AKHUWAT COLLEGE KASUR

DEPARTMENT OF INFORMATION TECHNOLOGY

SOFTWARE DESIGN SPECIFICATION (SDS)

AI-DRIVEN SECURITY MONITORING: ANOMALY DETECTION USING ELASTIC STACK & WAZUH

SUPERVISED BY:

EBRYX TEAM

&

MR. MUHAMMAD NAEEM AKHTAR

GROUP MEMEBERS:
Khalid Hussain
Muhammad Osama

Table of Contents

1. System Design

1.1 Product Perspective

The system will operate as a security monitoring tool, working with the Wazuh SIEM *(Security Information and Event Management)* platform and the Elastic Stack for log management. The system will interact with external systems for log collection and internal components for anomaly detection.

1.2 Design Considerations

- **Assumptions and Dependencies:**
  - Wazuh, Filebeat, and Elastic Stack are properly configured.
  - There is enough log data for AI analysis.
- **Limitations:**
  - The system will only send alerts; it will not automatically block threats.
  - It will support only **web-based dashboards** *(no mobile app).*
- **Risks:**
  - **Integration Challenges:** Issues in connecting with external log sources.
  - **Performance Issues:** The system may slow down with large log volumes.
  - **Data Quality:** Poor log data can reduce AI accuracy.

2. Requirements Traceability Matrix

Requirement ID	Requirement Description	Design Component	Test Case ID	Implementation Status
FR-1	Log Collection	Wazuh Log Collection Module	TC-01	Implemented
FR-2	AI-Based Detection	Machine Learning Model	TC-02	***In Progress***
FR-3	Alert System	Alert Generation & Notification System	TC-03	***In Progress***
FR-4	Dashboard	Kibana Dashboard	TC-04	Implemented
FR-5	Anomalous File Creation Detection	Wazuh File Integrity Monitoring (FIM)	TC-05	***In Progress***
FR-6	Suspicious Login Volume Detection	AI-Based Login Volume Anomaly Detection	TC-06	Implemented
FR-7	Alert Customization	User-Configurable Alert System	TC-07	Implemented
FR-8	Incident Reporting	Incident Management & Documentation Module	TC-08	Not Started

3. Design Models

3.1 Architectural Design

The system follows a **Multi-Tier Architecture**, consisting of the following layers:

3.1.1 Data Collection Layer:

- **Tool Used:** ***Wazuh Agents***
- **Purpose:** Collects security logs from endpoints, servers, and network devices.
- **How It Works:** Wazuh agents monitor system activities and gather raw log data for security analysis.

3.1.2 Processing Layer:

- **Tool Used:** ***Wazuh Manager***
- **Purpose:** Processes the collected logs to filter and analyze them.
- **How It Works:** Wazuh Manager organizes logs and applies basic security rules before forwarding.

3.1.3 Log Forwarding:

- **Tool Used:** ***Filebeat***
- **Purpose:** Sends logs from the Wazuh Manager to ***Elasticsearch***.
- **How It Works:** Filebeat acts as a lightweight shipper, ensuring logs are efficiently transferred to the storage layer.

3.1.4 Storage Layer:

- **Tool Used:** ***Elasticsearch***
- **Purpose:** Stores and indexes all log data securely.
- **How It Works:** Acts like a search engine for logs, enabling fast data retrieval and analysis.

3.1.5 Analysis Layer:

- **Tool Used:** ***AI-Based Anomaly Detection***
- **Purpose:** Identifies unusual patterns and potential security threats in the log data.
- **How It Works:** The AI model analyzes logs to detect anomalies such as unauthorized file creation or suspicious login attempts.

3.1.6 Visualization Layer:

- **Tool Used:** ***Kibana***
- **Purpose:** Provides dashboards for log visualization and alert management.
- **How It Works:** Displays security insights through graphs, charts, and real-time alerts, helping security analysts monitor and respond to threats effectively.

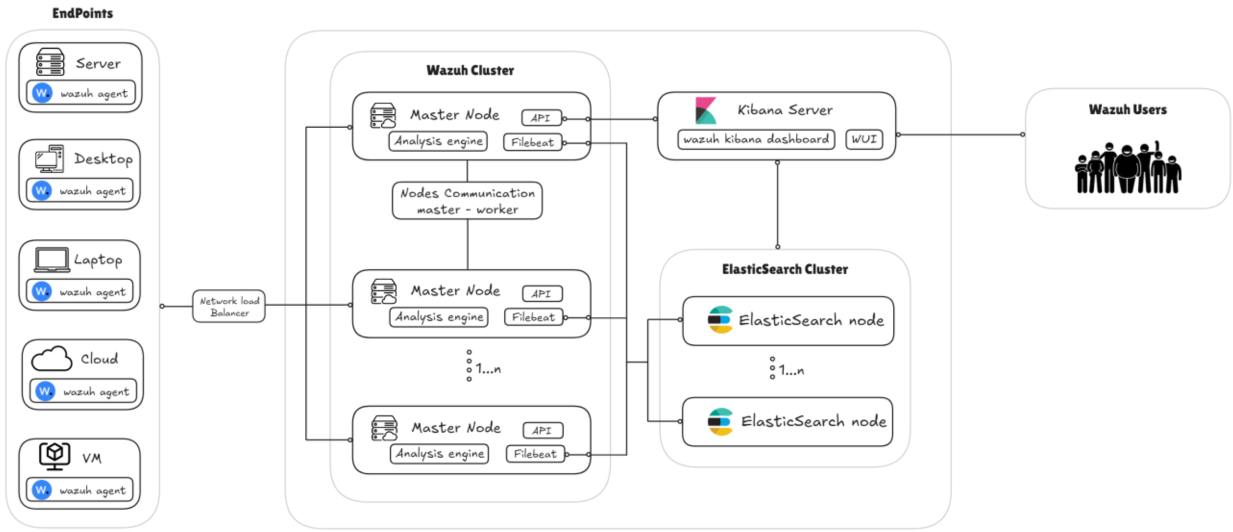
3.2 Data Design

The system uses **Elasticsearch** to store and manage all the log data. Elasticsearch is fast and makes it easy to search and analyze security information.

3.2.1 Data Structures Used:

1. Log Data:

- **Purpose:** Helps track activities across servers, computers, and network devices.
- **Example:** When a file is created or when someone tries to log in, this data is saved as a log entry.



2. Alert Data:

- **Purpose:** Maintains information about suspicious activities detected by AI.
- **Example:** If the system detects too many failed login attempts, it creates an alert with the time, source, and type of threat.

3. User Data:

- **Purpose:** Manages user roles and permissions to control who can access and manage the system.
- **Example:** An admin can change system settings, while a security analyst can only review alerts.

3.2.2 Data Dictionary

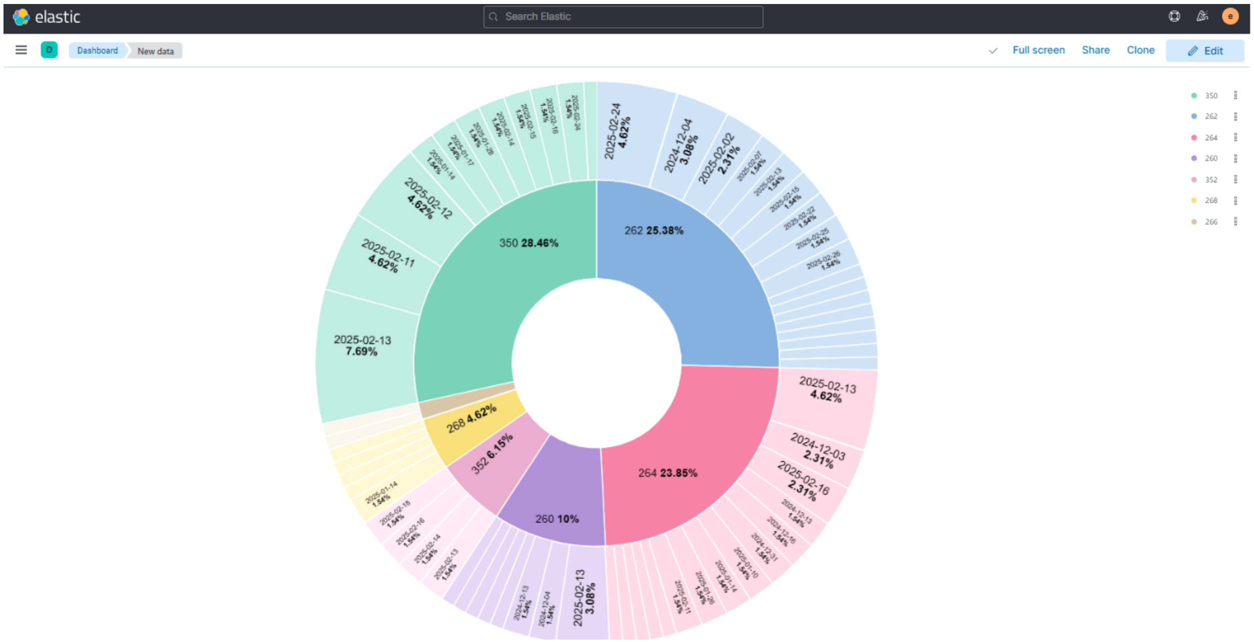
The Data Dictionary explains important terms used in the system's data design. It helps everyone understand what each data element means and how it is used.

Term	Description
Log Event	Stores each security log entry, including details like the time, event type, and source.
Anomaly Alert	Holds information about detected threats, such as the alert type and the action taken.
User Role	Defines the access levels of users, such as security analysts, administrators, and regular users.

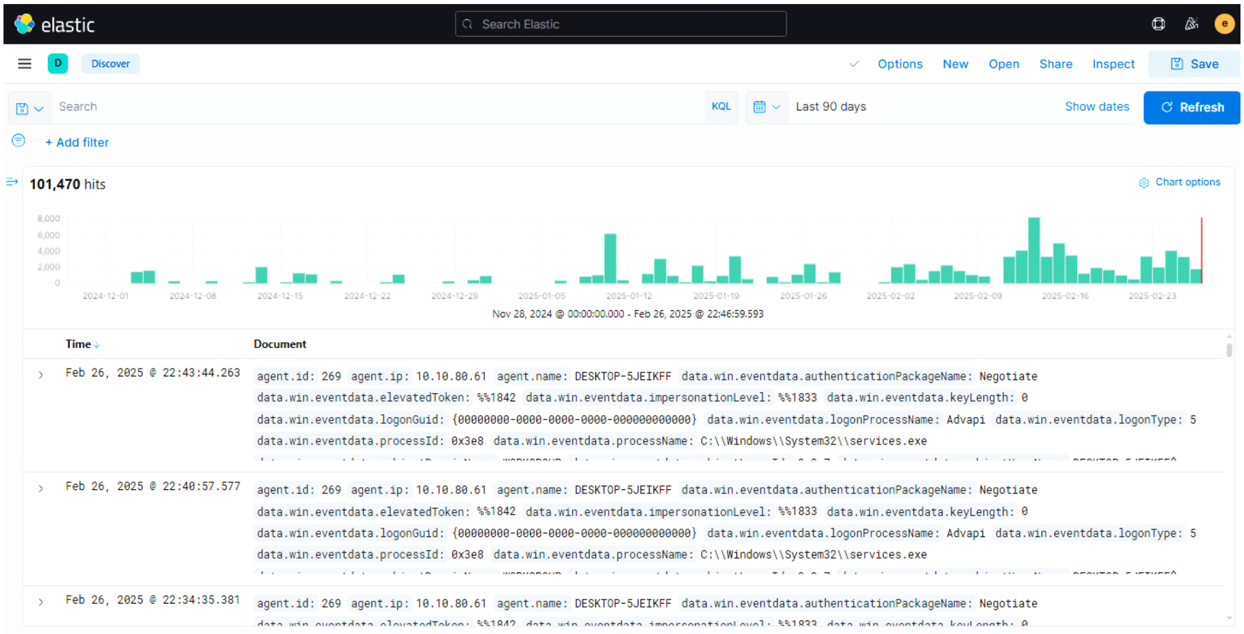
3.3 User Interface Design

3.3.1 The Kibana Dashboard:

- Show alerts with threat levels.
- Provide filters to analyze specific logs.
- Allow security analysts to mark alerts as legitimate or threats.



3.3.2 Logs View:



101,470 hits

Nov 28, 2024 @ 00:00:00.000 - Feb 26, 2025 @ 22:46:59.593

```
Time ↓                      Document
Feb 26, 2025 @ 22:43:44.263 agent.id: 269 agent.ip: 10.10.80.61 agent.name: DESKTOP-5JEIKFF data.win.eventdata.authenticationPackageName: Negotiate
                            data.win.eventdata.elevatedToken: %%1842 data.win.eventdata.impersonationLevel: %%1833 data.win.eventdata.keyLength: 0
                            data.win.eventdata.logonGuid: {00000000-0000-0000-0000-000000000000} data.win.eventdata.logonProcessName: Advapi data.win.eventdata.logonType: 5
                            data.win.eventdata.processId: 0x3e8 data.win.eventdata.processName: C:\\Windows\\System32\\services.exe
Feb 26, 2025 @ 22:40:57.577 agent.id: 269 agent.ip: 10.10.80.61 agent.name: DESKTOP-5JEIKFF data.win.eventdata.authenticationPackageName: Negotiate
                            data.win.eventdata.elevatedToken: %%1842 data.win.eventdata.impersonationLevel: %%1833 data.win.eventdata.keyLength: 0
                            data.win.eventdata.logonGuid: {00000000-0000-0000-0000-000000000000} data.win.eventdata.logonProcessName: Advapi data.win.eventdata.logonType: 5
                            data.win.eventdata.processId: 0x3e8 data.win.eventdata.processName: C:\\Windows\\System32\\services.exe
Feb 26, 2025 @ 22:34:35.381 agent.id: 269 agent.ip: 10.10.80.61 agent.name: DESKTOP-5JEIKFF data.win.eventdata.authenticationPackageName: Negotiate
```

3.4 Behavioural Model

The Behavioural Model shows how the system works step-by-step to detect security threats.

1. **Log Collection:**

   - The system uses Wazuh to collect log data from all connected devices *(endpoints)*.
   - These logs include activities like file changes, login attempts, and system events.

2. **Data Processing:**

   - The *AI model* reviews the collected logs to find any unusual patterns or behaviors.
   - It looks for signs of security threats, like strange file creations or too many failed logins.

3. **Anomaly Detection:**

   - The system uses *AI* to spot potential threats.
   - It can detect issues like unauthorized file creation or suspicious login attempts.

4. **Alert Generation:**

   - When a threat is detected, the system automatically creates an *alert*.
   - The alert is shown on the *Kibana dashboard*, making it easy for security teams to see.

5. **User Action:**

- ***Security analysts*** review the alerts on the dashboard.
- They decide whether the alert is a real threat or a false alarm and take the appropriate action, such as blocking access or marking the alert as safe.

3.4.1 Interaction Diagrams

- **Sequence Diagram:** Shows (*Figure: 04*) log flow from Wazuh to Elastic Stack and then to AI model.

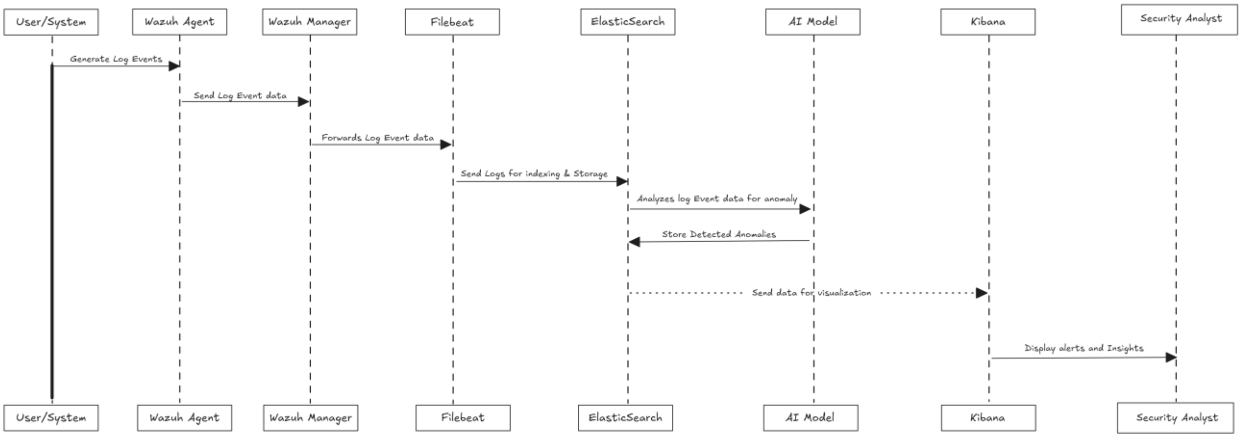


Figure: 04

4. Design Decisions

This section explains the main choices made during the design of the system and why they were chosen.

1. AI Model :

- **Used:** ***AI Algorithm***
- Helps build effective *anomaly detection models* to find unusual activities in log data.

2. Data Storage:

- **Tool Used:** ***Elasticsearch***
- Elasticsearch is great at *storing logs* and allows *fast searching* of large amounts of data.
- **Benefit:** Makes it easy to *store, search, and manage log data* efficiently.

3. Frontend Technology:

- **Tool Used: *Kibana***
- Kibana provides a *simple and easy-to-use interface* for displaying security alerts and data.
- Allows security analysts to *see alerts, analyze logs, and monitor system status* through clear and interactive *dashboards.*

5. Summary

Our security monitoring system makes it easy to detect threats in real time. It uses Wazuh, the Elastic Stack, and AI to watch for unusual activity and alert security teams quickly. With simple dashboards, teams can easily see security events and take action before problems get worse. AI helps reduce false alarms, so only real threats get attention. This system works for both small networks and large businesses, making sure data and systems stay safe. By catching risks early, it helps organizations respond fast and prevent damage.

References

- *Wazuh* All-in-one deployment with Elastic Stack. *https://documentation.wazuh.com/4.5/deployment-options/elastic-stack/all-in-onedeployment/index.html*

- Machine learning & security protecting systems with data and algorithms by *clarencechio & david freeman* *https://a.co/d/cFCmldy*

- ***Security Monitoring with Wazuh:*** A hands-on guide to effective enterprise security using real-life use cases in Wazuh by *Rajneesh Gupta* *https://a.co/d/540PA1L*