F24-008-D-IntraSec

Project Team

Muhammad Ahmad Hanif 21I-1557
Muhammad Arfat 21I-0554

Session 2021-2025

Supervised by

Dr. Muhammad Asim

Department of Computer Science

National University of Computer and Emerging Sciences
Islamabad, Pakistan

June, 2025

Contents

List of Figures

List of Tables

Chapter 1

Introduction

IntraSec is a client-less risk assessment and asset management tool , with the goal of providing real-time inventory of network assets and identifying security risks across the internal network without requiring agent/client installation individual machines.

1.1 Problem Statement

In large organizations, as the network constantly expands and becomes more and more complex. It becomes harder to keep track of the *assets*, *devices*, and *software* being used. Organizations are at risk of security gaps and struggle to meet industry standards without a centralized system for tracking assets and assessing risks.

Furthermore, the manual methods of assessing risks, prioritizing them, and deploying patches are not just time-consuming but also susceptible to mistakes.

IntraSec seeks to address these issues by providing a centralized, automated, and client-free solution that can effectively monitor assets, conduct real-time risk evaluations and deploy patches to improve security and assets management.

1.2 Scope

The scope of **IntraSec** encompasses the development and deployment of a client-less risk assessment and asset management system specifically designed for **internal networks**.

It includes discovering and inventorying all devices, services, and software within the network. The system continuously tracks these assets in real-time, ensuring up-to-date information. *IntraSec* integrates with the *NIST* vulnerability database to automatically assess risks associated with network assets and assigns a risk score based on the severity

of detected vulnerabilities.

It generates clear, actionable reports that help network administrators prioritize risk mitigation efforts. One of the key functionalities is the automatic deployment of patches to address vulnerabilities.In cases where automatic patching isn't available, administrators are alerted for manual intervention.

Additionally, the system provides continuous monitoring, real-time alerts, and notifications for critical security issues.

1.3 Modules

1.3.1 Module 1: Assets Inventory

The Assets Inventory module is responsible for discovering and tracking all devices, software, and services within the internal network. It ensures up-to-date information on network assets by continuously monitoring changes in real-time.

1. **Front-end:** User interface for viewing and managing asset details.
2. **Back-end:** Active Directory integration for centralized asset management.
3. **Back-end:** Real-time discovery of hosts, software, and services.
4. **Back-end:** Continuous tracking and updating of asset details in *Postgresql*

1.3.2 Module 2: Risk Assessment

The Risk Assessment module analyzes the assets identified in the inventory, assessing them for potential vulnerabilities using the *NIST* vulnerability database and *ElasticSearch*. It assigns a risk score based on severity, helping to prioritize remediation efforts.

1. **Front-end:** Dashboard for displaying risk assessment results and scores.
2. **Back-end:** Integration with the *NIST* Vulnerability Database.
3. **Back-end:** Vulnerability scanning using *ElasticSearch*.
4. **Back-end:** Assignment of risk scores to each asset based on vulnerability severity.
5. **Back-end:** Prioritization of remediation efforts for high-risk assets.

1.3.3 Module 3: Report Generation

After risk assessment, *IntraSec* generates detailed, easy-to-understand reports. These reports provide an overview of the network's security posture, including identified risks and recommended mitigation steps.

1. **Front-end:** Interface for viewing and exporting generated reports.
2. **Back-end:** Standardized risk scoring for consistency in evaluations.
3. **Back-end:** Detailed analysis of detected vulnerabilities and their impact.
4. **Back-end:** Recommendations for remediation of critical threats.
5. **Back-end:** Exportable reports for documentation and audits.

1.3.4 Module 4: CVE Prioritization

The CVE Prioritization module enables administrators to prioritize Common Vulnerabilities and Exposures (CVEs) identified during the risk assessment and select hosts for remediation. It leverages data from multiple APIs to assess vulnerability severity and exploitation likelihood, ensuring efficient and targeted vulnerability management.

1. **Front-end:** Interface for administrators to view and prioritize detected CVEs based on severity, exploitability, and asset criticality.
2. **Back-end:** Integration with APIs (NIST NVD, FIRST.org EPSS, CISA KEV, VulnCheck) to fetch and analyze CVE data, including severity scores, exploit probability, and known exploitation status.
3. **Back-end:** Functionality to prioritize hosts for remediation based on CVE prioritization and network asset details stored in Elasticsearch.
4. **Back-end:** Notifications to administrators for high-priority CVEs or hosts requiring immediate attention.

1.3.5 Module 5: Alerts and Notifications

This module provides real-time alerts and notifications regarding security issues or vulnerabilities that require immediate attention. It ensures that administrators are aware of critical threats as they arise.

1. **Front-end:** User interface for viewing alerts and notifications.

2. **Back-end:** Real-time alerts for vulnerabilities and security incidents.
3. **Back-end:** Customizable thresholds for alert generation.
4. **Back-end:** Notifications for vulnerabilities requiring urgent attention.

1.4 User Classes and Characteristics

Table 1.1: User Classes and Their Characteristics

User Class	Description
Network Security Team	This team is responsible for monitoring the internal networks. They will use IntraSec to assess vulnerabilities and take proactive measures to secure the network. Their focus is on threat detection and mitigation.
IT Administrators	They oversee the installation and upkeep of IntraSec. Their role includes configuring the system and ensuring it integrates smoothly with existing network management tools. They are crucial for the day-to-day operation of the system.
System Administrators	These users manage the technical side of the network, including patch deployments. They will rely on IntraSec to automate patch management and improve efficiency in addressing vulnerabilities.

Chapter 2

Project Requirements

2.1 Use-case Diagram

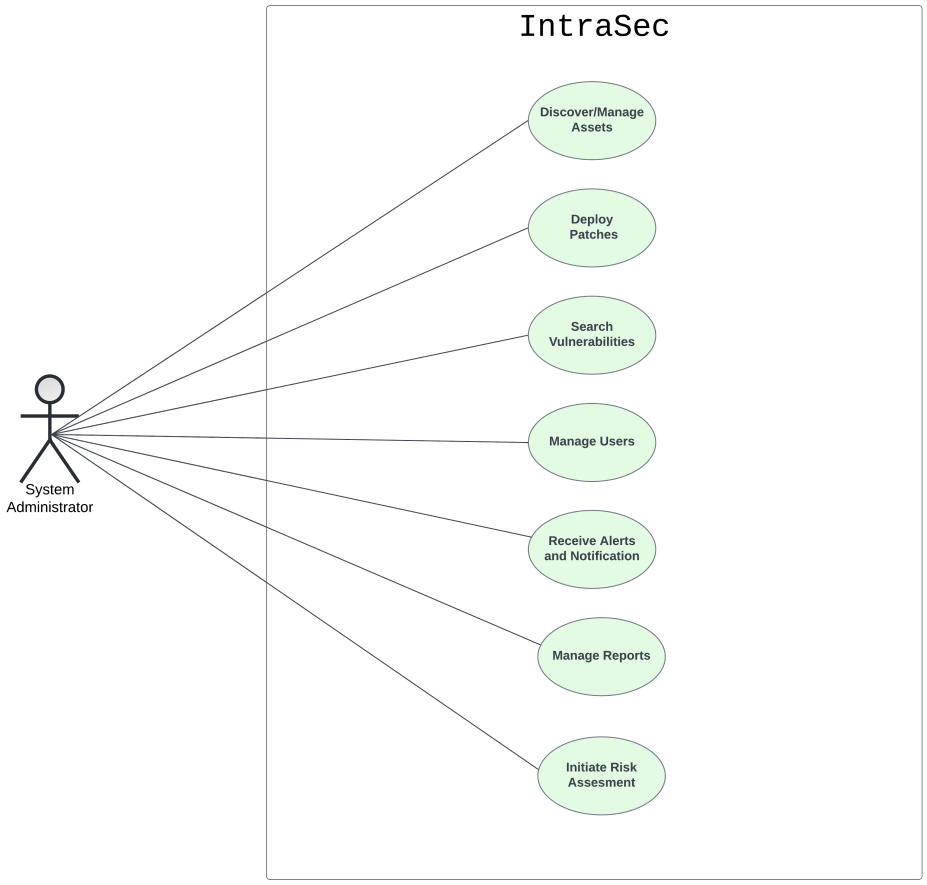

Figure 2.1: Use-Case Diagram

2.2 High-Level Use Case Table

Use Case	Actor	Type	Description
Discover/Manage Assets	System Administrator	Primary	Discover all assets on the network and manage the asset inventory, ensuring an up-to-date record of devices.
Deploy Patches	System Administrator	Primary	Deploy patches across the network to address identified vulnerabilities and ensure asset security.
Search Vulnerabilities	System Administrator	Supporting	Search the vulnerability database for specific assets, types of vulnerabilities, or other search parameters.
Manage Users	System Administrator	Supporting	Manage user roles, permissions, and access to the IntraSec system.
Receive Alerts/Notifications	System Administrator	Supporting	Receive alerts and notifications about critical vulnerabilities or abnormal activities requiring immediate attention.
Manage Reports	System Administrator	Primary	Generate, view, and manage risk and vulnerability reports for system review and decision-making.
Initiate Risk Assessment	System Administrator	Primary	Conduct a risk assessment by analyzing network assets for potential vulnerabilities based on predefined criteria.

Table 2.1: High-Level Use Case Table for IntraSec with System Administrator

2.3 Expanded Use Cases

2.3.1 UC-01: Discover/Manage Assets

Table 2.2: Expanded Use Case: Discover/Manage Assets

Attribute	**Details**
Use Case ID	UC-01
Use Case Name	Discover/Manage Assets
Actor	System Administrator
Type	Primary
Description	This use case allows the System Administrator to discover all assets connected to the network, maintain an up-to-date inventory, and manage asset details effectively. It involves scanning the network for devices, identifying their types, statuses, and configurations, and making necessary updates to the asset database.
Preconditions	• The System Administrator has access to the IntraSec application. • The network is operational and the administrator has the necessary permissions to scan for assets.
Postconditions	• The asset inventory is updated with the latest discovered devices. • The asset management system reflects any changes made during the process.

<table>
<tr><th>Attribute</th><th>Details</th></tr>
<tr><td>Basic Flow</td><td>
<table>
<tr><th>User Action</th><th>System Response</th></tr>
<tr><td>The System Administrator logs into the IntraSec application.</td><td>The system verifies the credentials and grants access to the application.</td></tr>
<tr><td>The administrator navigates to the "Asset Management" section.</td><td>The system displays the asset management interface with options to discover, view, and manage assets.</td></tr>
<tr><td>The administrator initiates an asset discovery scan.</td><td>The system starts scanning the network for connected devices.</td></tr>
<tr><td>The administrator waits for the scan to complete.</td><td>The system identifies all connected assets and compiles their details.</td></tr>
<tr><td>The system presents the list of discovered assets to the administrator.</td><td>The administrator reviews the list, which includes asset types, statuses, and configurations.</td></tr>
<tr><td>The administrator selects an asset to view its details.</td><td>The system retrieves and displays detailed information about the selected asset.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>

If no assets are found: The system notifies the administrator and prompts to check the network connection or permissions.
If the administrator chooses to cancel the scan: The system terminates the discovery process and returns to the asset management interface.

</td></tr>
<tr><td>Exceptions</td><td>

The system encounters network errors during the discovery scan.
The administrator does not have sufficient permissions to access certain asset details.

</td></tr>
<tr><td>Frequency of Use</td><td>This use case is performed regularly as part of network management tasks, typically initiated weekly or after significant changes to the network.</td></tr>
</table>

Attribute	Details
Special Requirements	• The discovery process should comply with network security protocols. • The system should provide a user-friendly interface for reviewing and managing assets.

2.3.2 UC-02: Prioritize CVEs

Table 2.3: Expanded Use Case: Prioritize CVEs

Attribute	**Details**
Use Case ID	UC-02
Use Case Name	Prioritize CVEs
Actor	System Administrator
Type	Primary
Description	This use case enables the System Administrator to prioritize Common Vulnerabilities and Exposures (CVEs) identified during risk assessment and select hosts for remediation. It leverages data from NIST NVD, FIRST.org EPSS, CISA KEV, VulnCheck, and Elasticsearch to assess CVE severity, exploitability, and asset criticality, ensuring efficient vulnerability management.
Preconditions	• The System Administrator has access to the IntraSec application. • CVEs have been detected and stored in Elasticsearch via risk assessment processes. • API connections to NIST NVD, FIRST.org EPSS, CISA KEV, and VulnCheck are operational.
Postconditions	• CVEs are prioritized based on severity, exploitability, and asset criticality. • Hosts are prioritized for remediation, with actions logged in the system. • Notifications are sent for high-priority CVEs or hosts requiring immediate attention.

<table>
<tr><th>Attribute</th><th>Details</th></tr>
<tr><td>Basic Flow</td><td>
<table>
<tr><th>User Action</th><th>System Response</th></tr>
<tr><td>The System Administrator logs into the IntraSec application.</td><td>The system verifies the credentials and grants access to the application.</td></tr>
<tr><td>The administrator navigates to the "CVE Prioritization" section.</td><td>The system displays a list of detected CVEs with details from NIST NVD, EPSS, CISA KEV, and VulnCheck, including severity scores, exploit probabilities, and known exploitation status.</td></tr>
<tr><td>The administrator selects CVEs to prioritize based on provided metrics.</td><td>The system ranks the selected CVEs using data such as CVSS scores, EPSS probabilities, and CISA KEV status, and displays asset criticality from Elasticsearch.</td></tr>
<tr><td>The administrator prioritizes hosts for remediation based on CVE rankings and asset details.</td><td>The system assigns remediation priorities to hosts and updates the prioritization status in Elasticsearch.</td></tr>
<tr><td>The administrator reviews the prioritization summary.</td><td>The system provides a summary report of prioritized CVEs and hosts, including any high-priority alerts.</td></tr>
<tr><td>The system sends notifications for high-priority actions.</td><td>The administrator receives alerts for CVEs or hosts requiring immediate attention and confirms the prioritization process.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>

If API data is unavailable: The system uses cached CVE data from Elasticsearch and notifies the administrator to check API connections.
If the administrator cancels the prioritization: The system discards unsaved changes and reverts to the previous prioritization state.

</td></tr>
</table>

Attribute	Details
Exceptions	• The system encounters missing or incomplete CVE data from APIs. • The administrator lacks permissions to prioritize certain CVEs or hosts.
Frequency of Use	This use case is performed regularly as part of vulnerability management, typically after risk assessments or when new CVEs are detected.
Special Requirements	• The prioritization process must comply with organizational vulnerability management policies. • The system must log all prioritization actions and API data retrievals for auditing purposes.

2.3.3 UC-03 : Search Vulnerabilities

Table 2.4: Expanded Use Case: Search Vulnerabilities

Attribute	**Details**
Use Case ID	UC-03
Use Case Name	Search Vulnerabilities
Actor	System Administrator
Type	Primary
Description	This use case allows the System Administrator to search for known vulnerabilities in the asset inventory based on various criteria such as severity, asset type, and status. It aids in identifying and prioritizing vulnerabilities for remediation.
Preconditions	• The System Administrator has access to the IntraSec application. • The asset inventory is populated with assets and associated vulnerabilities.
Postconditions	• A list of vulnerabilities matching the search criteria is displayed for review. • The system logs the search query for auditing purposes.

<table>
<tr><th>Attribute</th><th>Details</th></tr>
<tr><td>Basic Flow</td><td>
<table>
<tr><th>User Action</th><th>System Response</th></tr>
<tr><td>The System Administrator logs into the IntraSec application.</td><td>The system verifies the credentials and grants access to the application.</td></tr>
<tr><td>The administrator navigates to the "Vulnerability Management" section.</td><td>The system displays the search interface for vulnerabilities.</td></tr>
<tr><td>The administrator specifies search criteria (e.g., severity, asset type).</td><td>The system processes the search parameters and prepares to query the database.</td></tr>
<tr><td>The administrator initiates the search for vulnerabilities.</td><td>The system retrieves and displays a list of vulnerabilities matching the search criteria.</td></tr>
<tr><td>The administrator reviews the list of vulnerabilities.</td><td>The system presents detailed information about each vulnerability, including severity and affected assets.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>

If no vulnerabilities are found: The system notifies the administrator and suggests refining the search criteria.
If the administrator chooses to cancel the search: The system stops the search process and returns to the vulnerability management interface.

</td></tr>
<tr><td>Exceptions</td><td>

The system encounters issues accessing the vulnerability database.
The administrator lacks the necessary permissions to view certain vulnerabilities.

</td></tr>
<tr><td>Frequency of Use</td><td>This use case is performed regularly as part of security assessments, typically conducted after new vulnerabilities are disclosed or during scheduled audits.</td></tr>
</table>

Attribute	Details
Special Requirements	• The search process should comply with data privacy and security policies. • The system should provide an intuitive interface for specifying search criteria and reviewing results.

2.3.4 UC-04 : Manage Users

Table 2.5: Expanded Use Case: Manage Users

Attribute	**Details**
Use Case ID	UC-04
Use Case Name	Manage Users
Actor	System Administrator
Type	Primary
Description	This use case allows the System Administrator to manage user accounts, including adding, modifying, or deleting user information, assigning roles and permissions, and ensuring secure access control within the IntraSec system.
Preconditions	• The System Administrator has the necessary permissions to manage user accounts. • The system is operational and user data is accessible.
Postconditions	• User accounts are updated according to the actions performed by the administrator. • Role-based access control is correctly applied to user accounts.

<table>
<tr><th>Attribute</th><th>Details</th></tr>
<tr><td>Basic Flow</td><td>
<table>
<tr><th>User Action</th><th>System Response</th></tr>
<tr><td>The System Administrator logs into the IntraSec application.</td><td>The system verifies the credentials and grants access to the application.</td></tr>
<tr><td>The administrator navigates to the "User Management" section.</td><td>The system displays the current list of user accounts.</td></tr>
<tr><td>The administrator selects the option to add, modify, or delete a user.</td><td>The system provides the necessary input forms for the selected action.</td></tr>
<tr><td>The administrator enters or updates user details (e.g., username, role).</td><td>The system validates the input and saves the user data.</td></tr>
<tr><td>The administrator assigns roles and permissions to the user.</td><td>The system applies role-based access control and updates the user account accordingly.</td></tr>
<tr><td>The system confirms the completion of the user management task and displays a summary.</td><td>The administrator reviews and confirms the changes made to the user account.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>

If the administrator attempts to delete a user with active sessions: The system prompts the administrator to terminate the sessions before proceeding.
If the administrator assigns invalid roles or permissions: The system notifies the administrator and suggests valid options.

</td></tr>
<tr><td>Exceptions</td><td>

The administrator lacks permissions to modify certain user accounts.
The system encounters data integrity issues during the update process.

</td></tr>
<tr><td>Frequency of Use</td><td>This use case is performed as needed when user accounts require updates, or when new users are added to the system.</td></tr>
</table>

Attribute	Details
Special Requirements	• All changes to user accounts must be logged for auditing purposes. • The system should enforce strong password policies during account creation.

2.3.5 UC-05 : Receive Alerts and Notification

Table 2.6: Expanded Use Case: Receive Alerts and Notifications

Attribute	**Details**
Use Case ID	UC-05
Use Case Name	Receive Alerts and Notifications
Actor	System Administrator
Type	Primary
Description	This use case allows the System Administrator to receive real-time alerts and notifications about critical vulnerabilities, security threats, or system updates. It includes configurable alert settings, immediate notifications for critical issues, and a dashboard view of all current alerts.
Preconditions	• The System Administrator has the necessary access to the IntraSec application. • Alerts and notifications are configured in the system based on network monitoring and risk assessment rules.
Postconditions	• The System Administrator is notified about the relevant security events. • The system logs all alerts and notifications for review and audit purposes.

<table>
<tr><th>Attribute</th><th>Details</th></tr>
<tr><td>Basic Flow</td><td>
<table>
<tr><th>User Action</th><th>System Response</th></tr>
<tr><td>The System Administrator logs into the IntraSec application.</td><td>The system verifies credentials and provides access to the dashboard.</td></tr>
<tr><td>The administrator configures the alert settings for specific vulnerabilities or security events.</td><td>The system saves the configuration and sets up notifications based on the defined rules.</td></tr>
<tr><td>A security threat or system event occurs that triggers an alert.</td><td>The system generates an alert and sends notifications through the configured channels (email, SMS, dashboard).</td></tr>
<tr><td>The administrator receives the alert in real-time.</td><td>The system displays the alert with details on the dashboard and sends any additional notifications as configured.</td></tr>
<tr><td>The administrator views the alert details and decides on further actions.</td><td>The system provides detailed information about the event, including risk severity and suggested remediation steps.</td></tr>
<tr><td>The administrator marks the alert as acknowledged or resolved.</td><td>The system logs the response and updates the alert status.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>

If the system is unable to deliver alerts due to a communication issue: The system retries sending notifications and logs the failure for further investigation.
If the administrator configures incorrect alert thresholds: The system notifies the administrator of the misconfiguration and suggests corrective actions.

</td></tr>
<tr><td>Exceptions</td><td>

The system encounters connectivity issues while sending notifications.
The administrator lacks the necessary permissions to modify alert settings.

</td></tr>
</table>

Attribute	Details
Frequency of Use	This use case is performed continuously as alerts are generated based on real-time monitoring and security events.
Special Requirements	• The system should support configurable notification channels (email, SMS, etc.). • All alerts and notifications must be logged and accessible for audits.

2.3.6 UC-06 : Manage Reports

Table 2.7: Expanded Use Case: Manage Reports

Attribute	**Details**
Use Case ID	UC-06
Use Case Name	Manage Reports
Actor	System Administrator
Type	Primary
Description	This use case allows the System Administrator to generate, view, and export security reports based on vulnerability assessments and patch deployment statuses. It includes generating detailed reports, analyzing vulnerabilities, and exporting reports for documentation or audits.
Preconditions	• The System Administrator has access to the IntraSec application. • Risk assessments and patch deployments have been performed, and relevant data is available.
Postconditions	• A detailed report is generated with vulnerability analysis, risk scores, and recommended actions. • The report can be exported in various formats (e.g., PDF, CSV) for future reference or audits.

<table>
<tr><th>Attribute</th><th>Details</th></tr>
<tr><td>Basic Flow</td><td>
<table>
<tr><th>User Action</th><th>System Response</th></tr>
<tr><td>The System Administrator logs into the IntraSec application.</td><td>The system verifies credentials and grants access to the dashboard.</td></tr>
<tr><td>The administrator navigates to the "Reports" section.</td><td>The system displays the list of available reports and options to generate new reports.</td></tr>
<tr><td>The administrator selects the type of report (e.g., vulnerability report, patch deployment report).</td><td>The system gathers relevant data based on the selected report type and prepares it for generation.</td></tr>
<tr><td>The administrator configures report parameters (e.g., date range, asset selection).</td><td>The system applies the selected filters and generates a detailed report.</td></tr>
<tr><td>The system generates the report.</td><td>The administrator views the generated report on the dashboard.</td></tr>
<tr><td>The administrator exports the report in the desired format (e.g., PDF, CSV).</td><td>The system generates the report in the chosen format and provides a download link.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>

If the report fails to generate: The system notifies the administrator and provides error details for troubleshooting.
If the administrator chooses to cancel the report generation: The system halts the process and discards the incomplete report.

</td></tr>
<tr><td>Exceptions</td><td>

The system encounters missing or incomplete data for report generation.
The administrator lacks the necessary permissions to access certain reports.

</td></tr>
<tr><td>Frequency of Use</td><td>This use case is performed regularly after vulnerability assessments or patch deployments. Reports may also be generated during audits or compliance checks.</td></tr>
</table>

Attribute	Details
Special Requirements	• The system should support multiple export formats (e.g., PDF, CSV) for reports. • Detailed logging of report generation and export actions should be available for auditing purposes.

2.3.7 UC-07 :Initiate Risk Assessment

Table 2.8: Expanded Use Case: Initiate Risk Assessment

Attribute	**Details**
Use Case ID	UC-07
Use Case Name	Initiate Risk Assessment
Actor	System Administrator
Type	Primary
Description	This use case allows the System Administrator to initiate a risk assessment of networked assets. The risk assessment analyzes identified assets, assesses their vulnerabilities using the *NIST* Vulnerability Database, and assigns risk scores based on the severity of vulnerabilities.
Preconditions	• The System Administrator has access to the IntraSec application. • Assets have been identified and listed in the asset inventory. • The system is connected to the *NIST* Vulnerability Database (ElasticSearch)
Postconditions	• Vulnerabilities in the networked assets are identified and assigned risk scores. • A detailed risk assessment report is generated, highlighting critical vulnerabilities.

<table>
<tr><th>Attribute</th><th>Details</th></tr>
<tr><td>Basic Flow</td><td>
<table>
<tr><th>User Action</th><th>System Response</th></tr>
<tr><td>The System Administrator logs into the IntraSec application.</td><td>The system verifies credentials and grants access to the dashboard.</td></tr>
<tr><td>The administrator navigates to the "Risk Assessment" section.</td><td>The system displays a list of available assets and the option to initiate a new risk assessment.</td></tr>
<tr><td>The administrator selects the assets to be assessed.</td><td>The system loads the selected assets and their details for vulnerability analysis.</td></tr>
<tr><td>The administrator initiates the risk assessment.</td><td>The system queries the NIST Vulnerability Database (ElasticSearch) and begins scanning the assets for known vulnerabilities.</td></tr>
<tr><td>The system assigns risk scores based on the identified vulnerabilities.</td><td>The administrator reviews the progress as the system calculates risk scores based on vulnerability severity.</td></tr>
<tr><td>The system completes the risk assessment and provides a summary report.</td><td>The administrator views the detailed risk assessment report, which includes vulnerabilities, risk scores, and recommended remediation steps.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>

If the risk assessment process is interrupted: The system notifies the administrator and provides an option to resume or restart the process.
If the administrator cancels the risk assessment: The system halts the process and discards any partially collected data.

</td></tr>
<tr><td>Exceptions</td><td>

The system encounters network issues or fails to connect to the NIST Vulnerability Database (ElasticSearch).
The system is unable to assess some assets due to missing or incomplete information.

</td></tr>
</table>

Attribute	Details
Frequency of Use	This use case is typically performed after significant changes to the network, during routine security assessments, or following a security audit.
Special Requirements	• The system must maintain a reliable connection to the *NIST* Vulnerability Database (ElasticSearch). • The system should generate detailed logs of the assessment for future audits or troubleshooting.

2.4 Functional Requirements

2.4.1 Module 1: Assets Inventory

The Assets Inventory module is responsible for discovering and tracking all devices, software, and services within the internal network.

1. The system shall discover and display all devices, software, and services in real-time.
2. The system shall allow users to view, filter, and manage the details of assets through a front-end interface.
3. The system shall continuously track changes in asset details, including new additions or removals, and update them in the PostgreSQL database.
4. The system shall integrate with Active Directory to centralize asset management.

2.4.2 Module 2: Risk Assessment

The Risk Assessment module assesses assets for potential vulnerabilities and assigns a risk score to prioritize remediation efforts.

1. The system shall perform vulnerability scanning of assets using *ElasticSearch*.
2. The system shall continuously sync with the NIST Vulnerability Database API to ensure the most up-to-date vulnerability information.
3. The system shall calculate and assign risk scores based on the severity of detected vulnerabilities.
4. The system shall provide a dashboard to display risk scores, vulnerability details, and recommendations for remediation.

2.4.3 Module 3: Report Generation

The Report Generation module provides detailed reports on the network's security posture, risk assessment results, and recommended mitigation steps.

1. The system shall generate standardized reports that include identified vulnerabilities, their severity, and the corresponding risk score.

2. The system shall provide exportable reports for auditing and documentation purposes.
3. The system shall generate detailed analysis reports with recommendations for remediation of critical threats.
4. The system shall allow users to view, filter, and export reports through a front-end interface.

2.4.4 Module 4: Patch Deployment

The Patch Deployment module automates the application of patches to fix vulnerabilities and protect the network from known threats.

1. The system shall automate the deployment of patches for identified vulnerabilities.
2. The system shall provide users with an interface to monitor the status of patch deployment.
3. The system shall notify users if manual intervention is required when automatic patch deployment fails.
4. The system shall report the progress and status of patch deployment.

2.4.5 Module 5: Alerts and Notifications

The Alerts and Notifications module ensures that administrators are informed in real-time of any critical security issues or vulnerabilities that require attention.

1. The system shall generate real-time alerts for detected vulnerabilities or security incidents.
2. The system shall allow users to customize alert thresholds for specific types of vulnerabilities or incidents.
3. The system shall provide an interface for users to view active alerts and notifications.
4. The system shall send notifications for vulnerabilities that require immediate action.

2.5 Non-Functional Requirements

This section outlines the quantifiable non-functional requirements that specify the system's quality attributes, ensuring reliability, usability, performance, and security.

2.5.1 Reliability

- The system shall aim to achieve an up-time of at least **95%** during operational hours, with scheduled maintenance windows allowed.
- In case of a system failure, the system shall log the event and notify administrators within **5 minutes** of detection.
- The system shall recover from critical failures and resume operation within **10 minutes** of the failure being detected.

2.5.2 Usability

- Users shall be able to perform key tasks such as viewing asset information and generating reports in **3 interactions or fewer**.
- **80%** of new users shall be able to navigate and complete common tasks without needing external documentation after **10 minutes** of interaction with the system.

2.5.3 Performance

- The system, based on preliminary estimates and assumptions about network size and performance, shall be able to generate an asset inventory report for up to **100 assets** within approximately **15 minutes**.
- The dashboard page shall fully load within **5 seconds** for **80%** of users on a **>10 Mbps** internet connection.
- Database queries shall return results within **5 seconds** for typical operations, such as retrieving asset details or vulnerability data.

2.5.4 Security

- **80%** of administrative users shall be required to use multi-factor authentication (MFA) for logging into the system, if implemented.

- The system shall maintain logs of access to sensitive data for at least **3 months**, accessible to authorized administrators only.
- The system shall lock user accounts after **5 consecutive failed login attempts**, with a notification sent to the administrator within **10 minutes**.

Chapter 3

System Overview

IntraSec is designed to provide a comprehensive solution for internal network risk assessment and asset management. Its primary objective is to facilitate the discovery, management, and remediation of vulnerabilities across various assets within an active directory domain network.

3.0.1 Functionality (Modules)

The system encompasses several key functionalities and each functionality is handled by its respective micro-services , which communicate via RESTful APIs. The main functionalities are listed below:

- ***Asset Discovery and Management:*** The system allows system administrators to discover all assets connected to the network, maintain an up-to-date inventory, and manage asset details effectively. This includes scanning the network for devices, identifying their types, statuses, and configurations, and making necessary updates to the asset database.
- ***Risk Assessment:*** IntraSec integrates with the Common Vulnerabilities and Exposures (CVE) and Common Platform Enumerations (CPE) databases to identify and assess vulnerabilities associated with the discovered assets. This assessment helps in recognizing potential security threats and prioritizing them for remediation.
- ***Patch Management:*** The system supports the deployment of patches to address identified vulnerabilities. It maintains a repository of patches associated with specific CVEs, facilitating timely updates to mitigate risks.
- ***Reporting:*** IntraSec generates detailed reports summarizing the vulnerabilities identified, the assets affected, and the recommended actions. This reporting functionality ensures that compliance with security protocols and standards is maintained.

This system overview provides a high-level understanding of IntraSec's objectives and functionality. The subsequent sections will delve deeper into the specific use cases, data structures, and implementation details of the system.

3.1 Architectural Design

3.1.1 Architecture Diagram

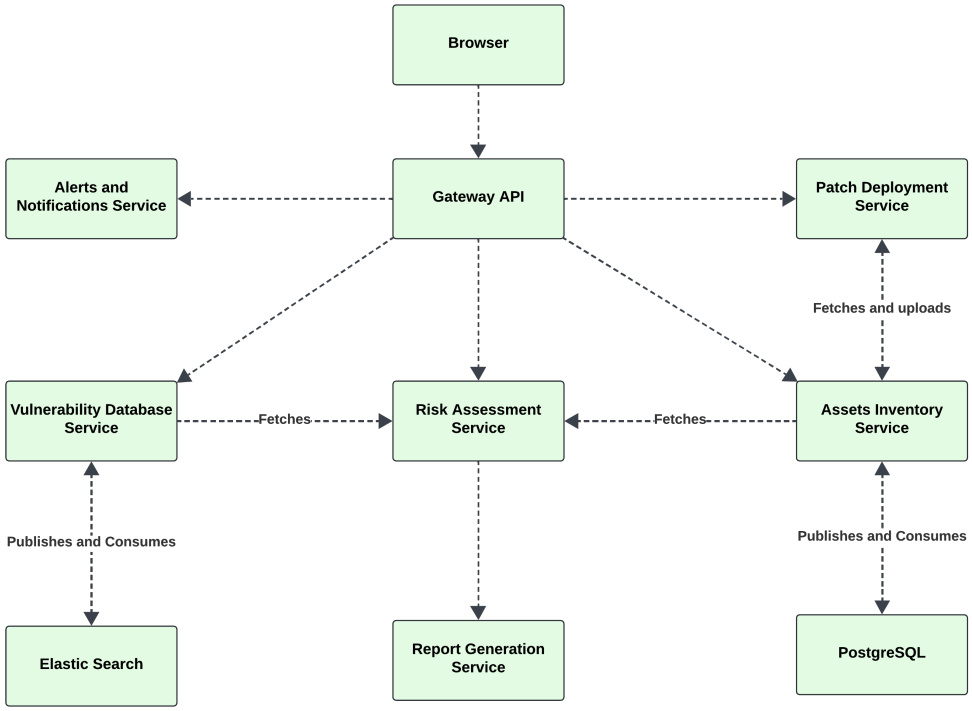

Figure 3.1: Architecture Diagram

The architecture design of **IntraSec** is based on Micro-services and their relations with each other.

Browser is the client-side which communicates with the back-end represented by ***Gateway API***. ***Gateway API*** then communicates with all services based on the input of the admin. Furthermore, Micro-services communicate with each other based on the required data.

For example, ***Risk Assessment Service*** requires data from ***Assets Inventory*** and ***Vulnerability database*** to perform the Risk Analysis. After the risk analysis, ***Risk Assessment Service*** commands the report generation service to generate easy to understand reports. On the other hand, ***Patch Deployment*** needs to communicate with the assets inventory service to fetch info and upload the possible patches.

3.1.2 Deployment Diagram

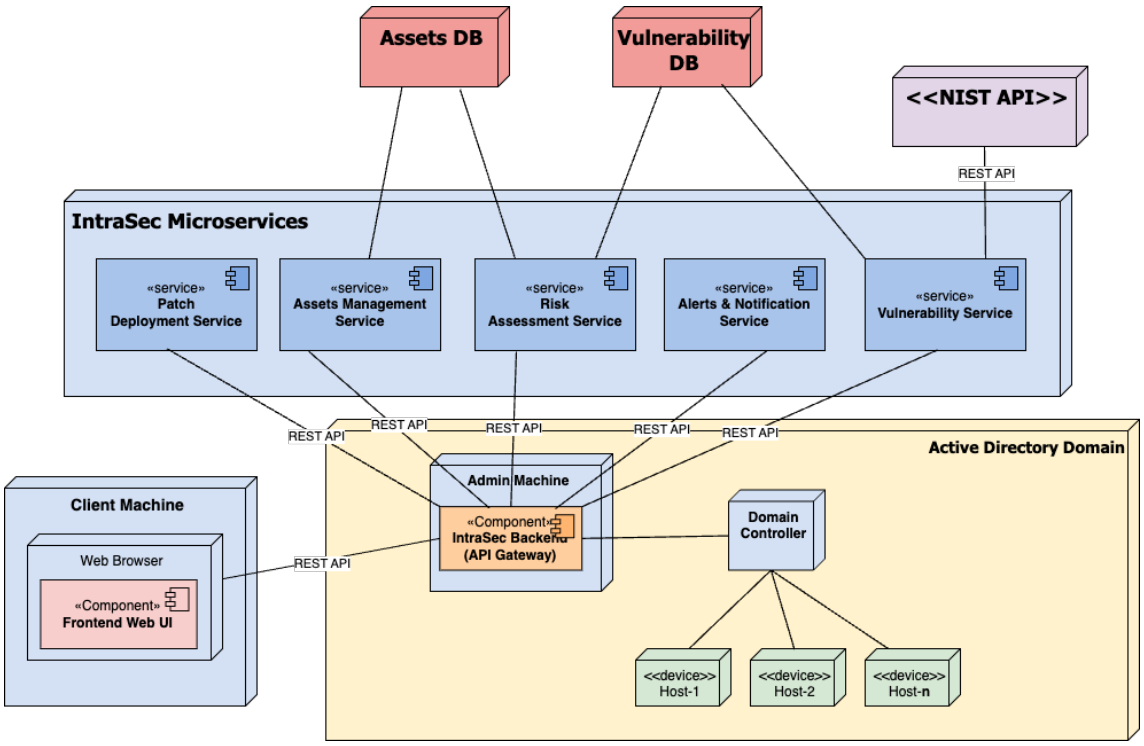

Figure 3.2: Deployment Diagram

3.2 Design Models

3.2.1 Data-flow Diagrams

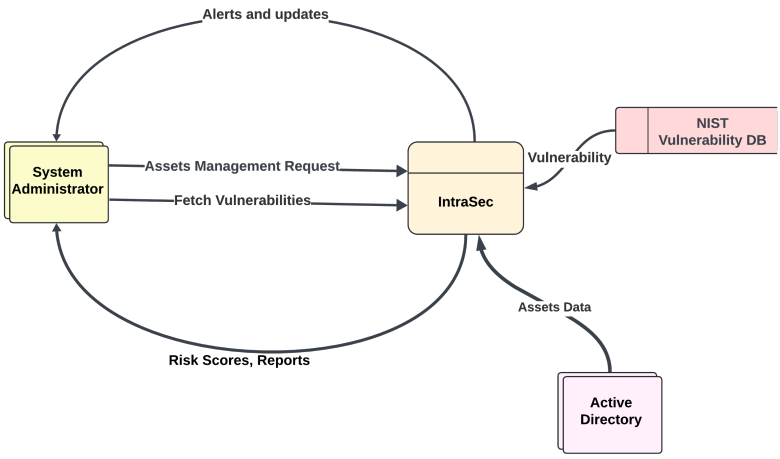

Figure 3.3: Data-flow Diagram - Level 0

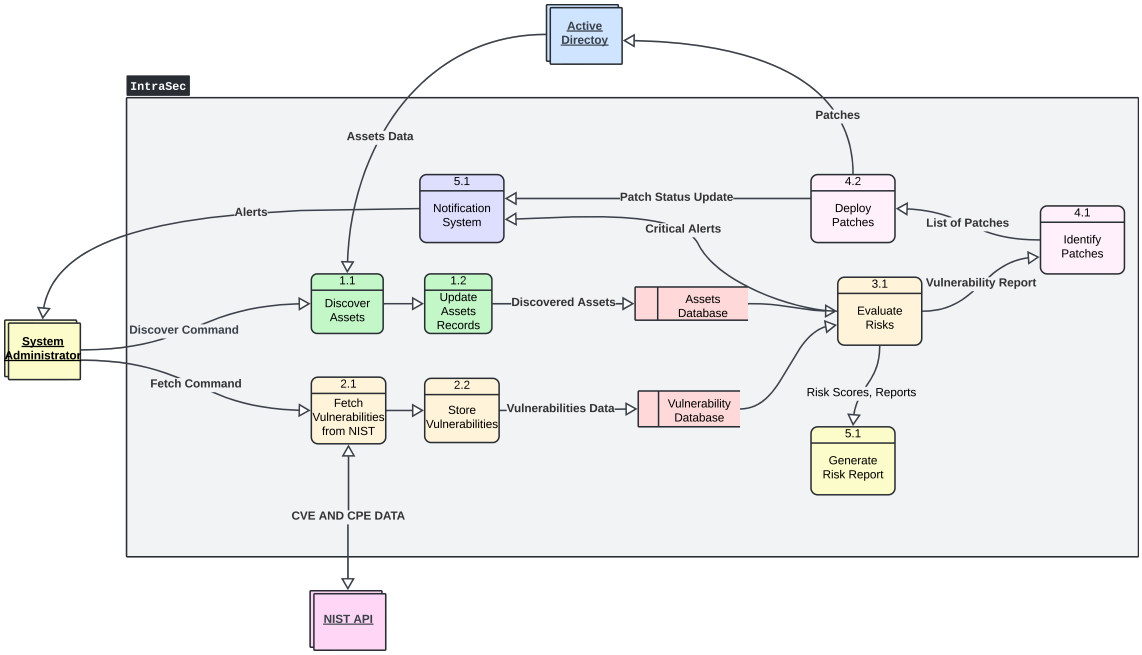

Figure 3.4: Data-flow Diagram - Level 2

3.2.2 Activity Diagrams

3.2.2.1 Discover Assets

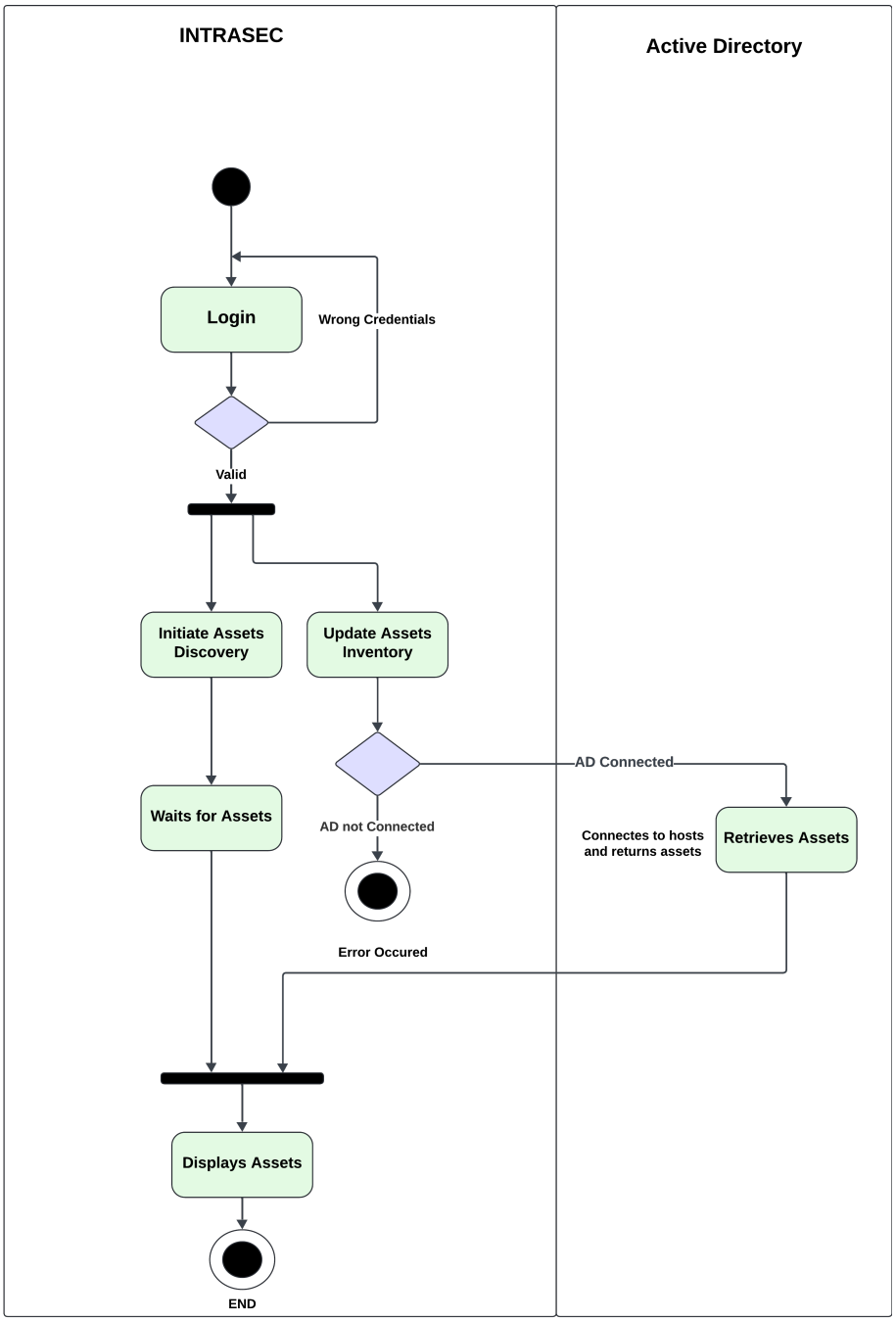

Figure 3.5: Discover Assets - Activity Diagram

3.2.2.2 Assess Risks

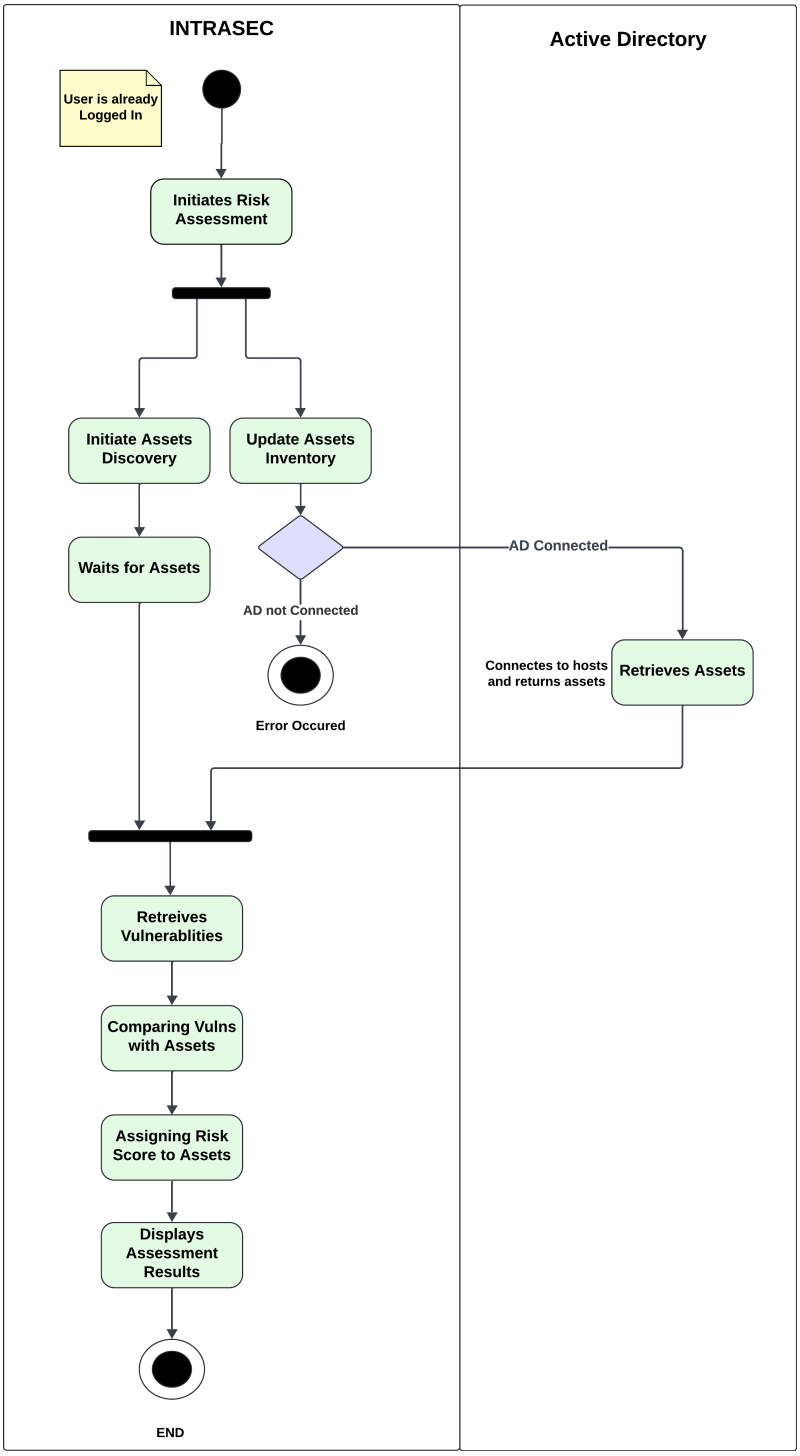

Figure 3.6: Assess Risk - Activity Diagram

3.2.2.3 Generate Report

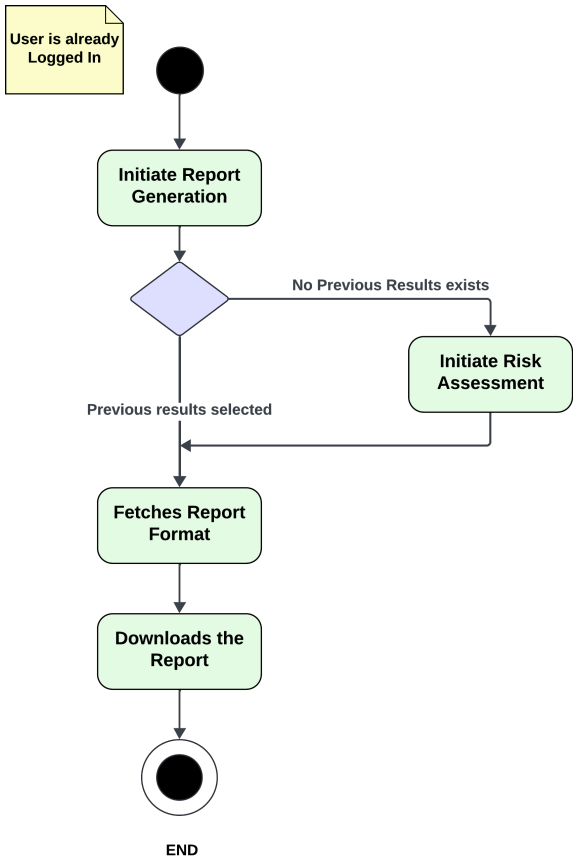

Figure 3.7: Generate Report - Activity Diagram

3.2.2.4 Deploy Patches

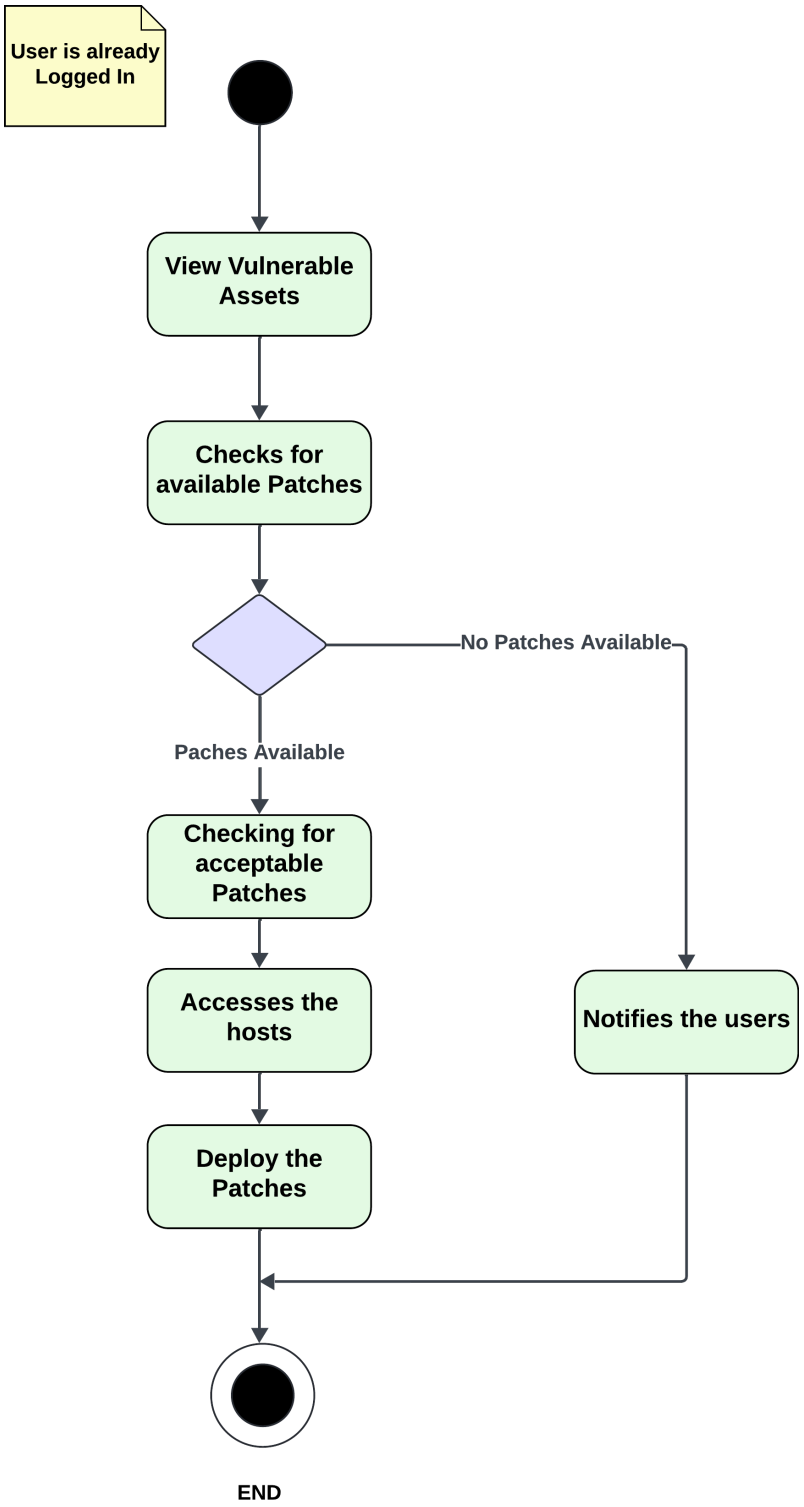

Figure 3.8: Deploy Patches - Activity Diagram

3.2.3 System-level Sequence Diagrams

3.2.3.1 Discover Assets

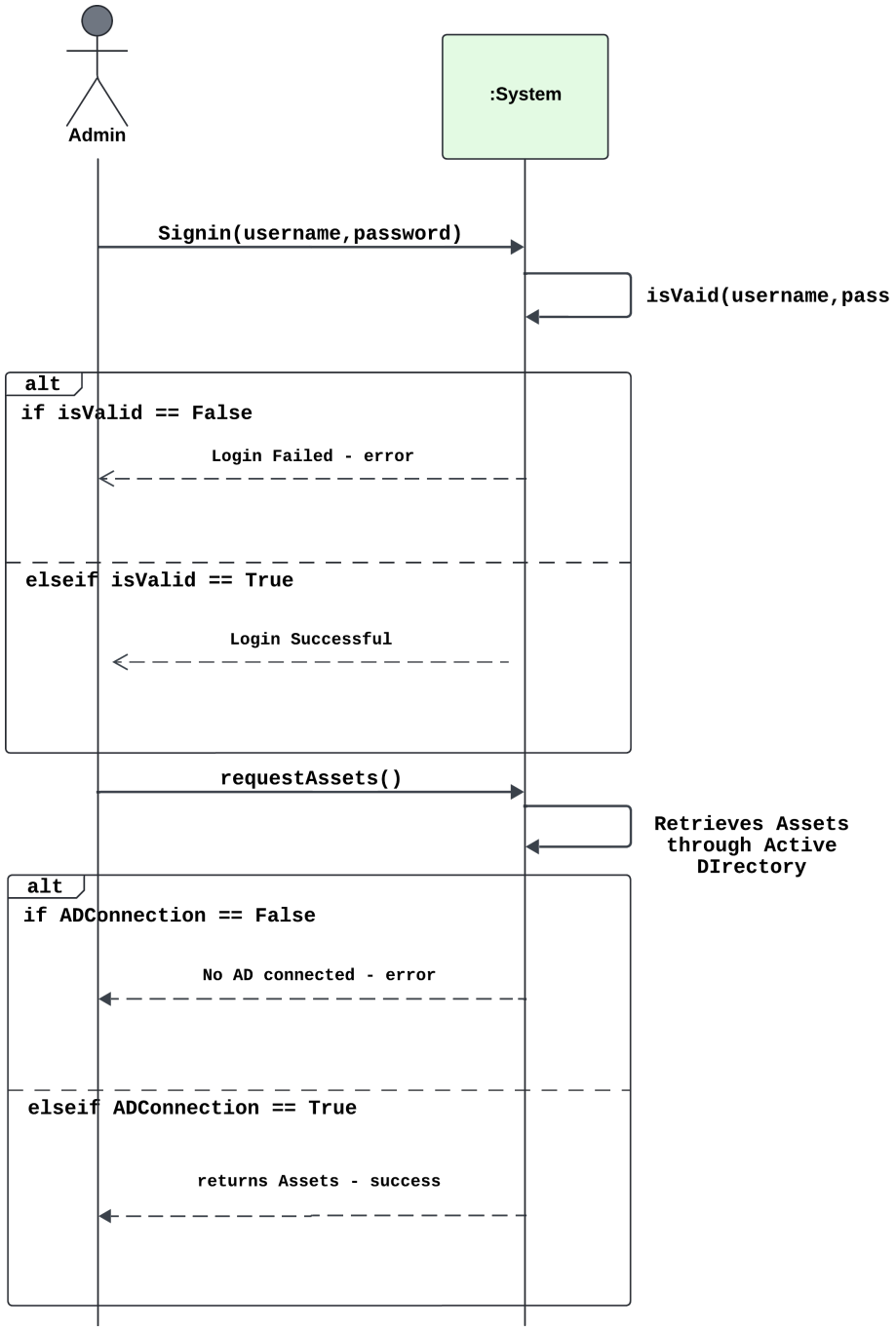

Figure 3.9: Discover Asset - System Sequence Diagram

3.2.3.2 Manage Assets

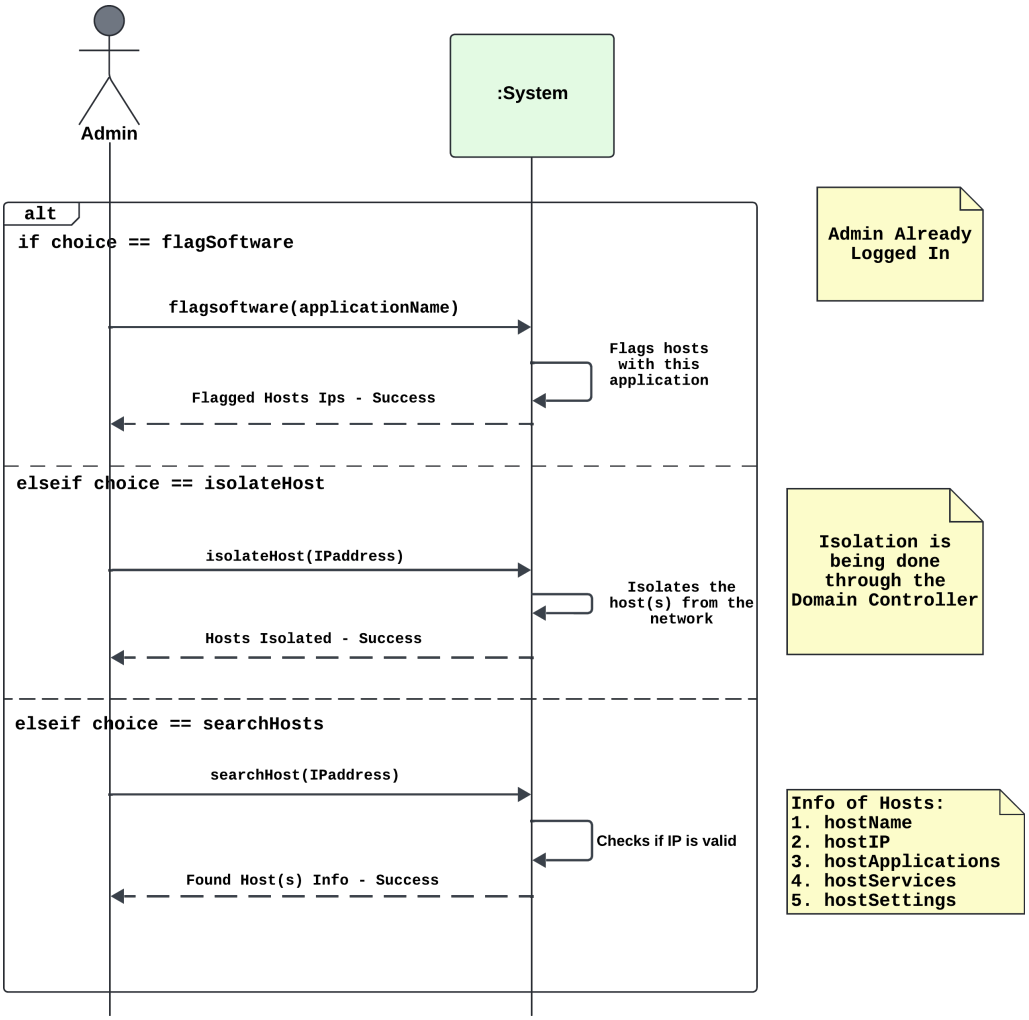

Figure 3.10: Manage Assets - System Sequence Diagram

3.2.3.3 Deploy Patches

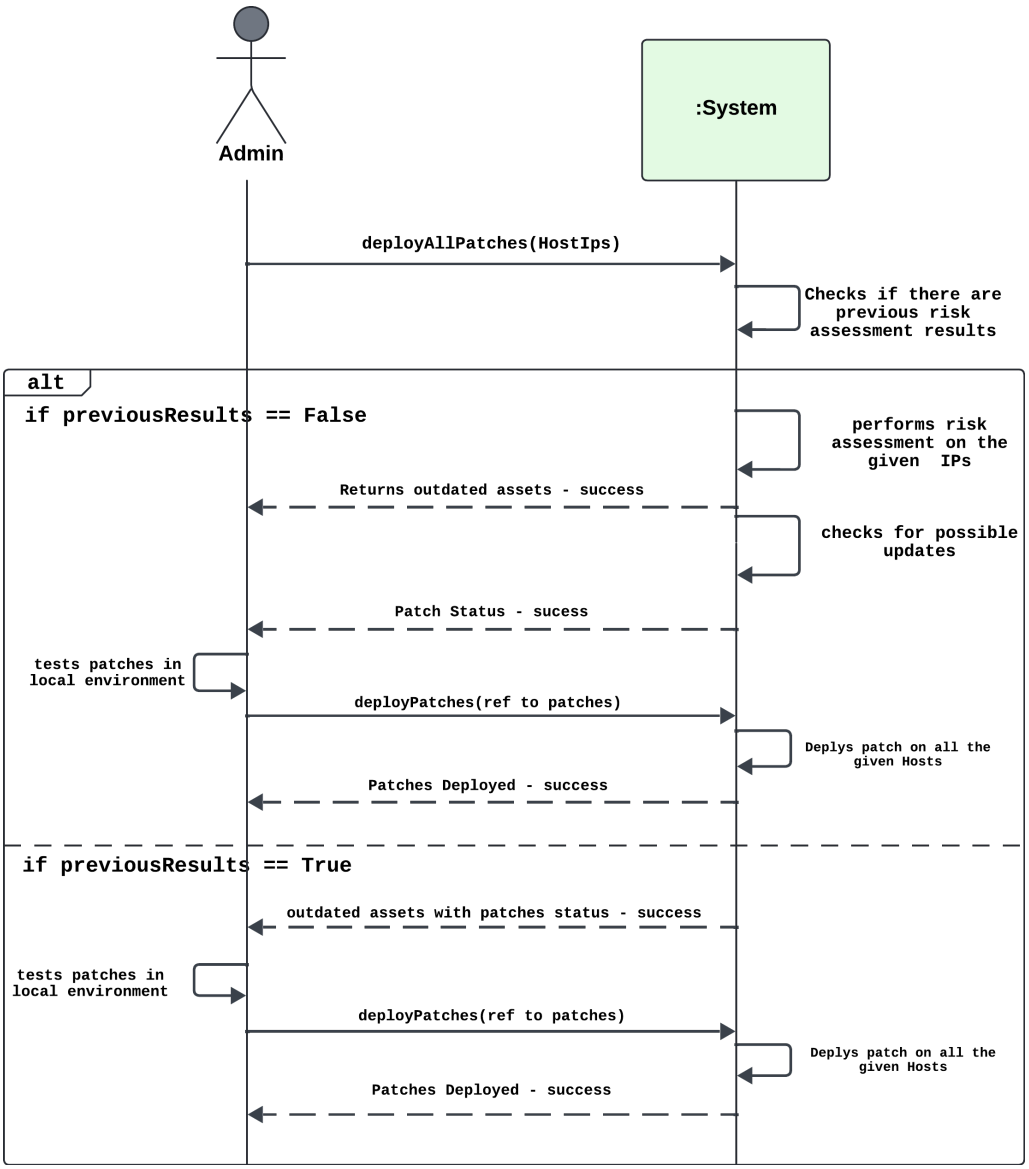

Figure 3.11: Deploy Patches - System Sequence Diagram

3.2.3.4 Assess Risks

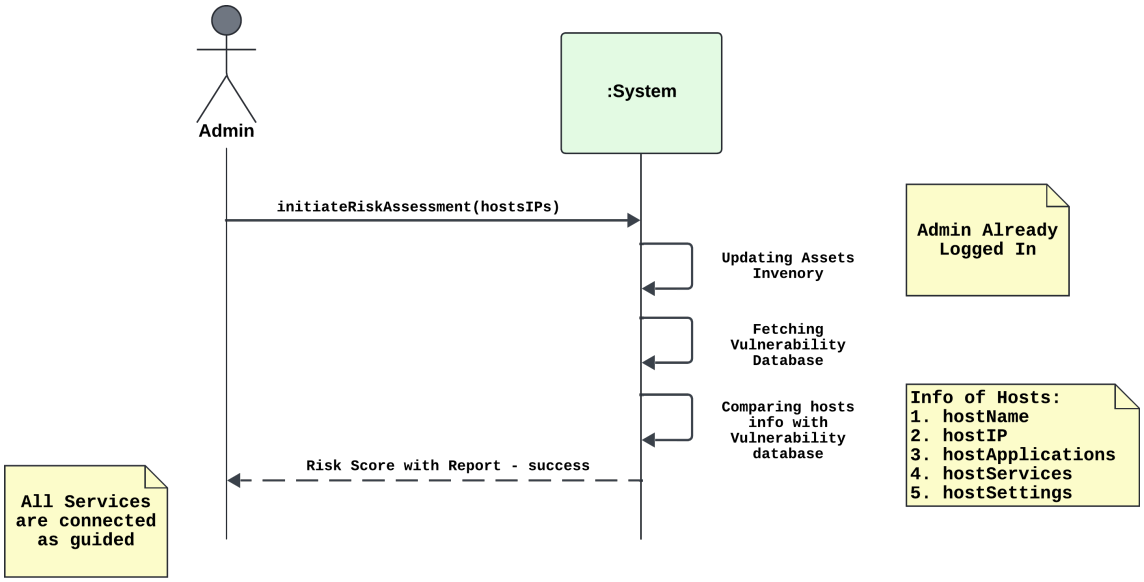

Figure 3.12: Assess Risks - System Sequence Diagram

3.3 Data Design

The information domain in our system consists of vulnerabilities represented as **Common Vulnerabilities and Exposures (CVEs)** and **Common Platform Enumerations (CPEs)**, along with related entities such as assets and patches. This data is subject to constant updates.

3.3.1 Data Structures

In our system, the main entities and their relationships can be represented as follows:

3.3.1.1 Key Entities

- **CVEs**: Common Vulnerabilities and Exposures, which represent identified vulnerabilities.
- **CPEs**: Common Platform Enumerations, which represent specific hardware and software configurations affected by vulnerabilities.
- **Assets**: Represent devices in the network that may be affected by vulnerabilities.
- **Patches**: Represent fixes or updates for the vulnerabilities.
- **Reports**: Reports based in risk assessment results

3.3.2 Elasticsearch Indexes

In Elasticsearch, data is organized in the form of indexes. The structure might include:

- **Vulnerabilities Index**:

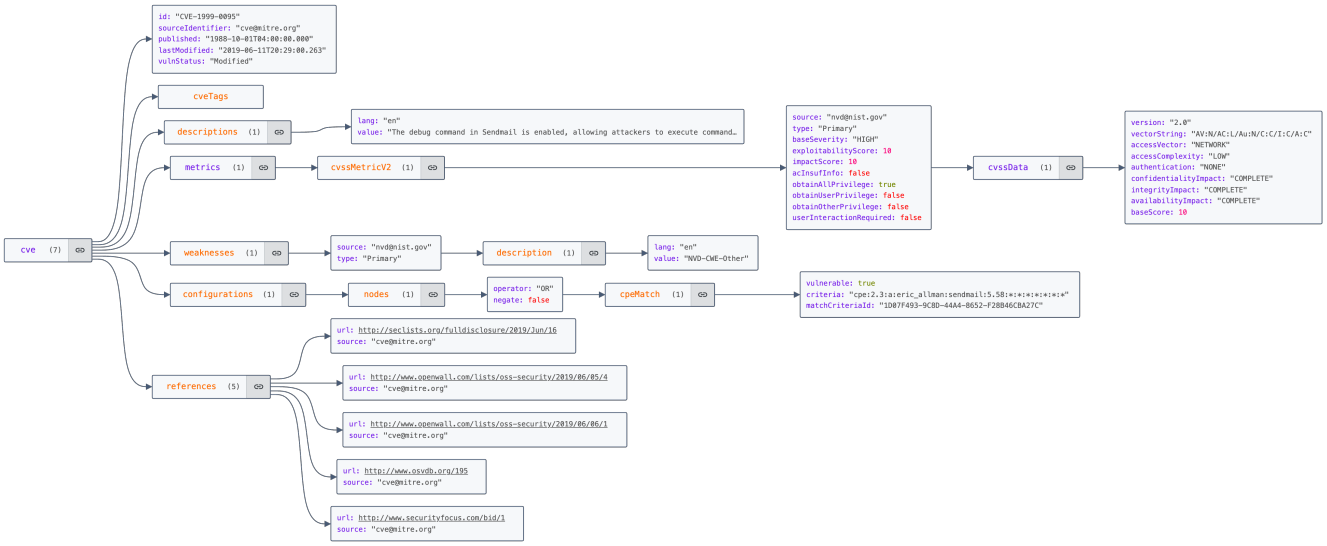

Figure 3.13: CVE JSON Schema

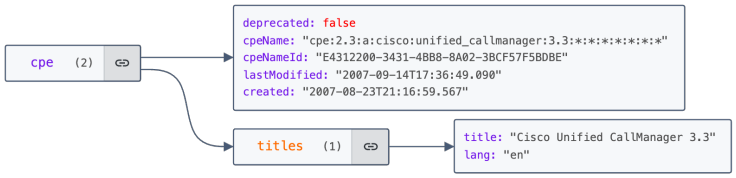

Figure 3.14: CPE JSON Schema

  - **Document Type**: `cve`
    - `cve_id` (string): Unique identifier for the CVE.
    - `description` (text): Description of the vulnerability.
    - `severity` (keyword): Severity level of the vulnerability (e.g., low, medium, high).
    - `published_date` (date): Date when the CVE was published.
    - `cpe_list` (nested): Array of CPEs associated with the CVE.

3.3.3 PostgreSQL Databases:

In our system, we utilize PostgreSQL to store data in structured tables. The structure might include:

- **Assets DB**:
  - Few of the attributes of this table are mentioned below:
    - `asset_id` (string): Unique identifier for the asset.
    - `asset_type` (keyword): Type of asset (e.g., server, workstation).
    - `status` (keyword): Status of the asset (e.g., active, inactive).
    - `cve_associated` (array): Array of CVEs associated with the asset.
- **Patches DB**:
  - Few of the attributes of this table are mentioned below:
    - `patch_id` (string): Unique identifier for the patch.
    - `cve_id` (string): CVE that the patch addresses.
    - `release_date` (date): Date when the patch was released.
- **Reports DB**:
  - Few of the attributes of this table are mentioned below:
    - `report_id` (string): Unique identifier for the report.
    - `report_date` (date): Date when the report was generated.
    - `asset_id` (string): Unique identifier of the asset related to the report.
    - `cve_id` (string): CVE identifier associated with the vulnerabilities addressed in the report.
    - `summary` (text): Brief summary of the findings in the report.
    - `recommendations` (text): Suggested actions or recommendations based on the report's findings.

Above is the information domain as per iteration 1 with current data requirements. Minor updates might be made when work starts on specific modules in the future.

Chapter 4

Implementation and Testing

The system encompasses several key functionalities and each functionality is handled by its respective micro-services , which communicate via RESTful APIs. The main functionalities upto current iteraton are listed below:

- ***Asset Discovery and Management:*** The system allows system administrators to discover all assets connected to the network, maintain an up-to-date inventory, and manage asset details effectively. This includes scanning the network for devices, identifying their types, statuses, and configurations, and making necessary updates to the asset database.
- ***Risk Assessment:*** IntraSec integrates with the Common Vulnerabilities and Exposures (CVE) and Common Platform Enumerations (CPE) databases to identify and assess vulnerabilities associated with the discovered assets. This assessment helps in recognizing potential security threats and prioritizing them for remediation.

4.1 Algorithm Design

The Risk Assessment Algorithms based on Reyes et al. [2022]

4.1.1 Risk Assesment

Main Algorithm: Risk Analysis
Input: $data \leftarrow$ List of hosts with vulnerabilities and ports
Output: Risk metrics for each host and the network
Initialize $network\_avt \leftarrow CalculateNetworkAVT(data)$
$total\_open\_ports, total\_vulnerabilities \leftarrow CalculateTotalPortsAndVulnerabilities(data)$
Print $total\_open\_ports, total\_vulnerabilities$
for each $host \in data$ **do**
    $host\_metrics \leftarrow CalculateHostRiskMetrics(host, total\_open\_ports, total\_vulnerabilities, network$
    Store $host\_metrics$
end for
Return: Network and host risk metrics

4.1.2 Calculate Total Ports and Vulnerabilities

Sub-algorithm: Calculate Total Ports and Vulnerabilities
Input: $data \leftarrow$ List of hosts
Output: $total\_open\_ports, total\_vulnerabilities$
Initialize $total\_open\_ports \leftarrow 0$, $total\_vulnerabilities \leftarrow 0$
for each $host \in data$ **do**
    $total\_open\_ports \leftarrow total\_open\_ports + |host.OpenPorts|$
    for each $software \in host.SoftwareInformation$ **do**
        $total\_vulnerabilities \leftarrow total\_vulnerabilities + |software.CVEs|$
    end for
end for
Return: $total\_open\_ports, total\_vulnerabilities$

4.1.3 Calculate Host Risk Metrics

Sub-algorithm: Calculate Host Risk Metrics
Input: $host, total\_open\_ports, total\_vulnerabilities, network\_avt$
Output: $host\_metrics$
$host\_ports \leftarrow |host.OpenPorts|$
$host\_vulns \leftarrow CalculateHostVulnerabilities(host)$
$poe \leftarrow CalculateProbabilityOfExploitation(host\_vulns, total\_vulnerabilities)$
$pop \leftarrow CalculateProbabilityOfPortExploitation(host\_ports, total\_open\_ports)$
$avg\_metrics \leftarrow CalculateAverageVulnerabilityMetrics(host)$
$po \leftarrow ((avg\_metrics.avg\_tr + avg\_metrics.avg\_ep)/2) \times ((poe + pop)/2) \times network\_avt$
$risk\_factor \leftarrow po \times avg\_metrics.avg\_impact$
Return: $host\_metrics \leftarrow \{hostname, host\_ports, host\_vulns, risk\_factor\}$

4.1.4 Calculate Host Vulnerabilities

Sub-algorithm: Calculate Host Vulnerabilities
Input: $host$
Output: $host\_vulns$
Initialize $host\_vulns \leftarrow 0$
for each $software \in host.SoftwareInformation$ **do**
    $host\_vulns \leftarrow host\_vulns + |software.CVEs|$
end for
Return: $host\_vulns$

4.1.5 Calculate Average Vulnerability Metrics

Sub-algorithm: Calculate Average Vulnerability Metrics
Input: $host$
Output: $avg\_metrics \leftarrow \{avg\_tr, avg\_ep, avg\_impact\}$
Initialize $total\_tr \leftarrow 0$, $total\_ep \leftarrow 0$, $total\_impact \leftarrow 0$
Initialize $vuln\_count \leftarrow 0$
for each $software \in host.SoftwareInformation$ **do**
    for each $cve \in software.CVEs$ **do**
        $vuln\_count \leftarrow vuln\_count + 1$
        $total\_tr \leftarrow total\_tr + CalculateTR(cve._source.references)$
        $cvss\_score \leftarrow GetCVSSScore(cve)$
        $total\_ep \leftarrow total\_ep + CalculateEP(cvss\_score)$
        $total\_impact \leftarrow total\_impact + cvss\_score/10$

```
        end for
    end for
    if vuln_count > 0 then
        avg_tr ← total_tr/vuln_count
        avg_ep ← total_ep/vuln_count
        avg_impact ← total_impact/vuln_count
    else
        avg_tr, avg_ep, avg_impact ← 0, 0, 0
    end if
    Return: avg_metrics ← {avg_tr, avg_ep, avg_impact}
```

4.2 Implementation Screenshots

4.2.1 Dashboard and Host Management

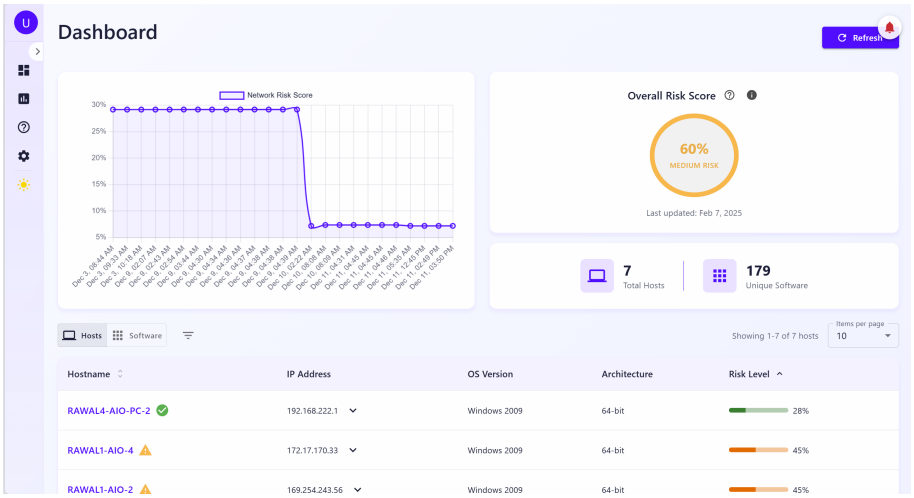

Figure 4.1: Main Dashboard Interface

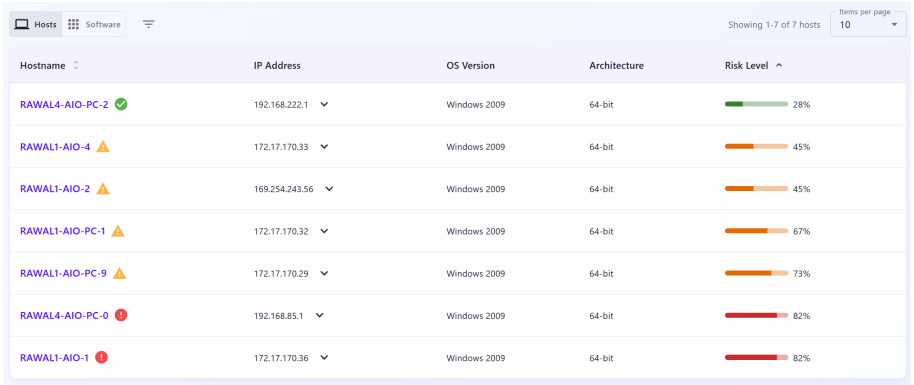

Figure 4.2: List of Connected Hosts

4.2.2 Asset Details and Analysis

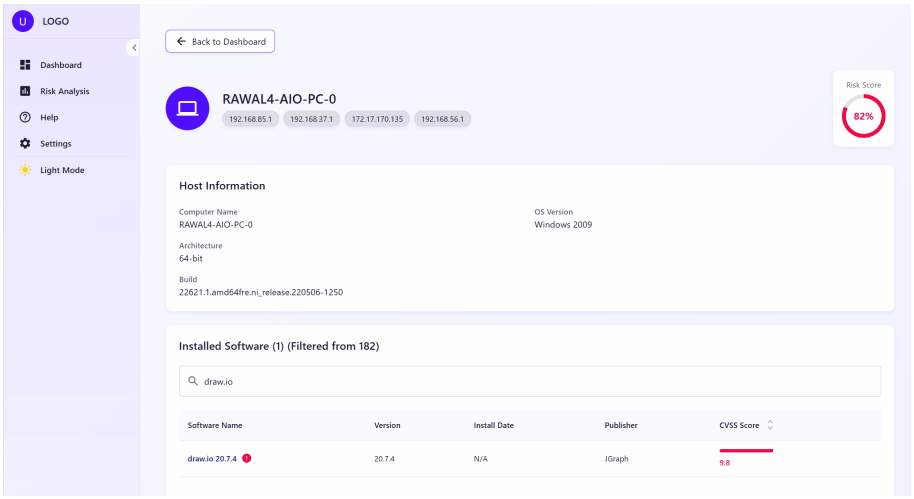

Figure 4.3: Detailed Host Information

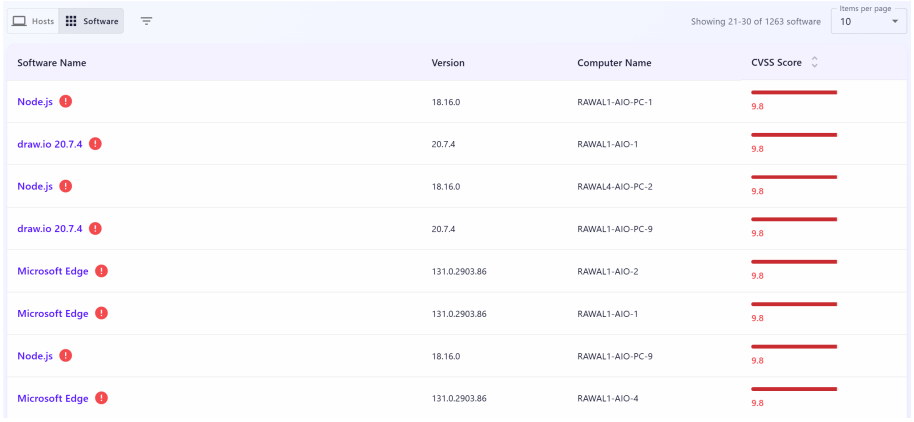

Figure 4.4: Software Inventory

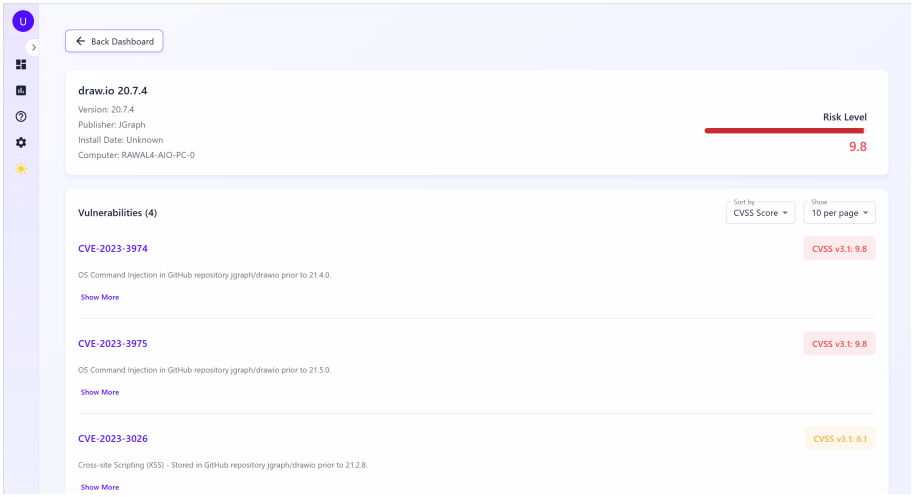

Figure 4.5: Software Details with CVEs

4.2.3 Risk Analysis and Configuration

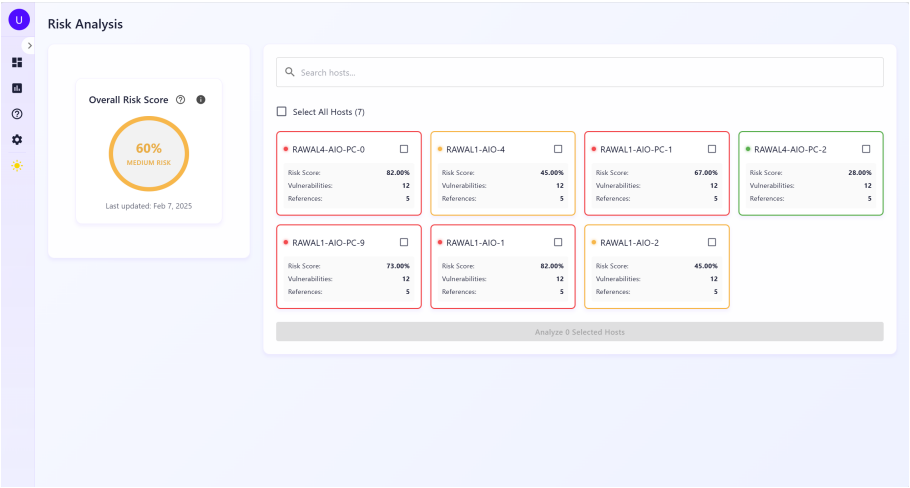

Figure 4.6: Risk Analysis Dashboard

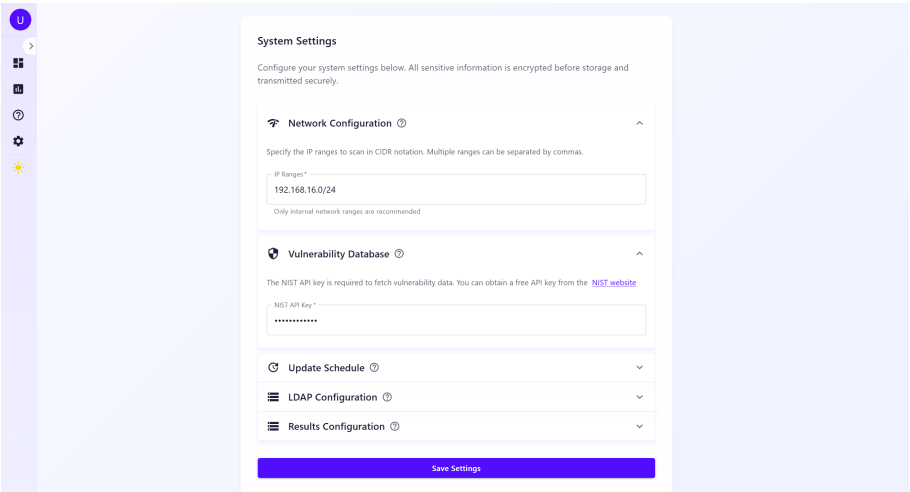

Figure 4.7: System Configuration Interface

4.2.4 Risk Visualization

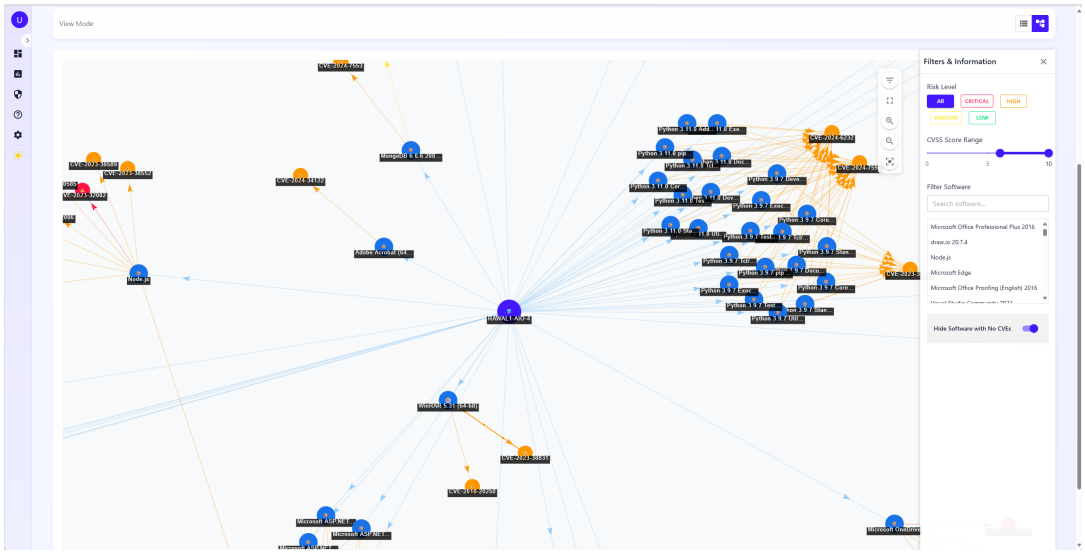

Figure 4.8: Network Risk Graph Visualization

4.2.5 CVE Prioritization

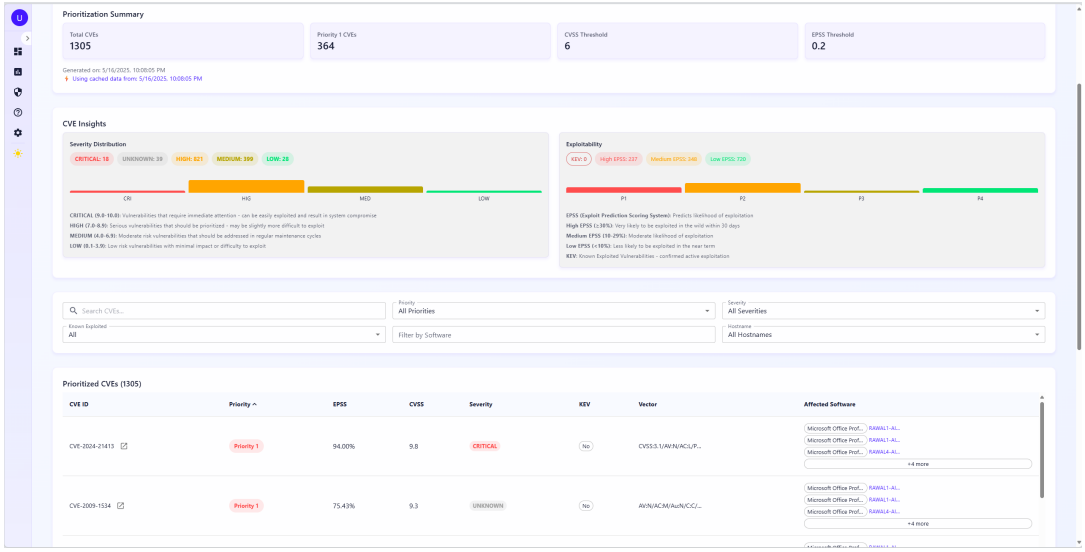

Figure 4.9: CVE Prioritization Interface

4.2.6 Risk Reports

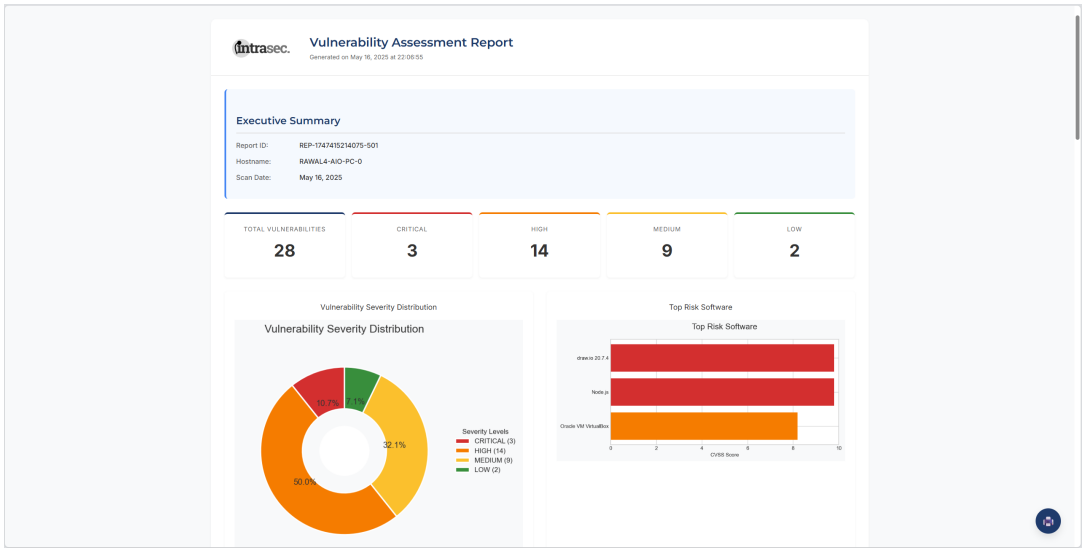

Figure 4.10: Report Summary

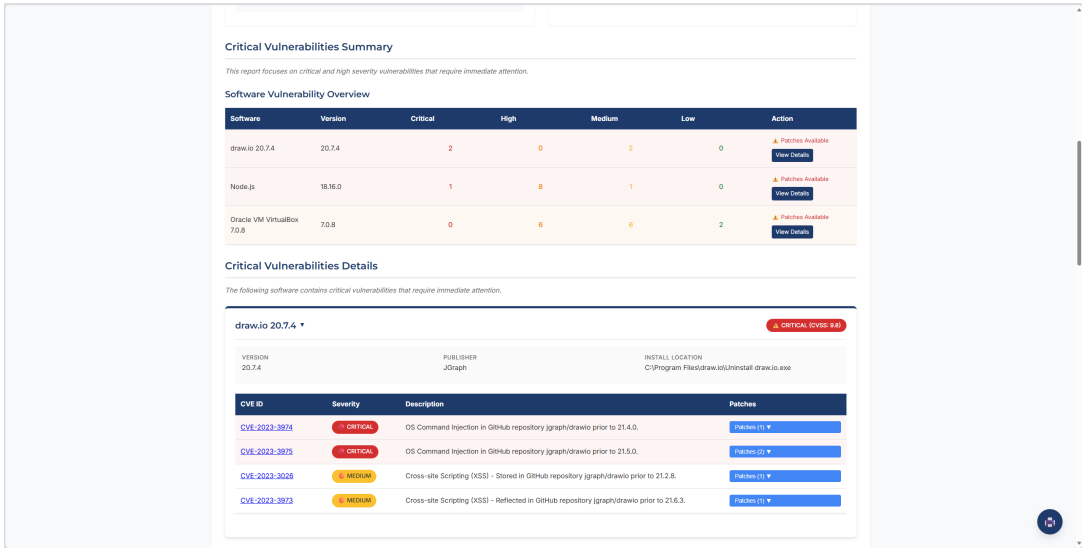

Figure 4.11: Detailed Risk Assessment Report

4.3 External APIs/SDKs

API/Service	**NIST NVD**
Description	The NVD is a U.S. government database for managing vulnerabilities, security metrics, and compliance using the Security Content Automation Protocol (SCAP). It automates processes and includes data on security checklists, software flaws, and impact metrics.
Purpose	Fetches CPEs (Common Platform Enumerations) and CVEs (Common Vulnerabilities and Exposures), which are stored locally in Elasticsearch. This data is used for efficient risk assessments by matching vulnerabilities with network assets.
Endpoint	• https://services.nvd.nist.gov/rest/json/cves/2.0 • https://services.nvd.nist.gov/rest/json/cpes/2.0
API/Service	**FIRST.org EPSS**
Description	The Exploit Prediction Scoring System (EPSS) is an open, data-driven effort for estimating the likelihood that a software vulnerability will be exploited in the wild.
Purpose	Retrieves exploit probability scores for CVEs to help prioritize vulnerabilities based on their likelihood of exploitation.
Endpoint	https://api.first.org/data/v1/epss
API/Service	**CISA KEV**
Description	The Cybersecurity and Infrastructure Security Agency (CISA) Known Exploited Vulnerabilities (KEV) Catalog provides information about vulnerabilities that are being actively exploited by threat actors.
Purpose	Retrieves information about vulnerabilities that are being actively exploited to help prioritize remediation efforts.
Endpoint	https://www.cisa.gov/sites/default/files/feeds/known_exploited_vulnera
API/Service	**VulnCheck**
Description	VulnCheck provides enhanced vulnerability intelligence and enriched CVE data through its NVD++ and KEV services.
Purpose	Provides additional context and intelligence for vulnerabilities, including information about ransomware associations.
Endpoint	• https://api.vulncheck.com/v3/index/nist-nvd2 • https://api.vulncheck.com/v3/index/vulncheck-kev

Table 4.1: Details of the APIs and Services Used for Risk Assessment

4.4 Unit Testing

4.4.1 Dashboard

Test ID	Objective	Details
TC001	Verify system info retrieval	**Precondition:** System exists in database **Steps:** Send GET to /api/getSysteminfo **Data:** ?hostname=test-host **Expected:** Status 200, system info with risk data **Reference:** app/api/getSysteminfo/route.js **Result:** Pass
TC002	Verify CVE data processing	**Precondition:** Valid software data exists **Steps:** Send POST to /api/getCVE **Data:** {softwares: [name: 'test', version: '1.0']} **Expected:** Status 200, processed CVE data **Reference:** app/api/getCVE/route.js **Result:** Pass
TC003	Verify CVSS score calculation	**Precondition:** CVE metrics available **Steps:** Call getCvssScore function **Data:** Metrics object with different CVSS versions **Expected:** Correct CVSS score based on priority **Reference:** app/softwareinfo/[hostname]/[name]/[version] **Result:** Pass
TC004	Verify software CVE mapping	**Precondition:** Software and CVE data available **Steps:** Execute createSoftwareCveMap **Data:** Software list with CVE data **Expected:** Correct mapping of software to CVEs **Reference:** app/api/riskanalysis/route.js **Result:** Pass
TC005	Verify software sorting	**Precondition:** Software list loaded **Steps:** Execute getSortedData function **Data:** Software array with CVSS scores **Expected:** Correctly sorted software list **Reference:** components/ComputerTable.js **Result:** Pass

TC006	Verify CVE version detection	**Precondition:** CVE metrics available **Steps:** Call getCvssVersion function **Data:** Metrics object **Expected:** Correct CVSS version string **Reference:** app/softwareinfo/[hostname]/[name]/[version] **Result:** Pass
TC007	Verify software enrichment	**Precondition:** Raw software data exists **Steps:** Process enrichedSoftware mapping **Data:** Software and CVE data **Expected:** Enriched software with CVE scores **Reference:** app/api/getSoftware/route.js **Result:** Pass
TC008	Verify risk color calculation	**Precondition:** CVSS score available **Steps:** Call getRiskColor function **Data:** CVSS score value **Expected:** Correct risk color code **Reference:** app/softwareinfo/[hostname]/[name]/[version] **Result:** Pass
TC009	Verify data filtering	**Precondition:** Data array available **Steps:** Execute filterData function **Data:** Array of items to filter **Expected:** Filtered data array **Reference:** components/ComputerTable.js **Result:** Pass
TC010	Verify sort direction toggle	**Precondition:** Sort config exists **Steps:** Call handleSort function **Data:** Sort key **Expected:** Updated sort configuration **Reference:** components/ComputerTable.js **Result:** Pass
TC011	Verify CVE details expansion	**Precondition:** CVE data loaded **Steps:** Call toggleCveDetails function **Data:** CVE ID **Expected:** Toggled expansion state **Reference:** app/softwareinfo/[hostname]/[name]/[version] **Result:** Pass

TC012	Verify results update with CVEs	**Precondition:** Results and software data available **Steps:** Call updateResultsWithCves **Data:** Results and software objects **Expected:** Updated results with CVE data **Reference:** app/api/riskanalysis/route.js **Result:** Pass

Table 4.2: Unit test cases

4.4.2 Assets Management

Test ID	Objective	Details
TC001	Verify LDAP Authentication	**Precondition:** Valid LDAP server configured **Steps:** Call RefreshAssets endpoint with credentials **Data:** Basic Auth header with valid credentials **Expected:** Status 200, successful authentication **Reference:** handlers/ldap.go **Result:** Pass
TC002	Verify Computer Discovery	**Precondition:** Authenticated LDAP connection **Steps:** Execute LDAPSearchComputers function **Data:** Valid baseDN and search filter **Expected:** List of computer entries **Reference:** ldaputils/ldaputils.go **Result:** Pass
TC003	Verify PowerShell Data Collection	**Precondition:** Valid computer connection **Steps:** Execute EnterPowerShellSession function **Data:** Computer name and credentials **Expected:** System information JSON output **Reference:** ldaputils/ldaputils.go **Result:** Pass
TC004	Verify Results Storage	**Precondition:** Valid PowerShell output **Steps:** Call StorePowerShellOutput function **Data:** JSON formatted computer data **Expected:** Data successfully stored in results.json **Reference:** ldaputils/ldaputils.go **Result:** Pass
TC005	Verify Active Hosts Count	**Precondition:** Results file with computer data **Steps:** Call GetHosts endpoint **Data:** Existing results.json file **Expected:** Correct count of active hosts **Reference:** handlers/hosts.go **Result:** Pass

TC006	Verify Software Inventory	**Precondition:** Results file with software data **Steps:** Call GetAllSoftware endpoint **Data:** Optional hostname parameter **Expected:** List of unique software entries **Reference:** handlers/software.go **Result:** Pass
TC007	Verify Refresh Status	**Precondition:** Ongoing or completed refresh operation **Steps:** Call GetRefreshStatus endpoint **Data:** None **Expected:** Current status with progress percentage **Reference:** handlers/ldap.go **Result:** Pass
TC008	Verify Risk Information Storage	**Precondition:** Valid risk data available **Steps:** Call StoreRisk endpoint **Data:** Risk information JSON **Expected:** Updated results with risk data **Reference:** handlers/results.go **Result:** Pass

Table 4.3: Unit test cases for LDAP Asset Discovery Tool

4.4.3 Risk Analysis API

Test ID	Objective	Details
TC001	Verify Risk Calculator Initialization	**Precondition:** Valid JSON input data **Steps:** Initialize RiskCalculator class **Data:** Host data with 2 systems, each having 3 vulnerabilities and 2 open ports **Expected:** Correct calculation of network totals (6 vulnerabilities, 4 ports) **Reference:** calculator/risk_calculator.py **Result:** Pass
TC002	Verify CVSS Score Processing	**Precondition:** Host with multiple CVEs **Steps:** Process vulnerability metrics **Data:** CVE entries with V3.1 (score 8.5), V3.0 (score 7.2), V2.0 (score 6.8) **Expected:** Highest V3.1 score (8.5) selected for impact calculation **Reference:** calculator/risk_calculator.py **Result:** Pass
TC003	Verify Risk Level Classification	**Precondition:** Calculated risk factors **Steps:** Convert risk percentages to levels **Data:** Risk values: 85% (HIGH), 65% (MEDIUM-HIGH), 45% (MEDIUM), 25% (MEDIUM-LOW), 15% (LOW) **Expected:** Correct risk level mapping for all test values **Reference:** utils/risk_levels.py **Result:** Pass
TC004	Verify Network AVT Calculation	**Precondition:** Multiple CVEs with different ages **Steps:** Calculate network-wide AVT **Data:** CVEs from 2020 (1095 days), 2021 (730 days), 2022 (365 days) **Expected:** $AVT = 730$ days $\rightarrow$ Normalized value $= 1.0$ **Reference:** calculator/metrics_calculator.py **Result:** Pass

TC005	Verify Probability of Occurrence	**Precondition:** Host with known metrics **Steps:** Calculate PO using formula **Data:** $TR = 0.8$, $EP = 1.0$, $POE = 0.6$, $POP = 0.4$, $AVT = 0.5$ **Expected:** $PO = ((0.8 + 1.0)/2) \times ((0.6 + 0.4)/2) \times 0.5 = 0.225$ **Reference:** calculator/risk_calculator.py **Result:** Pass
TC006	Verify JSON Data Loading	**Precondition:** Various JSON input files **Steps:** Load and validate data structure **Data:** Valid JSON, malformed JSON, empty file, invalid structure **Expected:** Proper error handling for all invalid cases **Reference:** utils/data_loader.py **Result:** Pass
TC007	Verify API Risk Calculation	**Precondition:** Running FastAPI server **Steps:** POST request to /calculate-risk/ endpoint **Data:** Complete host data with 5 systems, each having varied vulnerabilities **Expected:** Valid JSON response with host and network metrics **Reference:** api/main.py **Result:** Pass
TC008	Verify Network Average Calculation	**Precondition:** Multiple host risk assessments **Steps:** Calculate network-wide statistics **Data:** 3 hosts with risk factors $(0.8, 0.6, 0.4)$, vulnerabilities $(10, 15, 5)$ **Expected:** Average risk $= 0.6$, Total vulnerabilities $= 30$ **Reference:** calculator/risk_calculator.py **Result:** Pass

Table 4.4: Unit test cases for IntraSec Risk Analysis API

4.4.4 Vulnerability Database

Test ID	Objective	Details
TC001	CPE Search Functionality	**Precondition:** Elasticsearch with indexed CPE data **Steps:** Execute searchcpe() function **Data:** Software: "Postman x86" v11.20.0 **Expected:** Matching CPE entries with version validation **Reference:** api/logic.py **Result:** Pass
TC002	Version Range Validation	**Precondition:** Version strings in database **Steps:** Test version range matching **Data:** Version: "2.47.0", Pattern: "2." **Expected:** Correct version pattern matching **Reference:** api/logic.py **Result:** Pass
TC003	CVE Data Ingestion	**Precondition:** Valid CVE JSON files **Steps:** Run bulk ingestion process **Data:** Multiple CVE entries **Expected:** Successful ES ingestion **Reference:** ingester/cve.py **Result:** Pass
TC004	CPE String Parser	**Precondition:** CPE 2.3 strings **Steps:** Parse CPE components **Data:** Standard CPE 2.3 format string **Expected:** Valid component dictionary **Reference:** api/logic.py **Result:** Pass
TC005	ES Connection	**Precondition:** ES instance running **Steps:** Test connection and status **Data:** Dev/Prod configurations **Expected:** Connection success **Reference:** ingester/connection.py **Result:** Pass

TC006	CVE Filter Logic	**Precondition:** CVE dataset **Steps:** Apply version filters **Data:** Version-specific CVEs **Expected:** Correctly filtered results **Reference:** api/logic.py **Result:** Pass
TC007	Bulk Processing	**Precondition:** Data files ready **Steps:** Process multiple files **Data:** JSON data folder **Expected:** Complete processing **Reference:** ingester **Result:** Pass
TC008	API Response	**Precondition:** Valid request body **Steps:** Test /cves endpoint **Data:** Software list request **Expected:** Valid JSON response **Reference:** main api **Result:** Pass

Table 4.5: Unit test cases for IntraSec-VulnDB

Chapter 5

Conclusions and Future Work

5.1 Conclusion

IntraSec provides an effective client-less solution for risk assessment and asset management, enabling real-time discovery and prioritization of vulnerabilities in internal networks. By integrating with APIs such as NIST NVD, FIRST.org EPSS, CISA KEV, VulnCheck, and Elasticsearch, the system supports System Administrators in identifying and prioritizing Common Vulnerabilities and Exposures (CVEs) and notifying host owners via email with detailed vulnerability reports for remediation. Its microservices-based architecture ensures scalability and efficient management of assets and risks, addressing the challenges of complex network environments in large organizations.

5.2 Future Work

Future enhancements to IntraSec could include support for Linux and HarmonyOS to broaden device compatibility and asset discovery for IoT devices to address their growing presence in enterprise networks. Integrating machine learning for predictive CVE prioritization would further enhance threat intelligence.

Bibliography

Jorge Reyes, Walter Fuertes, Paco Arévalo, and Mayra Macas. An environment-specific prioritization model for information-security vulnerabilities based on risk factor analysis. *Electronics*, 11(9):1334, 2022. doi: 10.3390/electronics11091334. URL https://www.mdpi.com/journal/electronics.