuh by providing automated incident response capabilities and streamlined workflow orchestration. Leveraging the power of playbooks and case management, these platforms enable security teams to automate routine tasks, accelerate incident triage, and orchestrate responses to security events effectively.

MISP serves as a central hub for threat intelligence aggregation, correlation, and sharing, facilitating the integration of external feeds and indicators of compromise (IOCs) into the detection and response workflow. By enriching security events with contextual information and historical data, MISP enhances the system's ability to prioritize alerts and identify emerging threats proactively.

Cassandra and Minio address the challenges of data storage and scalability, offering robust solutions for storing and querying large volumes of security data. With their distributed architecture and horizontal scalability, these platforms ensure high availability and performance, enabling organizations to retain and analyze security logs effectively over time.

Shuffle plays a pivotal role in streamlining data processing and analysis workflows, enabling efficient threat hunting, investigation, and reporting. By providing a unified interface for data manipulation and visualization, Shuffle empowers security analysts to derive actionable insights from disparate sources of security telemetry, enhancing situational awareness and decision-making.

5.4 ADVANTAGES OF THE PROPOSED SOLUTION

The Phase 2 proposed system offers a multitude of advantages over the Phase 1 solution, positioning organizations to effectively combat the evolving threat landscape and safeguard critical assets.

- Automated Incident Response: Hive and Cortex automate incident response workflows, enabling security teams to respond rapidly to security incidents and minimize the impact of breaches.

- Integrated Threat Intelligence: Misp facilitates the integration of external threat intelligence feeds, enriching security events with contextual information and enabling proactive threat hunting and analysis.

- Scalability and Performance: Cassandra and Minio ensure scalability and high availability for storing and querying large volumes of security data, empowering organizations to retain and analyze logs effectively over time.

- Streamlined Workflow Orchestration: Shuffle simplifies data processing and analysis workflows, enabling security analysts to derive actionable insights and make informed decisions efficiently.

5.5 COMPARISON

- Advanced Threat Detection
    - Existing Solution

      The existing system employs traditional methods of threat detection, relying on signature-based detection and basic anomaly detection techniques. While these methods may detect known threats, they often struggle to identify sophisticated or emerging threats, resulting in higher false positive rates and delayed response times.

    - Proposed Solution

      The Phase 2 proposed solution incorporates Hive and Cortex, advanced incident response and threat intelligence platforms. Hive and Cortex leverage machine learning algorithms and behavioral analytics to identify and prioritize security incidents accurately. By automating incident analysis and response, these platforms enable organizations to detect and mitigate security threats rapidly and effectively, reducing the risk of data breaches and minimizing the impact of security incidents.

    - New Advantage

      Proactive Threat Hunting: The Phase 2 solution empowers organizations to proactively hunt for threats using advanced analytics and threat intelligence, enabling the identification of potential security risks before they escalate into full-blown incidents.

- Seamless Threat Intelligence Integration
    - Existing Solution

      The existing system lacks seamless integration with external threat intelligence feeds, relying primarily on internal security telemetry for threat detection and analysis. This limited visibility into external threats may result in missed opportunities to detect and respond to emerging threats effectively.

    - Proposed Solution

      In contrast, the Phase 2 proposed solution integrates Misp, a comprehensive threat intelligence platform. Misp serves as a central repository for aggregating, correlating, and sharing threat intelligence feeds from external sources. By enriching security event data with contextual information from Misp, the proposed solution enhances threat detection capabilities and enables proactive defense against emerging threats, significantly improving the organization's cybersecurity posture.

    - New Advantage

      Contextual Threat Analysis: By integrating external threat intelligence feeds, the Phase 2 solution provides contextual information about threats, enabling security analysts to better understand the motivations and tactics of adversaries and tailor their response accordingly.

- Scalability and Performance Optimization

    - Existing Solution

      The existing system may face scalability and performance challenges, particularly as data volumes continue to grow. Legacy storage solutions and processing methodologies may struggle to handle large volumes of security data effectively, leading to performance degradation and operational inefficiencies.

    - Proposed Solution

      The Phase 2 proposed solution addresses scalability and performance concerns by leveraging Cassandra and Minio for data storage and management. These scalable storage solutions ensure high performance and reliability, even in the face of increasing data volumes. By optimizing storage and processing capabilities, the proposed solution enhances operational efficiency and enables organizations to retain and analyze security data effectively over time.

    - New Advantage

      Elastic Scalability: With Cassandra and Minio, the Phase 2 solution offers elastic scalability, allowing organizations to seamlessly scale their infrastructure to accommodate fluctuating workloads and data volumes without compromising performance.

- Streamlined Workflow Orchestration
    - Existing Solution

      The existing system may rely on manual processes for incident response and workflow orchestration, leading to delays and inefficiencies in incident handling. Disparate tools and workflows may hinder collaboration among security teams and impede the timely detection and response to security incidents.

  - Proposed Solution

    In contrast, the Phase 2 proposed solution streamlines workflow orchestration with Shuffle, a workflow management and automation platform. Shuffle provides a unified interface for data processing and analysis, enabling seamless collaboration among security teams and facilitating rapid incident response. By automating routine tasks and orchestrating workflows, Shuffle empowers security analysts to respond promptly and effectively to security incidents, reducing response times and minimizing the impact of security breaches.

  - New Advantage

    Adaptive Workflow Automation: With Shuffle, the Phase 2 solution offers adaptive workflow automation, allowing organizations to customize and automate incident response processes based on evolving security requirements and operational needs. This flexibility enhances agility and responsiveness in addressing emerging threats and security challenges.

- Enhanced User Experience
  - Existing Solution

    The existing system may lack user-friendly interfaces and intuitive workflows, leading to challenges in usability and adoption. Complex configurations and cumbersome processes may hinder the effectiveness of security operations teams and increase the risk of human error.

- Proposed Solution

  In contrast, the Phase 2 proposed solution prioritizes user experience, offering intuitive interfaces and streamlined workflows. User-centric design principles are incorporated into the development process, ensuring that security operations teams can easily navigate the system and perform their tasks efficiently. By enhancing usability and accessibility, the proposed solution promotes user engagement and empowers security professionals to focus on strategic initiatives rather than administrative overhead.

- New Advantage

  Intuitive Interface: The Phase 2 solution features an intuitive interface that simplifies complex security tasks, reducing the learning curve for new users and increasing productivity across the organization. Interactive dashboards and customizable views enable security analysts to access relevant information quickly and make informed decisions in real-time.

- Enhanced Compliance and Reporting Capabilities
  - Existing Solution

    The existing system may lack robust compliance and reporting capabilities, making it challenging for organizations to demonstrate adherence to regulatory requirements and industry standards. Manual processes for generating reports and tracking compliance may be time-consuming and error-prone, leading to compliance gaps and audit findings.

  - Proposed Solution

In contrast, the Phase 2 proposed solution offers enhanced compliance and reporting capabilities, enabling organizations to streamline compliance management and demonstrate regulatory compliance effectively. Built-in reporting templates and automated compliance checks simplify the process of generating compliance reports and tracking adherence to regulatory requirements. By centralizing compliance management and reporting functions, the proposed solution reduces the administrative burden on security teams and ensures consistency in compliance efforts.

  - New Advantage
    Automated Compliance Management: The Phase 2 solution automates compliance management tasks, such as policy enforcement, audit trail generation, and regulatory reporting, reducing the time and effort required to maintain compliance. Real-time monitoring and alerts notify security teams of compliance deviations, allowing for prompt remediation and continuous improvement of compliance posture.

- Enhanced Threat Intelligence Sharing and Collaboration
  - Existing Solution
    The existing system may lack robust mechanisms for threat intelligence sharing and collaboration, limiting the organization's ability to benefit from collective insights and expertise. Siloed information and communication barriers may hinder collaboration among internal teams and external partners, reducing the effectiveness of threat intelligence programs.

  - Proposed Solution

In contrast, the Phase 2 proposed solution enhances threat intelligence sharing and collaboration capabilities, fostering a culture of collective defense and information sharing. Integrated collaboration tools and secure communication channels facilitate the exchange of threat intelligence within the organization and with trusted external partners. By promoting information sharing and collaboration, the proposed solution enables organizations to leverage collective insights and expertise to identify and respond to security threats more effectively

- New Advantage
  Secure Information Exchange: The Phase 2 solution provides secure channels for sharing sensitive threat intelligence information, ensuring confidentiality and integrity throughout the information exchange process. Access controls and encryption mechanisms safeguard shared data from unauthorized access and interception, enhancing the trust and reliability of information sharing initiatives.

CHAPTER 6
SYSTEM DESIGN

6.1 OBJECTIVES

In this phase, the primary objectives encompass the establishment of a robust and comprehensive architecture that seamlessly integrates various components to facilitate virtualized real-time monitoring of endpoints for intrusion detection and response, bolstered by threat intelligence. The overarching goals include:

1. Implementing a distributed deployment of Wazuh to ensure pervasive endpoint monitoring across the network, thus fortifying the organization's security posture.

2. Orchestrating the amalgamation of multiple components such as Hive, Cortex, Misp, Minio, Shuffle, and Virustotal to enhance the organization's ability to detect, analyze, and respond to security threats in a proactive and efficient manner.

3. Ensuring scalability, adaptability, and resilience of the system to accommodate the dynamic nature of cyber threats and the evolving needs of the organization.

4. Establishing seamless integration and interoperability among the diverse set of components to facilitate coherent threat intelligence analysis and response workflows, thereby maximizing the efficacy of the security infrastructure.

6.2 WAZUH DISTRIBUTED INSTALLATION

Wazuh, as the cornerstone of the system, necessitates a distributed installation approach to effectively monitor endpoints and detect potential security breaches. This involves several key steps:

- Wazuh Agent Installation: Each endpoint within the network is equipped with a Wazuh agent, which is responsible for capturing security-relevant events and transmitting them to the centralized Wazuh server manager for analysis. The deployment of agents ensures comprehensive coverage of endpoint activities, including system logs, file integrity changes, and registry modifications.

- Wazuh Server Manager Installation: A centralized Wazuh server manager is deployed to aggregate, process, and analyze the data collected by the distributed agents. This server acts as a nerve center for security operations, providing real-time visibility into the security posture of the network and generating alerts for suspicious activities. Configuration of the server manager involves setting up detection rules, thresholds, and notification mechanisms to facilitate timely incident response.

- Wazuh Dashboard Installation: To empower security analysts with intuitive visualization and management capabilities, the Wazuh dashboard is installed alongside the server manager. This web-based interface offers comprehensive insights into security events, facilitates forensic analysis, and enables efficient incident response coordination. Customization of the dashboard allows tailoring it to the specific requirements and preferences of the security team.

6.3 HIVE

Hive serves as a pivotal component for incident response orchestration and collaboration, enabling security teams to effectively manage and mitigate security incidents. Its functionalities include:

- Incident Documentation and Tracking: Hive provides a centralized platform for documenting and tracking security incidents, allowing security analysts to record essential details such as incident type, severity, affected assets, and mitigation actions taken. This facilitates a structured and organized approach to incident management, ensuring accountability and transparency throughout the resolution process.

- Workflow Automation: Through predefined workflows and automation capabilities, Hive streamlines incident response processes, reducing manual effort and accelerating response times. Automated tasks such as ticket generation, stakeholder notifications, and evidence collection enhance operational efficiency and enable security teams to focus on higher-value activities.

- Collaborative Investigation: Hive fosters collaboration and knowledge sharing among security personnel by enabling real-time communication and collaboration within incident timelines. Security analysts can collaborate on investigation activities, share insights, and leverage collective expertise to expedite incident resolution and enhance the organization's overall security posture.

6.4 CORTEX

Cortex is an indispensable component for threat intelligence analysis and enrichment, empowering security teams to enhance their understanding of security events and make informed decisions. Its features include:

- Automated Enrichment: Cortex automates the enrichment of security events by integrating with various threat intelligence feeds, open-source tools, and external services. It enriches security alerts with contextual information such as threat actor profiles, malware analysis reports, and historical incident data, enabling analysts to assess the severity and credibility of security incidents more effectively.

- Analysis and Correlation: Cortex facilitates the analysis and correlation of security events by aggregating and correlating data from multiple sources. By correlating disparate security events and identifying patterns or anomalies, Cortex helps uncover hidden threats and prioritize incident response efforts based on the perceived risk and impact to the organization.

- Customization and Extensibility: Cortex offers customization and extensibility capabilities, allowing organizations to tailor the platform to their specific requirements and integrate with existing security infrastructure seamlessly. Custom analyzers, responders, and integrations can be developed and deployed to enhance the platform's functionality and address unique operational needs.

6.5 MISP

Misp, or Malware Information Sharing Platform & Threat Sharing, plays a crucial role in facilitating the collection, sharing, and analysis of threat intelligence data within the organization and across the broader security community. Its functionalities include:

- Threat Intelligence Sharing: Misp provides a collaborative environment for sharing indicators of compromise (IOCs), threat intelligence reports, and other relevant security information with trusted partners and peers. By participating in threat intelligence sharing communities, organizations can gain valuable insights into emerging threats and enhance their defensive capabilities.

- IOC Management and Analysis: Misp enables organizations to manage and analyze IOCs efficiently, facilitating the identification of potential security threats and proactive threat hunting activities. Security analysts can import, export, and correlate IOCs within the platform, allowing them to correlate disparate security events and uncover hidden connections or patterns indicative of malicious activity.

- Integration with External Sources: Misp integrates with external sources of threat intelligence, including commercial feeds, open-source databases, and proprietary threat intelligence platforms. This integration enables organizations to enrich their threat intelligence data with additional context and attribution, enhancing the accuracy and relevance of their analysis and response efforts.

6.6 MINIO

Minio serves as a high-performance, distributed object storage system that provides scalable and resilient storage infrastructure for security logs, threat intelligence data, and other security-related artifacts. Its functionalities include:

- Scalable Object Storage: Minio offers scalable object storage capabilities, allowing organizations to store and manage large volumes of security data efficiently. By leveraging distributed storage architecture and advanced caching mechanisms, Minio ensures high availability and low latency access to critical security artifacts.

- Data Protection and Security: Minio incorporates robust data protection and security features to safeguard sensitive information against unauthorized access, data breaches, and data loss incidents. Encryption at rest, encryption in transit, access control policies, and data integrity verification mechanisms are among the security measures implemented by Minio to ensure the confidentiality, integrity, and availability of stored data.

- Integration with Analytical Tools: Minio seamlessly integrates with analytical tools and data processing frameworks, enabling organizations to perform advanced analytics, machine learning, and data mining on their security data. By providing a unified data storage platform, Minio simplifies data access and analysis workflows, accelerating insights generation and decision-making processes.

6.7 SHUFFLE

Shuffle is an automation framework designed to streamline and automate repetitive security tasks and workflows, thereby enhancing operational efficiency and reducing manual effort. Its functionalities include:

- Task Automation: Shuffle automates a wide range of security tasks, including incident triage, alert investigation, threat hunting, and response orchestration. By defining predefined workflows and automation rules, Shuffle accelerates incident response times and minimizes human error, allowing security teams to focus on more strategic initiatives.

- Integration with Security Tools: Shuffle integrates seamlessly with existing security tools and technologies, including SIEMs, EDR solutions, threat intelligence platforms, and ticketing systems. This integration enables Shuffle to orchestrate end-to-end security workflows, from alert ingestion to incident resolution, by leveraging the capabilities of diverse security tools and maximizing their collective impact.

- Customization and Extensibility: Shuffle offers customization and extensibility features, allowing organizations to tailor the automation framework to their specific requirements and operational workflows. Custom plugins, scripts, and integrations can be developed and integrated into Shuffle, enabling organizations to address unique use cases and operational challenges effectively.

6.8 VIRUSTOTAL

Virustotal is a web-based service that aggregates and analyzes suspicious files and URLs to detect malware and other security threats. Its functionalities include:

- File and URL Analysis: Virustotal allows users to submit files and URLs for analysis, leveraging multiple antivirus engines and threat intelligence feeds to identify potential malware infections and security risks. The platform provides detailed reports on the analysis results, including detection rates, behavioral indicators, and associated threat intelligence.

- Threat Intelligence Sharing: Virustotal serves as a platform for sharing threat intelligence data with the broader security community, enabling organizations to contribute to and benefit from collective threat intelligence efforts. By sharing analysis results, IoCs, and malware samples, Virustotal facilitates collaboration and knowledge sharing among security practitioners, enhancing the collective defense against cyber threats.

- Integration with Security Tools: Virustotal integrates seamlessly with security tools and technologies, allowing organizations to incorporate threat intelligence data into their existing security workflows and processes. Integration with SIEMs, threat intelligence platforms, and endpoint security solutions enables organizations to enrich their security telemetry with Virustotal's analysis results, enhancing threat detection and response capabilities.

6.9 INTEGRATION OF SERVICES

The integration of Wazuh, Hive, Cortex, Misp, Minio, Shuffle, and Virustotal is orchestrated to create a cohesive and interoperable security ecosystem that maximizes the organization's ability to detect, analyze, and respond to security threats effectively. This integration involves:

- Configuring data exchange and communication protocols between different components to facilitate seamless information sharing and collaboration.

- Developing custom connectors, APIs, and integration points to enable interoperability and data interchange between disparate systems and technologies.

- Implementing event-driven workflows and automation rules to orchestrate end-to-end security processes, from alert ingestion to incident resolution.

- Establishing governance mechanisms and access controls to ensure the confidentiality, integrity, and availability of sensitive information shared between integrated components.

This integrated approach to security operations enables organizations to leverage the collective capabilities of diverse security technologies and tools, enhancing their ability to detect, analyze, and mitigate security threats in real-time. By fostering collaboration and interoperability among different components, organizations can achieve greater visibility into their security posture, improve incident response times, and ultimately, strengthen their overall cybersecurity defenses.

CHAPTER 7
OVERVIEW OF TECHNIQUES

7.1 DISTRIBUTED INSTALL OF WAZUH AND ITS COMPONENTS

7.1.1 WAZUH AGENT INSTALLATION

Wazuh agent installation is a critical step in the distributed deployment of Wazuh for endpoint monitoring. The process involves the following steps:

- Agent Deployment: Begin by deploying Wazuh agents to all endpoints within the network that require monitoring. This can be achieved using deployment scripts, group policy objects (GPOs), or manual installation procedures depending on the organization's preferences and requirements.

- Configuration: Once deployed, configure each Wazuh agent to communicate with the centralized Wazuh server manager. This involves specifying the IP address or hostname of the server manager, configuring authentication credentials, and defining communication protocols (e.g., TLS) to ensure secure communication between agents and the server manager.

- Registration: Register each Wazuh agent with the Wazuh server manager to establish a trusted relationship between the agent and the server manager. This involves generating and exchanging cryptographic keys to authenticate the agent and validate its identity during communication.

7.1.2 WAZUH SERVER MANAGER INSTALLATION

The installation of the Wazuh server manager is a pivotal step in centralizing and managing security events from distributed Wazuh agents. The process entails:

- Server Deployment: Deploy the Wazuh server manager on a dedicated server or virtual machine (VM) within the organization's network. Ensure that the server meets the minimum system requirements and has sufficient resources to handle the anticipated volume of security events.

- Software Installation: Install the Wazuh server manager software on the designated server, following the installation instructions provided by the vendor. This typically involves downloading the installation package, executing the installation script, and configuring basic parameters such as installation directory and server ports.

- Configuration: Configure the Wazuh server manager to communicate with Wazuh agents and ingest security events from distributed endpoints. This includes specifying listening ports, enabling encryption (e.g., TLS) for secure communication, and configuring data retention policies to manage storage resources effectively.

7.1.3 WAZUH DASHBOARD INSTALLATION

The installation of the Wazuh dashboard provides security analysts with a user-friendly interface for visualizing and managing security events. The process comprises:

- Dashboard Deployment: Deploy the Wazuh dashboard on a web server or application server within the organization's infrastructure. Ensure that the server meets the system requirements for hosting the dashboard and has access to the Wazuh server manager for data retrieval.

- Software Installation: Install the Wazuh dashboard software on the designated server, following the installation instructions provided by the vendor. This typically involves downloading the dashboard package, extracting the files, and configuring the web server to serve the dashboard application.

- Configuration: Configure the Wazuh dashboard to connect to the Wazuh server manager and retrieve security event data for visualization. This includes specifying the IP address or hostname of the server manager, configuring authentication settings, and defining user access permissions based on roles and responsibilities.

7.2 DOCKER DEPLOYMENT OF THREAT INTELLIGENCE STACK

7.2.1 HIVE

Hive, being a pivotal component for incident response, is deployed within a Docker environment to facilitate ease of deployment and scalability. The process entails:

- Docker Setup: Begin by setting up a Docker environment on a host system or a cluster of systems. Ensure that Docker Engine is installed and configured properly to enable containerized application deployment.

- Hive Container Deployment: Pull the Hive Docker image from the official repository or build a custom image based on specific requirements. Create a Docker container for Hive, specifying configuration parameters such as network settings, volume mounts, and environment variables.

- Configuration: Configure Hive within the Docker container to connect to external data sources, such as security incident feeds and threat intelligence platforms. Customize Hive settings as per organizational requirements, including user authentication, access controls, and integration with other security tools.

7.2.2 CORTEX

Cortex, being an integral component for threat intelligence analysis, is deployed using Docker containers to ensure portability and scalability. The deployment process involves:

- Container Orchestration: Utilize Docker Compose or Kubernetes to orchestrate the deployment of Cortex containers across multiple hosts or a cluster environment. Define service specifications, resource limits, and networking configurations to optimize performance and resource utilization.

- Image Selection: Choose the appropriate Cortex Docker image from the official repository or build a custom image tailored to specific requirements. Ensure that the image includes all necessary dependencies and configurations for seamless operation within the Docker environment.

- Configuration: Configure Cortex containers to interact with external data sources, such as threat intelligence feeds, malware analysis platforms, and enrichment services. Customize Cortex settings to enable automated enrichment, analysis, and correlation of security events, leveraging the full potential of the platform.

7.2.3 CASSANDRA

Cassandra, serving as the distributed database for storing threat intelligence data, is deployed using Docker containers to ensure scalability and fault tolerance. The deployment process includes:

- Containerization: Containerize Cassandra using Docker to encapsulate the database software and its dependencies within isolated environments. Use Docker images from official repositories or build custom images based on specific requirements and configurations.

- Cluster Deployment: Deploy Cassandra containers in a clustered configuration to distribute data across multiple nodes and ensure high availability and fault tolerance. Configure replication factors, consistency levels, and partitioning strategies to optimize data distribution and resilience.

- Configuration: Configure Cassandra containers with appropriate settings for data storage, indexing, and compaction. Fine-tune performance parameters such as cache sizes, read/write throughput, and compaction strategies to optimize database performance and resource utilization.

7.2.4 MISP

MISP, being a central platform for sharing and analyzing threat intelligence data, is deployed within Docker containers to simplify installation and management. The deployment process involves:

- Container Setup: Set up Docker on the host system or cluster environment where MISP will be deployed. Ensure that Docker Engine is installed and configured correctly to support containerized applications.

- MISP Container Deployment: Pull the MISP Docker image from the official repository or build a custom image based on specific requirements and configurations. Create Docker containers for MISP components, including the MISP application, database, and message broker.

- Configuration: Configure MISP containers with appropriate settings for database connectivity, message queuing, and integration with external services. Customize MISP configurations, such as organization details, user permissions, and data retention policies, to align with organizational requirements and security policies.

7.2.5 MINIO

Minio, serving as the distributed object storage system for storing security logs and threat intelligence data, is deployed using Docker containers to ensure scalability and resilience. The deployment process includes:

- Containerization: Containerize Minio using Docker to encapsulate the storage software and its dependencies within isolated environments. Utilize Docker images from official repositories or build custom images tailored to specific requirements and configurations.

- Cluster Deployment: Deploy Minio containers in a clustered configuration to distribute data across multiple nodes and ensure high availability and fault tolerance. Configure distributed storage policies, erasure coding settings, and access control mechanisms to optimize data resilience and security.

- Configuration: Configure Minio containers with appropriate settings for data storage, access control, and data retention. Customize Minio configurations, such as bucket policies, encryption settings, and storage classes, to meet organizational requirements and compliance standards.

7.2.6 ELASTICSEARCH

Elasticsearch, serving as the search and analytics engine for indexing and querying security event data, is deployed using Docker containers for scalability and performance. The deployment process involves:

- Container Orchestration: Use Docker Compose or Kubernetes to orchestrate the deployment of Elasticsearch containers across multiple hosts or a cluster environment. Define service specifications, resource limits, and networking configurations to optimize performance and resource utilization.

- Image Selection: Choose the appropriate Elasticsearch Docker image from official repositories or build custom images tailored to specific requirements and configurations. Ensure that the image includes all necessary dependencies and configurations for seamless operation within the Docker environment.

- Configuration: Configure Elasticsearch containers with appropriate settings for data indexing, storage, and querying. Fine-tune performance parameters such as heap size, shard allocation, and replica settings to optimize search and analytics performance for security event data.

7.3 DOCKER DEPLOYMENT OF AUTOMATION STACK

7.3.1 INSTALLATION AND CONFIGURATION OF SHUFFLE AUTOMATION

Shuffle, being an automation framework for streamlining security tasks, is deployed within Docker containers to facilitate ease of deployment and scalability. The installation and configuration process involve:

- Container Setup: Set up Docker on the host system or cluster environment where Shuffle will be deployed. Ensure that Docker Engine is installed and configured correctly to support containerized applications.

- Shuffle Container Deployment: Pull the Shuffle Docker image from the official repository or build a custom image based on specific requirements and configurations. Create Docker containers for Shuffle components, including the core engine, plugins, and integrations.

- Configuration: Configure Shuffle containers with appropriate settings for task automation, workflow orchestration, and integration with external services. Customize Shuffle configurations, such as task definitions, workflow templates, and notification settings, to align with organizational requirements and security policies.

7.4 INTEGRATION OF SERVICES

Integrating the various components of the system involves configuring data exchange, communication protocols, and interoperability mechanisms to facilitate seamless information sharing and collaboration. The integration process includes:

- Data Exchange Configuration: Configure data exchange mechanisms between different components, such as Wazuh, Hive, Cortex, MISP, Minio, Shuffle, and Virustotal. Define data formats, protocols, and endpoints for exchanging security events, threat intelligence data, and automation triggers.

- Communication Protocols: Establish communication protocols and standards for inter-component communication, ensuring compatibility and interoperability between diverse systems and technologies. Use industry-standard protocols such as RESTful APIs, message queuing systems, and event-driven architectures to facilitate seamless integration.

- Interoperability Mechanisms: Implement interoperability mechanisms, such as data normalization, transformation, and enrichment, to harmonize data formats and structures between integrated components. Ensure that data exchanged between different systems is consistent, accurate, and actionable, enabling effective analysis and response workflows.

- Governance and Access Controls: Define governance mechanisms and access controls to regulate access to shared data and resources between integrated components. Implement role-based access controls (RBAC), data encryption, and audit trails to ensure the confidentiality, integrity, and availability of sensitive information shared between different systems.

- Continuous Monitoring and Maintenance: Continuously monitor and maintain the integrated system to ensure smooth operation and timely response to security events. Monitor system performance, data flows, and security alerts to identify and address any issues or anomalies that may arise during operation. Perform regular maintenance tasks such as software updates, security patches, and configuration changes to keep the integrated system secure and up-to-date.

- Testing and Validation: Conduct thorough testing and validation of the integrated system to verify functionality, performance, and security. Test data exchange workflows, communication protocols, and interoperability mechanisms to ensure that integrated components work seamlessly together. Validate system behavior under various use cases, including normal operation, failure scenarios, and security incidents, to assess resilience and effectiveness.

- Documentation and Training: Document integration procedures, configuration settings, and operational workflows to facilitate system management and troubleshooting. Provide training and guidance to security personnel on how to use and maintain the integrated system effectively. Ensure that documentation is comprehensive, up-to-date, and accessible to all relevant stakeholders, enabling smooth operation and effective response to security incidents.

- Continuous Improvement: Continuously evaluate and improve the integration of system components based on feedback, lessons learned, and emerging security trends. Solicit input from security teams, stakeholders, and users to identify areas for enhancement and optimization. Implement iterative improvements, updates, and refinements to the integration architecture to adapt to evolving threats and organizational requirements.

CHAPTER 8
CONCLUSION AND FUTURE ENHANCEMENT

In conclusion, the virtualized real-time monitoring of endpoints for intrusion detection and response, integrated with threat intelligence, represents a significant advancement in enhancing the organization's security posture and resilience against cyber threats. Through the deployment and integration of various components such as Wazuh, Hive, Cortex, MISP, Minio, Shuffle, and Virustotal, the system has been able to achieve comprehensive visibility into security events, automate incident response workflows, and enrich threat intelligence analysis.

The deployment of Wazuh agents and server manager has enabled pervasive endpoint monitoring and real-time analysis of security events, facilitating proactive threat detection and response. The integration of Hive and Cortex has streamlined incident response processes, enabling efficient collaboration and coordination among security teams. MISP has served as a central repository for sharing and analyzing threat intelligence data, enhancing the organization's ability to identify and mitigate emerging threats.

Furthermore, the deployment of Minio has provided scalable and resilient storage infrastructure for storing security logs and threat intelligence data, ensuring data availability and integrity. The integration of Shuffle has enabled the automation of repetitive security tasks and workflows, increasing operational efficiency and reducing response times to security incidents. Virustotal has augmented threat intelligence analysis by providing comprehensive insights into potential malware infections and security risks.

Looking towards future enhancements, leveraging additional tools and services such as Velociraptor and AbuseIPDB could further strengthen the organization's security posture. Velociraptor, with its advanced endpoint monitoring capabilities, could enhance the detection and analysis of sophisticated threats, while AbuseIPDB could provide valuable insights into malicious IP addresses and domains used in phishing attacks.

Additionally, utilizing Shuffle for phishing email monitoring could help identify and mitigate phishing threats more effectively by automating the analysis of suspicious emails and extracting actionable intelligence for incident response. By continuously evolving and enhancing the system with new tools, technologies, and methodologies, the organization can stay ahead of evolving cyber threats and protect its assets more effectively in the dynamic threat landscape.

In essence, the virtualized real-time monitoring of endpoints for intrusion detection and response, coupled with threat intelligence integration, lays a solid foundation for robust cybersecurity defenses. By embracing innovation and continuously enhancing the system with new capabilities, the organization can adapt to emerging threats and ensure the security and integrity of its digital assets in an ever-changing cybersecurity landscape.

APPENDIX

A1 - SOURCE CODE

Install Procedure of Wazuh Indexer :

Install and configure the Wazuh indexer as a single-node or multi-node cluster following step-by-step instructions. Wazuh indexer is a highly scalable full-text search engine and offers advanced security, alerting, index management, deep performance analysis, and several other features.

The installation process is divided into three stages :

1. Certificates creation:
    a. curl -sO https://packages.wazuh.com/4.7/wazuh-certs-tool.sh
    b. curl -sO https://packages.wazuh.com/4.7/config.yml
    c. Run ./wazuh-certs-tool.sh to create the certificates. For a multi-node cluster, these certificates need to be later deployed to all Wazuh instances in your cluster.
       bash ./wazuh-certs-tool.sh -A
    d. Compress all the necessary files.
       tar -cvf ./wazuh-certificates.tar -C ./wazuh-certificates/ .
    e. rm -rf ./wazuh-certificates
    f. Copy the wazuh-certificates.tar file to all the nodes, including the Wazuh indexer, Wazuh server, and Wazuh dashboard nodes. This can be done by using the scp utility.

2. Nodes installation
    a. Installing package dependencies
    b. Install the following packages if missing:
       apt-get install debconf adduser procps
    c. Adding the Wazuh repository

d. Install the following packages if missing.

e. apt-get install gnupg apt-transport-https

f. Install the GPG key.

```
curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gpg
--no-default-keyring --keyring
gnupg-ring:/usr/share/keyrings/wazuh.gpg --import && chmod 644
/usr/share/keyrings/wazuh.gpg
```

g. Add the repository.

```
echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg]
https://packages.wazuh.com/4.x/apt/ stable main" | tee -a
/etc/apt/sources.list.d/wazuh.list
```

h. Update the package's information.

```
apt-get update
```

i. Installing the Wazuh indexer

j. Install the Wazuh indexer package.

```
apt-get -y install wazuh-indexer
```

3. Cluster initialization

a. Starting the service

b. Enable and start the Wazuh indexer service.

```
systemctl daemon-reload
systemctl enable wazuh-indexer
systemctl start wazuh-indexer
```

Install Procedure of Wazuh Server:

Install and configure the Wazuh server as a single-node or multi-node cluster following step-by-step instructions. The Wazuh server is a central component that includes the Wazuh manager and Filebeat. The Wazuh manager collects and analyzes data from the deployed Wazuh agents. It triggers alerts when threats or anomalies are detected. Filebeat securely forwards alerts and archived events to the Wazuh indexer.

The installation process :

1. Wazuh server node installation

   a. Adding the Wazuh repository

   b. Install the following packages if missing.

   ```
   apt-get install gnupg apt-transport-https
   ```

   c. Install the GPG key.

   ```
   curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gpg
   --no-default-keyring --keyring
   gnupg-ring:/usr/share/keyrings/wazuh.gpg --import && chmod 644
   /usr/share/keyrings/wazuh.gpg
   ```

   d. Add the repository.

   ```
   echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg]
   https://packages.wazuh.com/4.x/apt/ stable main" | tee -a
   /etc/apt/sources.list.d/wazuh.list
   ```

   e. Update the packages information.

   ```
   apt-get update
   ```

   f. Installing the Wazuh manager

   g. Install the Wazuh manager package.

   ```
   apt-get -y install wazuh-manager
   ```

   h. Enable and start the Wazuh manager service.

```
systemctl daemon-reload
systemctl enable wazuh-manager
systemctl start wazuh-manager
```

i. Run the following command to verify the Wazuh manager status.

```
systemctl status wazuh-manager
```

j. Installing Filebeat

k. Install the Filebeat package.

l. apt-get -y install filebeat

Install Procedure of Wazuh Dashboard:

Install and configure the Wazuh dashboard following step-by-step instructions. The Wazuh dashboard is a web interface for mining and visualizing the Wazuh server alerts and archived events.

1. Wazuh dashboard installation
    a. Installing package dependencies
    b. Install the following packages if missing.
    apt-get install debhelper tar curl libcap2-bin #debhelper version 9 or later
    c. Adding the Wazuh repository
    d. Install the following packages if missing.
    apt-get install gnupg apt-transport-https
    e. Install the GPG key.
    curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gpg --no-default-keyring --keyring gnupg-ring:/usr/share/keyrings/wazuh.gpg --import && chmod 644 /usr/share/keyrings/wazuh.gpg
    f. Add the repository.
    echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://packages.wazuh.com/4.x/apt/ stable main" | tee -a /etc/apt/sources.list.d/wazuh.list
    g. Update the packages information.
    apt-get update
    h. Installing the Wazuh dashboard
    i. Install the Wazuh dashboard package.
    apt-get -y install wazuh-dashboard

Install Procedure of Hive Threat Intelligence stack :

Docker-Compose Stack :

```
version: "3.7"
services:
  thehive:
    image: strangebee/thehive:5.2
    restart: unless-stopped
    depends_on:
      - cassandra
      - elasticsearch
      - minio
      - cortex
    mem_limit: 1500m
    ports:
      - 9000:9000
    environment:
      - JVM_OPTS=-Xms1024M -Xmx1024M
    command:
      - "--secret"
      - "***********************"
      - "--cql-hostnames"
      - "cassandra"
      - "--index-backend"
      - "elasticsearch"
      - "--es-hostnames"
      - "elasticsearch"
      - "--cortex-hostnames"
      - "cortex"
      - "--cortex-port"
      - "9001"
      - "--s3-endpoint"
      - "http://minio:9002"
      - "--s3-access-key"
      - "***********"
      - "--s3-secret-key"
      - "***********"
      - "--s3-use-path-access-style"

    volumes:
      - /srv/hivedata/application.conf:/etc/thehive/application.conf
    networks:
      - SOC_NET

  cassandra:
    image: cassandra:4
    restart: unless-stopped
    ports:
      - 9042:9042
    environment:
      - CASSANDRA_CLUSTER_NAME=TheHive
    volumes:
      - /srv/cassandradata:/var/lib/cassandra
    networks:
      - SOC_NET

  elasticsearch:
    image: elasticsearch:7.17.18
    restart: unless-stopped
    mem_limit: 1536m
    ports:
      - 9200:9200
      - 9300:9300
    environment:
      - discovery.type=single-node
      - xpack.security.enabled=false
      - cluster.name=hive
      - http.host=0.0.0.0
      - "ES_JAVA_OPTS=-Xms1536m -Xmx1536m"
    volumes:
      - /srv/esdata:/usr/share/elasticsearch/data
    healthcheck:
      test: [ "CMD-SHELL", "curl --silent --fail
localhost:9200/_cat/nodes?v\\&pretty || exit 1" ]
    networks:
      - SOC_NET

  minio:
    image: quay.io/minio/minio
    restart: unless-stopped
    command: ["minio", "server", "/data", "--console-address", ":9002"]
    environment:
      - MINIO_ROOT_USER=********
      - MINIO_ROOT_PASSWORD=********
    ports:
      - 9002:9002
    volumes:
      - /srv/miniodata:/data
    networks:
      - SOC_NET

#appended .local onto the container name because when we integrate cortex with TheHive using the new GUI menu it only accept a FQDN.
  cortex:
    image: thehiveproject/cortex:latest
    restart: unless-stopped
    environment:
      - /srv/cortexdata/job_directory=/tmp/cortex-jobs
      - /srv/cortexdata/docker_job_directory=/tmp/cortex-jobss
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - /srv/cortexdata/tmp/cortex-jobs:/tmp/cortex-jobs
      - /srv/cortexdata/cortex/logs:/var/log/cortex
      - /srv/cortexdata/cortex/application.conf:/cortex/application.conf
      - /var/lib/docker:/var/lib/docker
    depends_on:
      - elasticsearch
    ports:
      - 9001:9001
    networks:
      - SOC_NET
  misp:
    image: coolacid/misp-docker:core-latest
    restart: unless-stopped
    depends_on:
      - misp_mysql
    ports:
      - 80:80
      - 443:443
    volumes:
      - /srv/mispdata/server-configs/:/var/www/MISP/app/Config/
      - /srv/mispdata/logs/:/var/www/MISP/app/tmp/logs/
      - /srv/mispdata/files/:/var/www/MISP/app/files
      - /srv/mispdata/ssl/:/etc/nginx/certs
    environment:
      - MYSQL_HOST=misp_mysql
      - MYSQL_DATABASE=****db
      - MYSQL_USER=****
      - MYSQL_PASSWORD=****
      - MISP_ADMIN_EMAIL=*******@lab.local
      - MISP_ADMIN_PASSPHRASE=****
      - MISP_BASEURL=http://localhost
      - TIMEZONE=Asia/Kolkata
      - "INIT=true"
      - "CRON_USER_ID=1"
      - "REDIS_FQDN=redis"
    networks:
      - SOC_NET

  misp_mysql:
    image: mariadb:latest
    ports:
      - 3306:3306
    restart: unless-stopped
    volumes:
      - /srv/mispsqldata:/var/lib/mysql
    environment:
      - MYSQL_DATABASE=***db
      - MYSQL_USER=****
      - MYSQL_PASSWORD=****
      - MYSQL_ROOT_PASSWORD=****
    networks:
      - SOC_NET
  redis:
    image: redis:latest
    restart: unless-stopped
    volumes:
      - /srv/redisdata:/data
    networks:
      - SOC_NET
  misp-modules:
    image: coolacid/misp-docker:modules-latest
    restart: unless-stopped
    ports:
      - 6666:6666
    environment:
      - REDIS_BACKEND=redis
    depends_on:
      - redis
      - misp_mysql
    networks:
      - SOC_NET
networks:
    SOC_NET:
          driver: bridge
```

Install Procedure of Shuffle Automation stack :

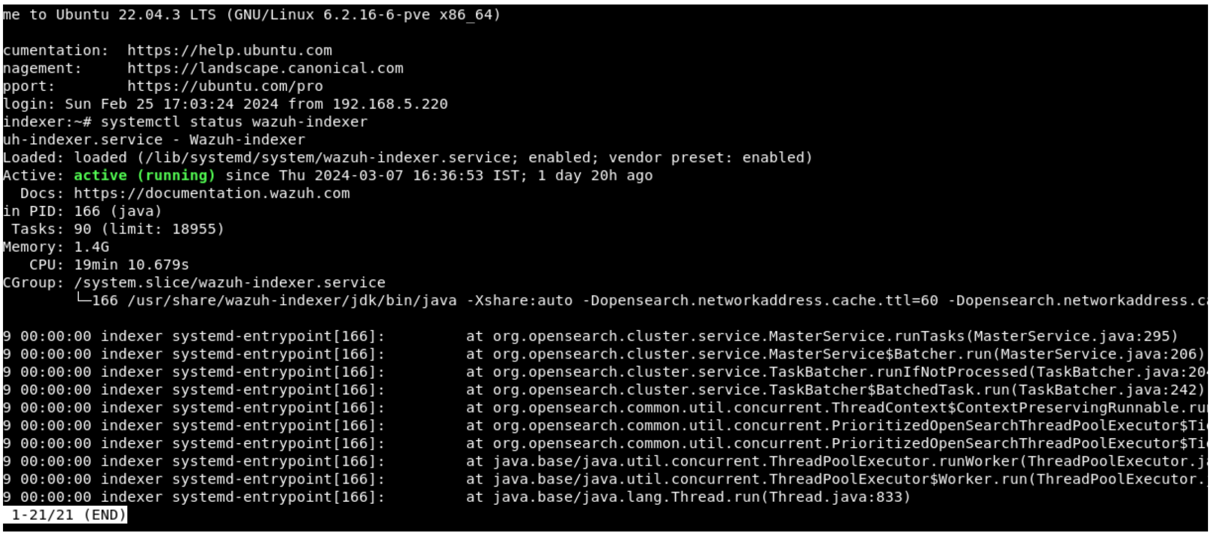
```
version: '3'
services:
  frontend:
    image: ghcr.io/shuffle/shuffle-frontend:latest
    container_name: shuffle-frontend
    hostname: shuffle-frontend
    ports:
      - "${FRONTEND_PORT}:80"
      - "${FRONTEND_PORT_HTTPS}:443"
    networks:
      - shuffle
    environment:
      - BACKEND_HOSTNAME=${BACKEND_HOSTNAME}
    restart: unless-stopped
    depends_on:
      - backend
  backend:
    image: ghcr.io/shuffle/shuffle-backend:latest
    container_name: shuffle-backend
    hostname: ${BACKEND_HOSTNAME}
    # Here for debugging:
    ports:
      - "${BACKEND_PORT}:5001"
    networks:
      - shuffle
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - ${SHUFFLE_APP_HOTLOAD_LOCATION}:/shuffle-apps:z
      - ${SHUFFLE_FILE_LOCATION}:/shuffle-files:z
    env_file: .env
    environment:
      #- DOCKER_HOST=tcp://docker-socket-proxy:2375
      - SHUFFLE_APP_HOTLOAD_FOLDER=/shuffle-apps
      - SHUFFLE_FILE_LOCATION=/shuffle-files
    restart: unless-stopped
  orborus:
    image: ghcr.io/shuffle/shuffle-orborus:latest
    container_name: shuffle-orborus
    hostname: shuffle-orborus
    networks:
      - shuffle
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    environment:
      - SHUFFLE_APP_SDK_TIMEOUT=300
      - SHUFFLE_ORBORUS_EXECUTION_CONCURRENCY=7 # The
amount of concurrent executions Orborus can handle.
      #- DOCKER_HOST=tcp://docker-socket-proxy:2375
      - ENVIRONMENT_NAME=${ENVIRONMENT_NAME}
      - BASE_URL=http://${OUTER_HOSTNAME}:5001
      - DOCKER_API_VERSION=1.40
      -
SHUFFLE_BASE_IMAGE_NAME=${SHUFFLE_BASE_IMAGE_NAME}
      -
SHUFFLE_BASE_IMAGE_REGISTRY=${SHUFFLE_BASE_IMAGE_REGI
STRY}
```

```
      -
SHUFFLE_BASE_IMAGE_TAG_SUFFIX=${SHUFFLE_BASE_IMAGE_TA
G_SUFFIX}
      - HTTP_PROXY=${HTTP_PROXY}
      - HTTPS_PROXY=${HTTPS_PROXY}
      -
SHUFFLE_PASS_WORKER_PROXY=${SHUFFLE_PASS_WORKER_PRO
XY}
      - SHUFFLE_PASS_APP_PROXY=${SHUFFLE_PASS_APP_PROXY}
      - SHUFFLE_STATS_DISABLED=true
    restart: unless-stopped
    security_opt:
      - seccomp:unconfined
  opensearch:
    image: opensearchproject/opensearch:2.11.0
    hostname: shuffle-opensearch
    container_name: shuffle-opensearch
    environment:
      - "OPENSEARCH_JAVA_OPTS=-Xms2048m -Xmx2048m" # minimum
and maximum Java heap size, recommend setting both to 50% of system RAM
      - bootstrap.memory_lock=true
      - DISABLE_PERFORMANCE_ANALYZER_AGENT_CLI=true
      - cluster.initial_master_nodes=shuffle-opensearch
      - cluster.routing.allocation.disk.threshold_enabled=false
      - cluster.name=shuffle-cluster
      - node.name=shuffle-opensearch
      - node.store.allow_mmap=false
      - discovery.seed_hosts=shuffle-opensearch
    ulimits:
      memlock:
        soft: -1
        hard: -1
      nofile:
        soft: 65536
        hard: 65536
    volumes:
      - ${DB_LOCATION}:/usr/share/opensearch/data:z
    ports:
      - 9200:9200
    networks:
      - shuffle
    restart: unless-stopped

  #memcached:
  #  image: memcached:latest
  #  container_name: shuffle-cache
  #  hostname: shuffle-cache
  #  mem_limit: 1024m
  #  environment:
  #    - MEMCACHED_MEMORY=1024
  #    - MEMCACHED_MAX_CONNECTIONS=2500
  #  ports:
  #    - 11211:11211

  #docker-socket-proxy:
  #  image: tecnativa/docker-socket-proxy
  #  container_name: docker-socket-proxy
  #  hostname: docker-socket-proxy
  # privileged: true
  # environment:
  #   - SERVICES=1
  #   - TASKS=1
  #   - NETWORKS=1
  #   - NODES=1
  #   - BUILD=1
  #   - IMAGES=1
  #   - GRPC=1
  #   - CONTAINERS=1
  #   - PLUGINS=1
  #   - SYSTEM=1
  #   - VOLUMES=1
  #   - INFO=1
  #   - DISTRIBUTION=1
  #   - POST=1
  #   - AUTH=1
  #   - SECRETS=1
  #   - SWARM=1
  # volumes:
  #   - /var/run/docker.sock:/var/run/docker.sock
  # networks:
  #   - shuffle
  #
networks:
  shuffle:
    driver: bridge

    # uncomment to set MTU for swarm mode.
    # MTU should be whatever your host's preferred MTU is.
    # Refer to this doc to figure out what your host's MTU is:
    # https://shuffler.io/docs/troubleshooting#TLS_timeout_error/Timeout_Errors/EOF_Errors
    # driver_opts:
    #   com.docker.network.driver.mtu: 1460
```

APPENDIX

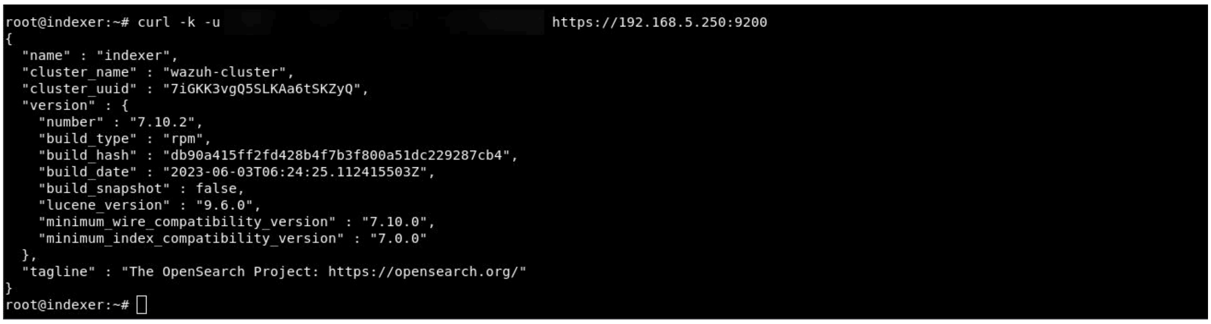
A2 – SCREENSHOTS

WAZUH INSTALLATIONS



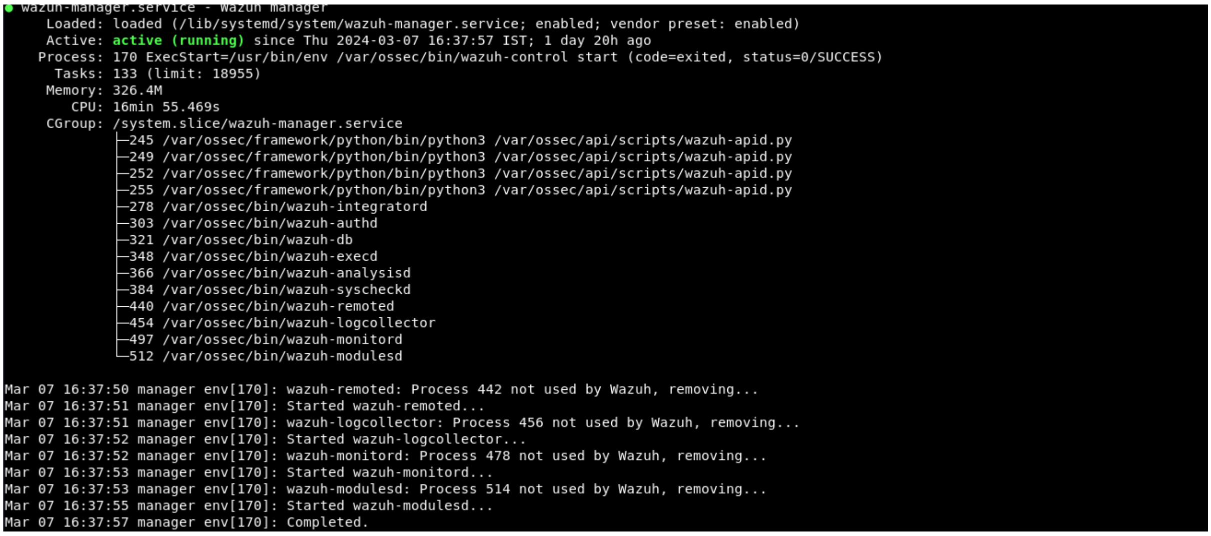
```
me to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.16-6-pve x86_64)

cumentation:  https://help.ubuntu.com
nagement:     https://landscape.canonical.com
pport:        https://ubuntu.com/pro
login: Sun Feb 25 17:03:24 2024 from 192.168.5.220
indexer:~# systemctl status wazuh-indexer
uh-indexer.service - Wazuh-indexer
Loaded: loaded (/lib/systemd/system/wazuh-indexer.service; enabled; vendor preset: enabled)
Active: active (running) since Thu 2024-03-07 16:36:53 IST; 1 day 20h ago
  Docs: https://documentation.wazuh.com
in PID: 166 (java)
 Tasks: 90 (limit: 18955)
Memory: 1.4G
   CPU: 19min 10.679s
CGroup: /system.slice/wazuh-indexer.service
        └─166 /usr/share/wazuh-indexer/jdk/bin/java -Xshare:auto -Dopensearch.networkaddress.cache.ttl=60 -Dopensearch.networkaddress.ca

9 00:00:00 indexer systemd-entrypoint[166]:         at org.opensearch.cluster.service.MasterService.runTasks(MasterService.java:295)
9 00:00:00 indexer systemd-entrypoint[166]:         at org.opensearch.cluster.service.MasterService$Batcher.run(MasterService.java:206)
9 00:00:00 indexer systemd-entrypoint[166]:         at org.opensearch.cluster.service.TaskBatcher.runIfNotProcessed(TaskBatcher.java:20
9 00:00:00 indexer systemd-entrypoint[166]:         at org.opensearch.cluster.service.TaskBatcher$BatchedTask.run(TaskBatcher.java:242)
9 00:00:00 indexer systemd-entrypoint[166]:         at org.opensearch.common.util.concurrent.ThreadContext$ContextPreservingRunnable.ru
9 00:00:00 indexer systemd-entrypoint[166]:         at org.opensearch.common.util.concurrent.PrioritizedOpenSearchThreadPoolExecutor$Ti
9 00:00:00 indexer systemd-entrypoint[166]:         at org.opensearch.common.util.concurrent.PrioritizedOpenSearchThreadPoolExecutor$Ti
9 00:00:00 indexer systemd-entrypoint[166]:         at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.j
9 00:00:00 indexer systemd-entrypoint[166]:         at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.
9 00:00:00 indexer systemd-entrypoint[166]:         at java.base/java.lang.Thread.run(Thread.java:833)
 1-21/21 (END)
```

Figure A2.1 : Status of Wazuh Indexer



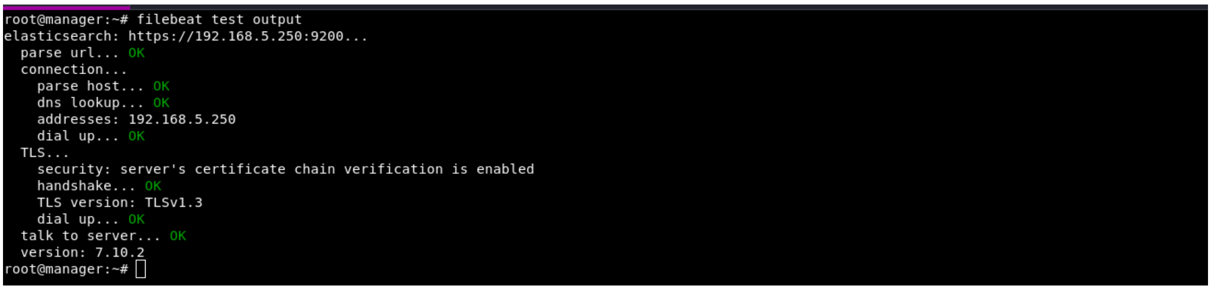
```
root@indexer:~# curl -k -u [redacted] https://192.168.5.250:9200
{
  "name" : "indexer",
  "cluster_name" : "wazuh-cluster",
  "cluster_uuid" : "7iGKK3vgQ5SLKAa6tSKZyQ",
  "version" : {
    "number" : "7.10.2",
    "build_type" : "rpm",
    "build_hash" : "db90a415ff2fd428b4f7b3f800a51dc229287cb4",
    "build_date" : "2023-06-03T06:24:25.112415503Z",
    "build_snapshot" : false,
    "lucene_version" : "9.6.0",
    "minimum_wire_compatibility_version" : "7.10.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "The OpenSearch Project: https://opensearch.org/"
}
root@indexer:~# 
```

Figure A2.2 : Wazuh Indexer Connection Output



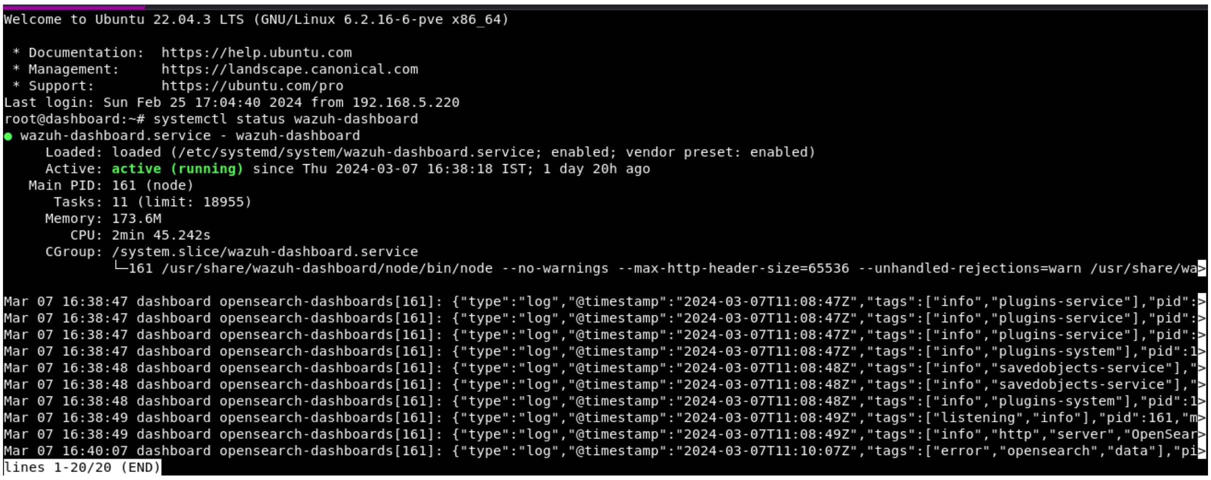
```
● wazuh-manager.service - Wazuh manager
     Loaded: loaded (/lib/systemd/system/wazuh-manager.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2024-03-07 16:37:57 IST; 1 day 20h ago
    Process: 170 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
      Tasks: 133 (limit: 18955)
     Memory: 326.4M
        CPU: 16min 55.469s
     CGroup: /system.slice/wazuh-manager.service
             ├─245 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh-apid.py
             ├─249 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh-apid.py
             ├─252 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh-apid.py
             ├─255 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh-apid.py
             ├─278 /var/ossec/bin/wazuh-integratord
             ├─303 /var/ossec/bin/wazuh-authd
             ├─321 /var/ossec/bin/wazuh-db
             ├─348 /var/ossec/bin/wazuh-execd
             ├─366 /var/ossec/bin/wazuh-analysisd
             ├─384 /var/ossec/bin/wazuh-syscheckd
             ├─440 /var/ossec/bin/wazuh-remoted
             ├─454 /var/ossec/bin/wazuh-logcollector
             ├─497 /var/ossec/bin/wazuh-monitord
             └─512 /var/ossec/bin/wazuh-modulesd

Mar 07 16:37:50 manager env[170]: wazuh-remoted: Process 442 not used by Wazuh, removing...
Mar 07 16:37:51 manager env[170]: Started wazuh-remoted...
Mar 07 16:37:51 manager env[170]: wazuh-logcollector: Process 456 not used by Wazuh, removing...
Mar 07 16:37:52 manager env[170]: Started wazuh-logcollector...
Mar 07 16:37:52 manager env[170]: wazuh-monitord: Process 478 not used by Wazuh, removing...
Mar 07 16:37:53 manager env[170]: Started wazuh-monitord...
Mar 07 16:37:53 manager env[170]: wazuh-modulesd: Process 514 not used by Wazuh, removing...
Mar 07 16:37:55 manager env[170]: Started wazuh-modulesd...
Mar 07 16:37:57 manager env[170]: Completed.
```

Figure A2.3 : Status of Wazuh Manager (Server)



```
root@manager:~# filebeat test output
elasticsearch: https://192.168.5.250:9200...
  parse url... OK
  connection...
    parse host... OK
    dns lookup... OK
    addresses: 192.168.5.250
    dial up... OK
  TLS...
    security: server's certificate chain verification is enabled
    handshake... OK
    TLS version: TLSv1.3
    dial up... OK
  talk to server... OK
  version: 7.10.2
root@manager:~# 
```

Figure A2.4 : Output of Filebeat



```
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.16-6-pve x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro
Last login: Sun Feb 25 17:04:40 2024 from 192.168.5.220
root@dashboard:~# systemctl status wazuh-dashboard
● wazuh-dashboard.service - wazuh-dashboard
     Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2024-03-07 16:38:18 IST; 1 day 20h ago
   Main PID: 161 (node)
      Tasks: 11 (limit: 18955)
     Memory: 173.6M
        CPU: 2min 45.242s
     CGroup: /system.slice/wazuh-dashboard.service
             └─161 /usr/share/wazuh-dashboard/node/bin/node --no-warnings --max-http-header-size=65536 --unhandled-rejections=warn /usr/share/wa>

Mar 07 16:38:47 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:47Z","tags":["info","plugins-service"],"pid":>
Mar 07 16:38:47 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:47Z","tags":["info","plugins-service"],"pid":>
Mar 07 16:38:47 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:47Z","tags":["info","plugins-service"],"pid":>
Mar 07 16:38:47 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:47Z","tags":["info","plugins-system"],"pid":1>
Mar 07 16:38:48 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:48Z","tags":["info","savedobjects-service"],">
Mar 07 16:38:48 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:48Z","tags":["info","savedobjects-service"],">
Mar 07 16:38:48 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:48Z","tags":["info","plugins-system"],"pid":1>
Mar 07 16:38:49 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:49Z","tags":["listening","info"],"pid":161,"m>
Mar 07 16:38:49 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:08:49Z","tags":["info","http","server","OpenSear>
Mar 07 16:40:07 dashboard opensearch-dashboards[161]: {"type":"log","@timestamp":"2024-03-07T11:10:07Z","tags":["error","opensearch","data"],"pi>
lines 1-20/20 (END)
```

Figure A2.5 : Status of Wazuh Dashboard

WEB INTERFACES

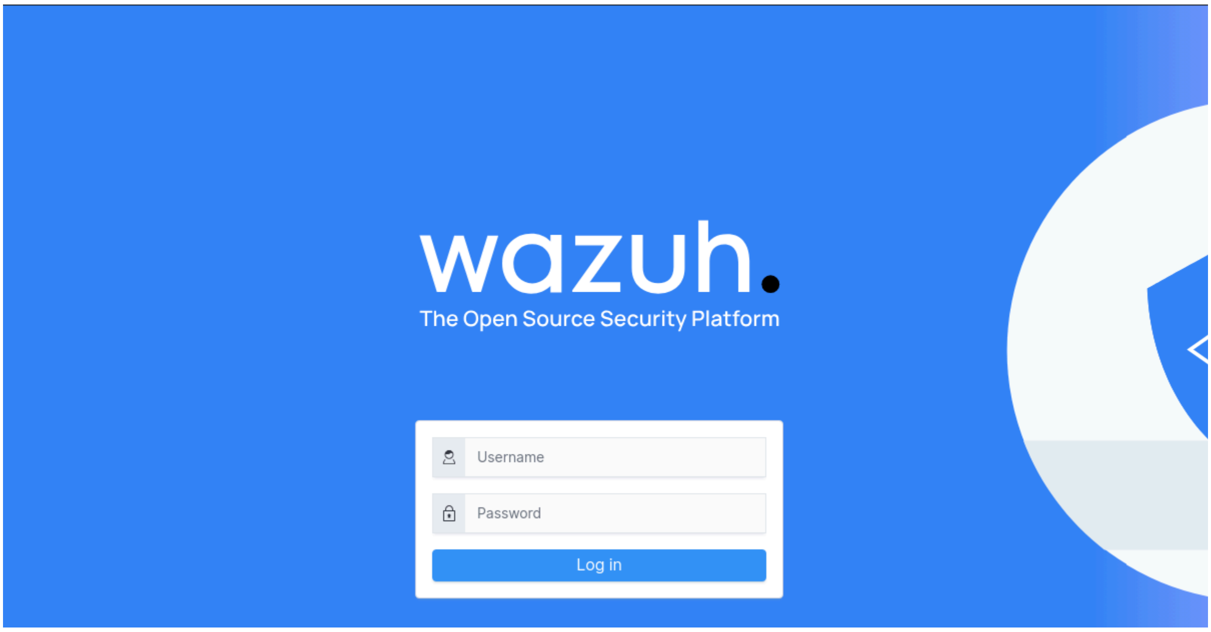
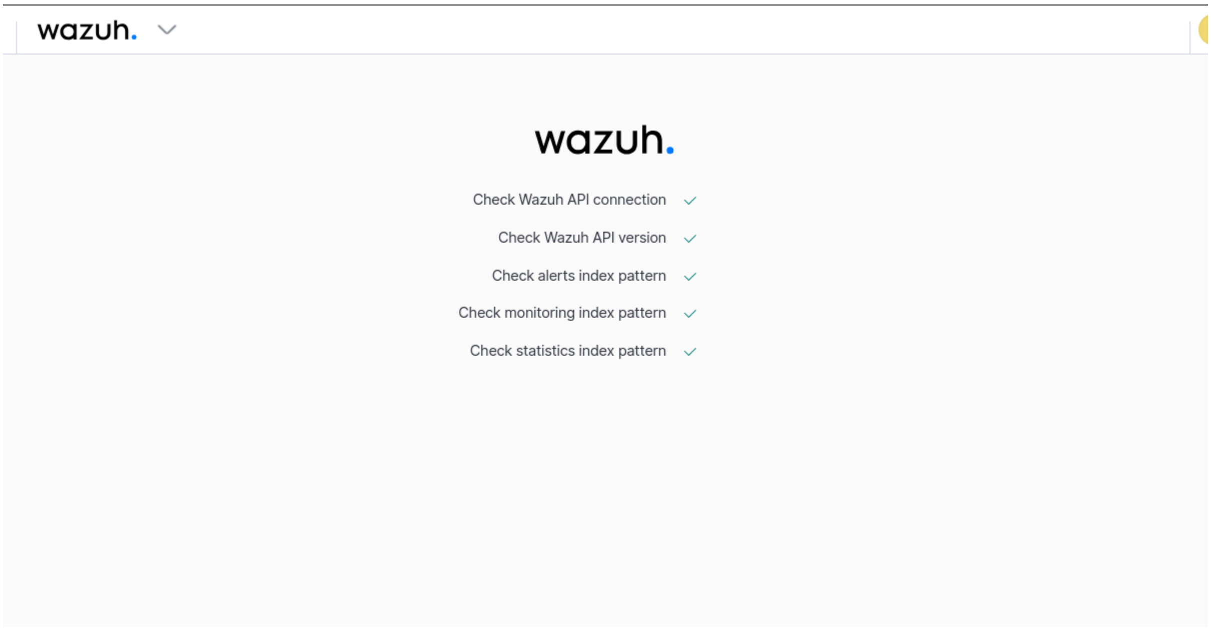


Figure A2.6 : Wazuh Login Interface



Check Wazuh API connection ✓

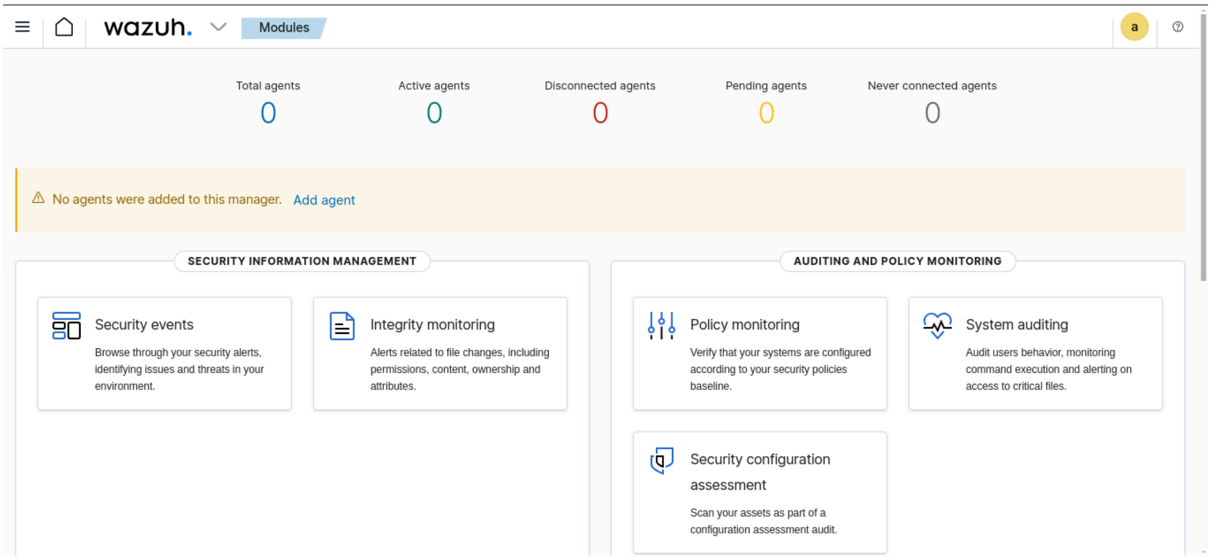
Check Wazuh API version ✓

Check alerts index pattern ✓

Check monitoring index pattern ✓

Check statistics index pattern ✓

Figure A2.7 : Wazuh Post Login Check



Total agents 0

Active agents 0

Disconnected agents 0

Pending agents 0

Never connected agents 0

⚠ No agents were added to this manager. Add agent

SECURITY INFORMATION MANAGEMENT

Security events
Browse through your security alerts, identifying issues and threats in your environment.

Integrity monitoring
Alerts related to file changes, including permissions, content, ownership and attributes.

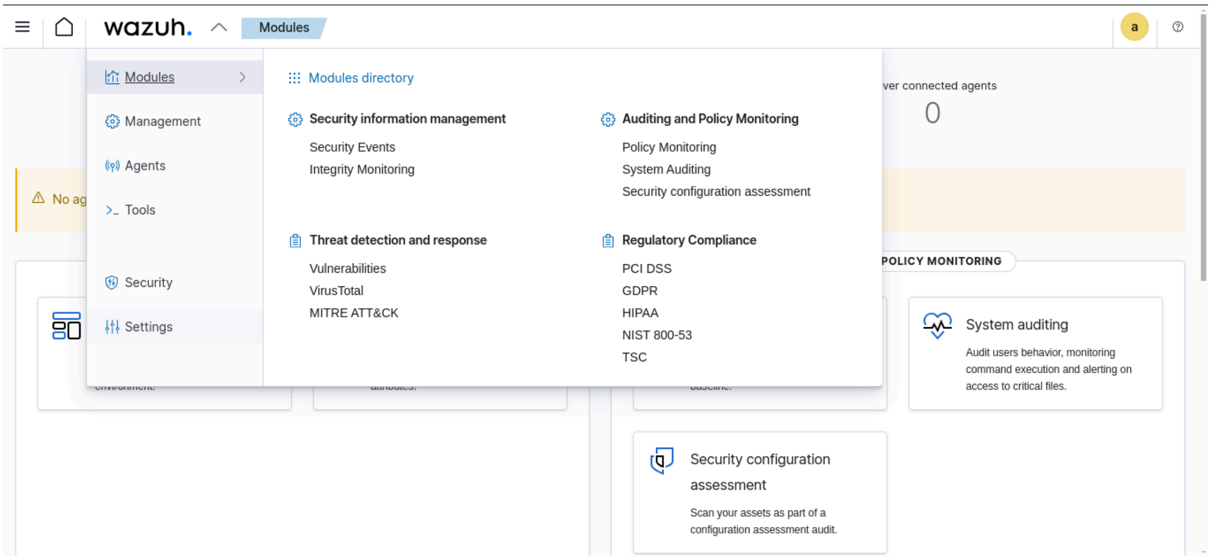
AUDITING AND POLICY MONITORING

Policy monitoring
Verify that your systems are configured according to your security policies baseline.

System auditing
Audit users behavior, monitoring command execution and alerting on access to critical files.

Security configuration assessment
Scan your assets as part of a configuration assessment audit.

Figure A2.8: Wazuh Dashboard Interface



Security information management
- Security Events
- Integrity Monitoring

Auditing and Policy Monitoring
- Policy Monitoring
- System Auditing
- Security configuration assessment

Threat detection and response
- Vulnerabilities
- VirusTotal
- MITRE ATT&CK

Regulatory Compliance
- PCI DSS
- GDPR
- HIPAA
- NIST 800-53
- TSC

Figure A2.9 : Wazuh Features Drop Down Box

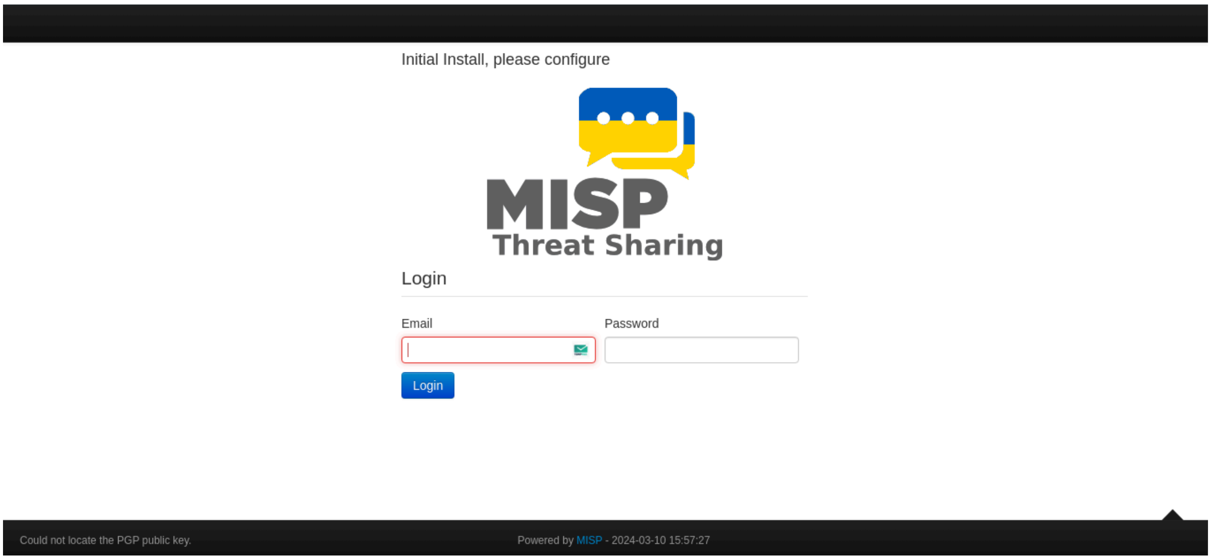

Initial Install, please configure
MISP
Threat Sharing
Login
Email
Password
Login
Could not locate the PGP public key.
Powered by MISP - 2024-03-10 15:57:27

Figure A2.10 : MISP Login Interface

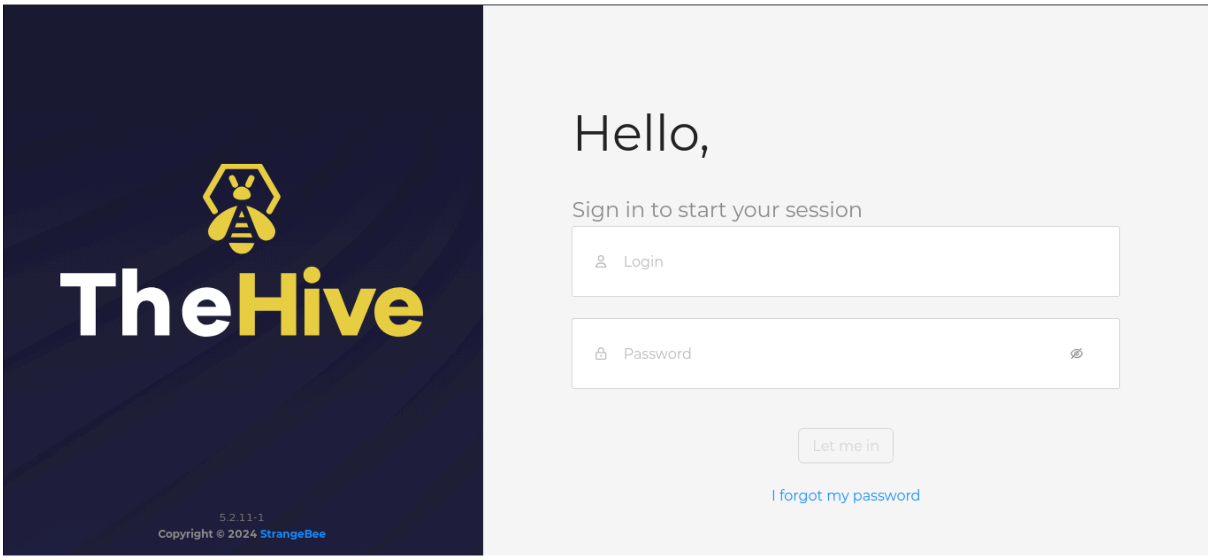
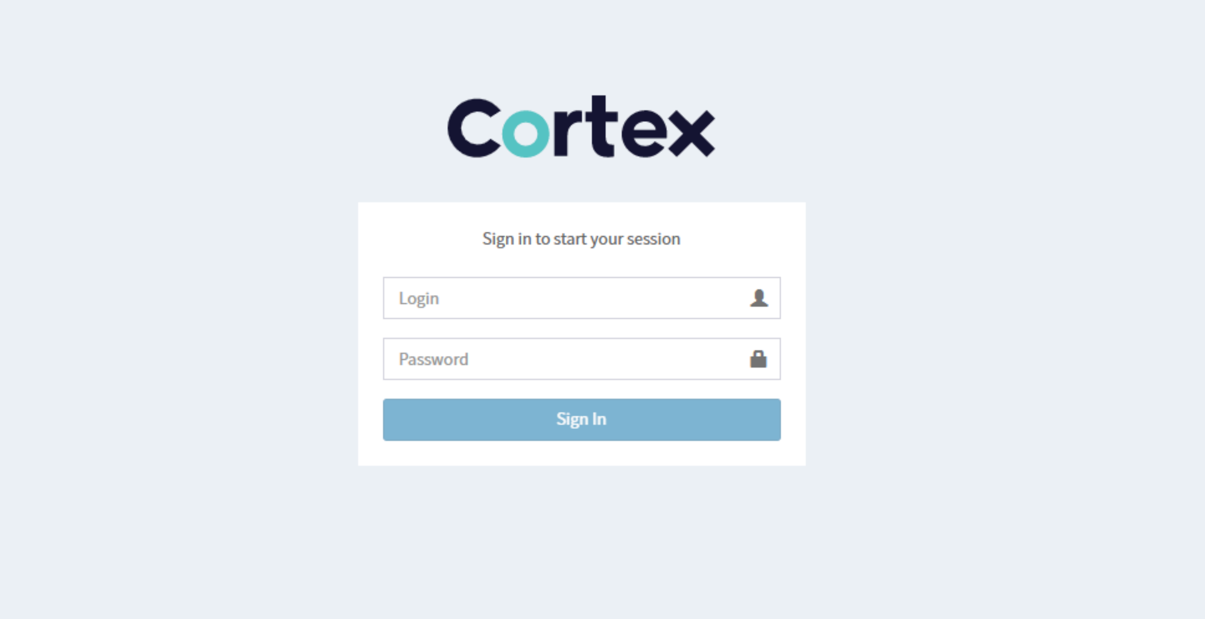


Figure A2.11 : Hive Login Interface

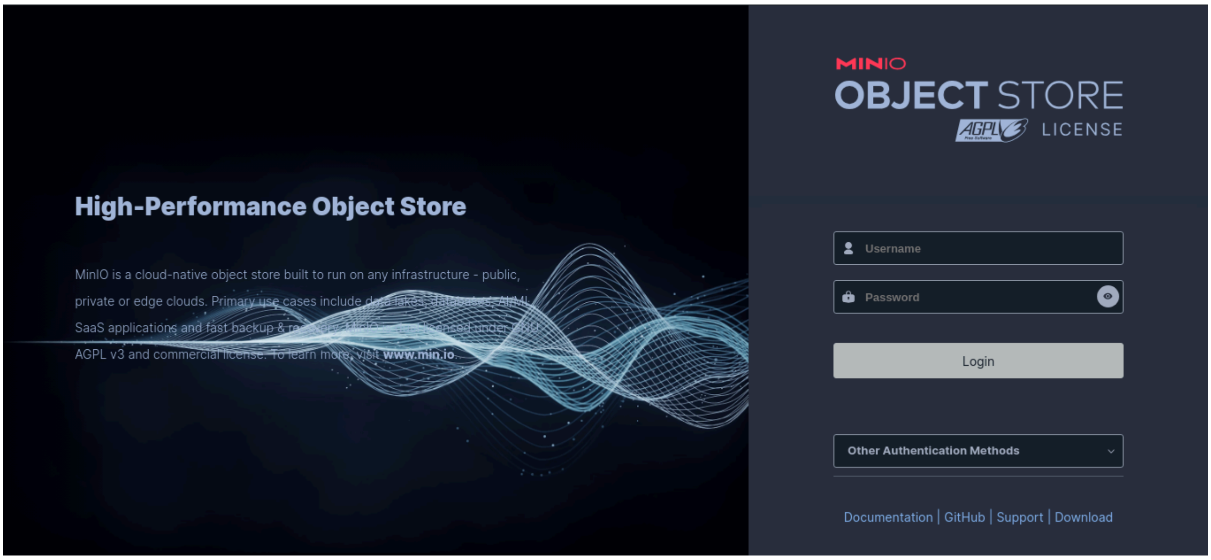


Sign in to start your session

Login

Password

Figure A2.12 : Cortex Login Interface



High-Performance Object Store

MinIO is a cloud-native object store built to run on any infrastructure - public, private or edge clouds. Primary use cases include data lakes, databases, AI/ML SaaS applications and fast backup & recovery. MinIO is dual-licensed under GNU AGPL v3 and commercial license. To learn more, visit **www.min.io**.

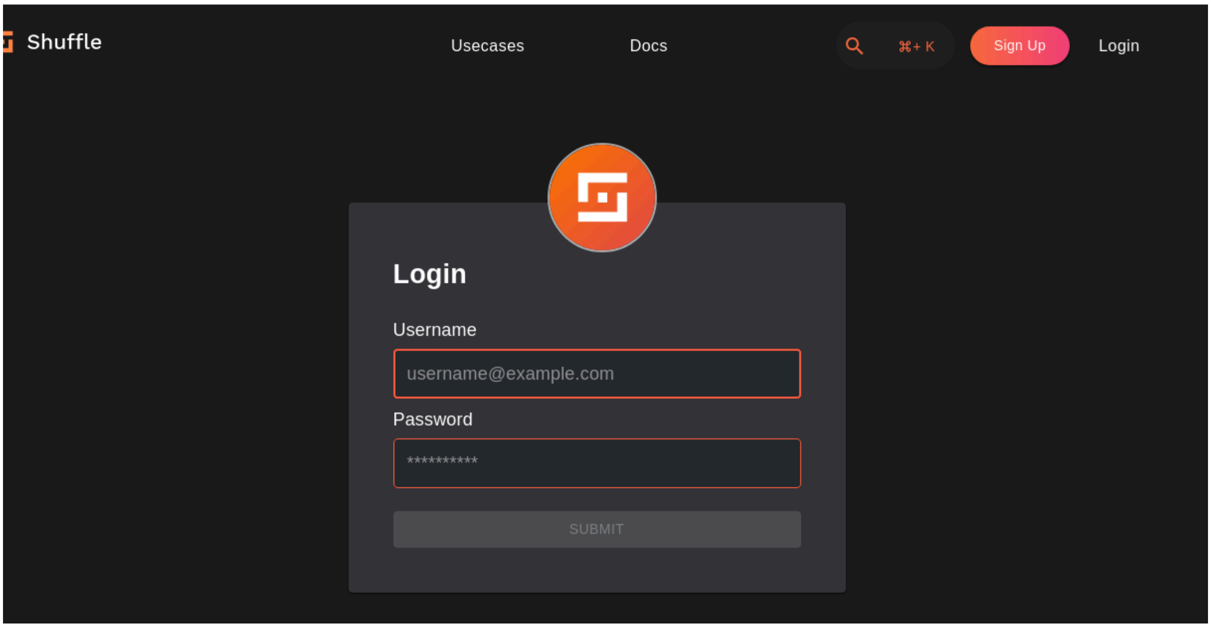
MINIO
OBJECT STORE
AGPLv3 LICENSE

Username

Password

Login
Other Authentication Methods
Documentation | GitHub | Support | Download

Figure A2.13 : Minio Login Interface



Login

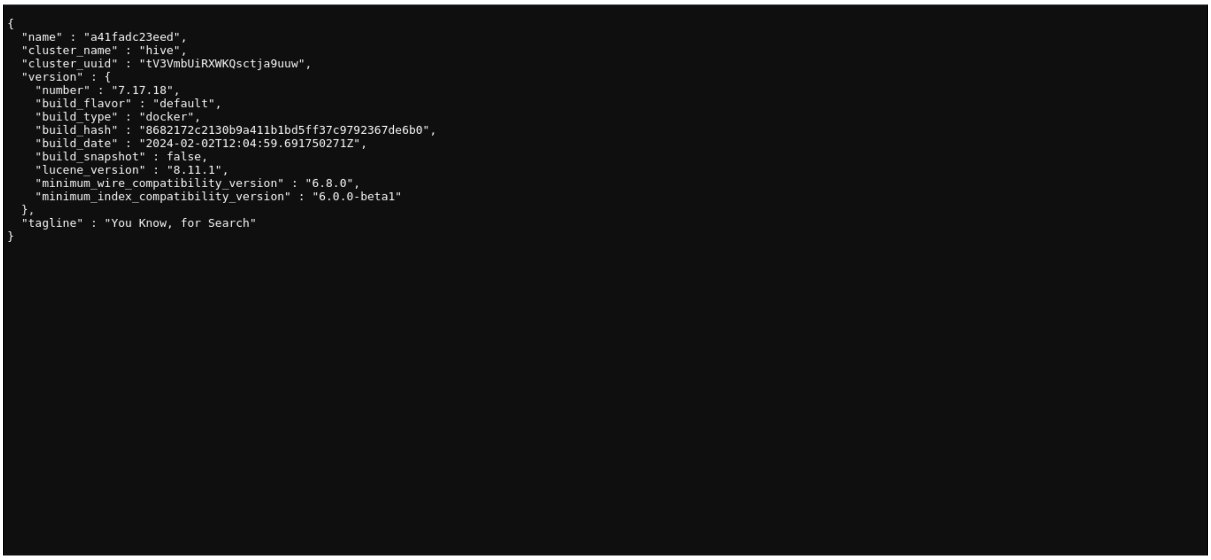
Username

username@example.com

Password

****

Figure A2.14 : Shuffle Login Interface



```
{
  "name" : "a41fadc23eed",
  "cluster_name" : "hive",
  "cluster_uuid" : "tV3VmbUiRXWKQsctja9uuw",
  "version" : {
    "number" : "7.17.18",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "8682172c2130b9a411b1bd5ff37c9792367de6b0",
    "build_date" : "2024-02-02T12:04:59.691750271Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Figure A2.15 : ElasticSearch Connection status

REFERENCES

1. L. F. Ilca, O. P. Lucian, and T. Bălan, "Enhancing Cyber-Resilience for Small and Medium-Sized Organizations with Prescriptive Malware Analysis, Detection and Response," *Sensors*, Jul. 28, 2023. https://www.mdpi.com/1424-8220/23/15/6757

2. "Implementation of SOC using ELK with Integration of Wazuh and Dedicated File Integrity Monitoring," *IEEE Conference Publication | IEEE Xplore*, Aug. 17, 2023. https://ieeexplore.ieee.org/document/10334992

3. O. Negoita and M. Carabaș, "Enhanced Security Using Elasticsearch and Machine Learning," *Advances in intelligent systems and computing*, Jan. 01, 2020. https://link.springer.com/chapter/10.1007/978-3-030-52243-8_19

4. "A SIEM and Multiple Analysis Software Integrated Malware Detection Approach," *IEEE Conference Publication | IEEE Xplore*, Dec. 11, 2023. https://ieeexplore.ieee.org/document/10425463

5. Wazuh, "Getting started with Wazuh · Wazuh documentation." https://documentation.wazuh.com/current/getting-started/index.html

6. "TheHive 5 Documentation," *StrangeBee Docs*. https://docs.strangebee.com/thehive/setup/

7. "TheHive 5 Documentation," *StrangeBee Docs*. https://docs.strangebee.com/cortex/

8. Misp, "MISP Documentation and Support," *MISP Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing*. https://www.misp-project.org/documentation/

9. “Shuffle,” *Shuffle*. https://shuffler.io/docs/about

10. “MinIO Key Encryption Service,” *Documentation*. https://min.io/docs/kes/