



Module Code & Module Title

CS6PO5NT-Final Year Project

Assessment Weightage & Type

Final Year Project Interim Report (25%)

Year and Semester

2023-24 Spring

Student Name: Aayush Wanem Limbu

London Met ID: 22072043

College ID: np05cp4a220010

Assignment Submission Date: 8th January 2025

Internal Supervisor: Amit Shrestha

External Supervisor: Prakash Basnet

Title: VISTA

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

22072043 Aayush Wanem Limbu

 Islington College, Nepal

Document Details

Submission ID

trnoid::3618:78096932

Submission Date

Jan 8, 2025, 11:24 AM GMT+5:45

Download Date

Jan 8, 2025, 11:25 AM GMT+5:45

File Name

22072043 Aayush Wanem Limbu

File Size

67.2 KB

81 Pages





9,929 Words

57,917 Characters

12% Overall Similarity

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

Match Groups

-  **102 Not Cited or Quoted 12%**
Matches with neither in-text citation nor quotation marks
-  **6 Missing Quotations 1%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 5%  Internet sources
- 1%  Publications
- 11%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

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

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

Match Groups

- 102 Not Cited or Quoted 12%**
Matches with neither in-text citation nor quotation marks
- 6 Missing Quotations 1%**
Matches that are still very similar to source material
- 0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 5% **Internet sources**
- 1% **Publications**
- 11% **Submitted works (Student Papers)**

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	
www.snort.org		1%
2	Internet	
www.coursehero.com		1%
3	Submitted works	
British University in Egypt on 2016-05-16		1%
4	Submitted works	
West Herts College on 2023-03-10		0%
5	Submitted works	
Universiti Teknologi Malaysia on 2022-04-18		0%
6	Submitted works	
University of Wolverhampton on 2021-05-06		0%
7	Submitted works	
University of East London on 2024-09-07		0%
8	Submitted works	
Requis University on 2023-11-14		0%
9	Submitted works	
aou on 2025-01-02		0%
10	Submitted works	
University of Greenwich on 2023-11-29		0%

11	Submitted works	University of East London on 2025-01-07	0%
12	Submitted works	University of Greenwich on 2012-05-22	0%
13	Submitted works	Universiti Teknologi Malaysia on 2024-06-26	0%
14	Submitted works	University of Greenwich on 2024-05-08	0%
15	Submitted works	Coventry University on 2024-08-09	0%
16	Submitted works	University of Northampton on 2023-09-22	0%
17	Submitted works	University of Wolverhampton on 2021-05-07	0%
18	Submitted works	PSB Academy (ACP eSolutions) on 2018-10-29	0%
19	Submitted works	University of Tampa on 2024-04-05	0%
20	Submitted works	University of Greenwich on 2024-11-29	0%
21	Submitted works	Queensland University of Technology on 2024-10-07	0%
22	Submitted works	University of Sunderland on 2015-04-09	0%
23	Internet	bushra44.wordpress.com	0%
24	Submitted works	Middlesex University on 2023-11-12	0%

Table of Contents

1. Introduction	1
1.1. Project Description	1
1.2. Current Scenario	1
1.3. Problem Domain and the Project Solution	2
1.4. Aims and Objectives	4
1.4.1. Aims.....	4
1.4.2. Objectives.....	6
2. Background	8
2.1. About Client.....	8
2.2. End Users	8
2.3. Project elaboration	9
2.4. Similar Projects	10
2.4.1. Snort.....	10
2.4.2. Suricata	12
2.4.3. FortiGate Firewall	14
2.5. Comparisons	16
3. Development	16
3.1. Considered Methodology	16
3.1.1. Scrum	17
3.1.2. Waterfall model.....	19
3.1.3. Prototype Methodology.....	22
3.2. Selected Methodology.....	24
3.2.1. RUP	24
3.3. Methodology Comparison	28

4.	Progress.....	29
4.1.	Progress Table.....	29
4.2.	Development.....	32
4.2.1.	UI/UX pages	33
4.2.2.	Backend Testing.....	36
4.3.	Gantt Chart.....	39
4.4.	Work Breakdown Structure	39
5.	Further Work	40
6.	References.....	43
7.	Appendix	45
7.1.	SRS Document	45
7.1.1.	Purpose	46
7.1.2.	Scope	46
7.2.1.	Overall Description	46
7.2.2.	Product Perspective	46
7.2.3.	Product Functions	47
7.2.4.	User Classes and Characteristics	47
7.2.5.	Operating Environment	47
7.2.6.	Assumptions and Dependencies.....	48
7.3.	Specific Requirements	48
7.3.1	Functional Requirements	48
7.3.2	Real-Time Packet Capture	48
7.3.3.	Alert Generation	48
7.3.4.	Rule Management.....	48
7.3.5.	Report Generation.....	48

7.3.6. PCAP File Analysis	49
7.4.1. Non-Functional Requirements.....	49
7.4.2. Performance.....	49
7.4.3. Scalability	49
7.4.4. Usability.....	49
7.4.5. Security	49
7.5.6. Maintainability	49
7.2. Data Flow Diagram	49
7.3. Use Case	51
7.3.1. Use Case Diagram of System.....	51
7.3.2. High Level Use Case Description	51
7.4. Collaboration Diagram	54
7.5. Sequence Diagram	57
7.6. Final ERD.....	61
7.6.1. Data Dictionary	62
7.7. Survey Result.....	65
7.7.1 Post Survey Key Findings	65
7.7.2. Visual Representation of Survey Result.....	70

Table of Figures

Figure 1 Snort Logo (www.snort.org, 2025)	10
Figure 2 Snort Command Line	11
Figure 3 Suricata Logo (suricata.io, 2025)	13
Figure 4 FortiGate Dashboard (fortinet.com, 2025).....	15
Figure 5 Scrum Methodology (scrum.org, 2025)	17
Figure 6 Waterfall model (lucidchart.com, 2025).....	20
Figure 7 Prototype Model (BoardInfinity.com, 2025)	23
Figure 8 Rational Unified Process (Graphics, 2022)	24
Figure 9 Login UI/UX.....	33
Figure 10 Home Dashboard	33
Figure 11 Wireframe of Alerts Page	34
Figure 12 Wireframe of Upload PCAP page.....	34
Figure 13 PCAP Summary Report page	35
Figure 14 SRC/Destination Address Plotting.....	36
Figure 15 User Model Django	36
Figure 16 Login View Django	37
Figure 17 Logout View Django	37
Figure 18 Admin's Operator Registration View.....	38
Figure 19 Registering Custom User model to admin site	38
Figure 20 Updated Project Gantt Chart	39
Figure 21 Project WBS.....	39
Figure 22 Level 0 DFD	50
Figure 23 Use Case Diagram.....	51
Figure 24 Collaboration Diagram od Register User.....	54
Figure 25 Collaboration Diagram of Login User	55
Figure 26 Collaboration Diagram of Realtime Packet Analysis	56
Figure 27 Collaboration Diagram of PCAP Analyzer	56
Figure 28 Sequence Diagram of User Registration.....	57
Figure 29 Sequence Diagram of User Login	58
Figure 30 Sequence Diagram of Operator CRUD operation	59

Figure 31 Sequence Diagram of Discover Devices	60
Figure 32 Sequence Diagram of Network Traffic analysis.....	60
Figure 33 Sequence Diagram of PCAP Reader	61
Figure 34 Final ERD.....	61
Figure 35 Survey Question 1	70
Figure 36 Survey Question 2.....	70
Figure 37 Survey Question 3.....	71
Figure 38 Survey Question 4.....	71
Figure 39 Survey Question 5.....	72
Figure 40 Survey Question 6.....	72
Figure 41 Survey Question 7	73
Figure 42 Survey Question 8.....	73
Figure 43 Survey Question 9.....	74
Figure 44 Survey Question 10.....	74
Figure 45 Survey Question 11.....	75
Figure 46 Survey Question 12.....	75
Figure 47 Survey Question 13.....	76
Figure 48 Survey Question 14.....	76
Figure 49 Survey Question 15.....	77
Figure 50 Survey Question 16.....	77

Table of Tables

Table 1 Reasons for Scrum methodology	19
Table 2 Reasons for not using Waterfall model.....	21
Table 3 Reasons for not using Prototype model	23
Table 4 Reasons for Choosing RUP	28
Table 5 Methodology Comparison	28
Table 6 Progress table	29
Table 7 Data Dictionary f User Table	62
Table 8 Data Dictionary of Rule Table.....	62
Table 9 Data Dictionary of AlertsTriggered Table	63
Table 10 Data Dictionary of HostDeviceInfo table	64
Table 11Data Dictionary of RunningServices table	65
Table 12 Data Dictionary of PCAPHistory Table	65

1. Introduction

In today's rapidly evolving digital landscape, where data breaches and cyberattacks have become all too common, robust cybersecurity measures are of great importance. In the digital age, cyber threats are becoming more sophisticated and frequent, necessitating robust security solutions. Cyber criminals or hackers' targets and attacks organizations either small or big, Network Intrusion Detection Systems (NIDS) are crucial for safeguarding networks by monitoring traffic for malicious activity. They detect and alert network administrators of potential or ongoing attacks, such as DoS, port scanning, virus infections, and unauthorized access attempts. NIDS is essential for a comprehensive network security strategy, helping to identify and respond to threats before they cause significant damage or compromise sensitive data. This project aims to design and implement a rule-based NIDS, similar to Snort, to provide a customizable and efficient detection mechanism for known threats.

1.1. Project Description

The project is named VISTA (Visual Intrusion Surveillance and Threat Analysis), VISTA is a web-based Network Intrusion Detection System (NIDS) with a rule set to detect and identify cyber-attacks and to alert the network administrators, IT managers, system administrators and SOC analysts. It enables them to identify and get notified about threats after a cyber-attack has been detected. This project will be written in python and the library that will be used is Scapy. This project aims to solve the major issue in the field of information security.

1.2. Current Scenario

Today networks are a target by malicious actors who seeking to exploit vulnerabilities to assist unauthorized access, data theft, or disruption of services. With it being business and organization dependent on their network infrastructure they wholly rely on a row of strong security system to monitor and protect the network traffic. Sadly, however, traditional security mechanisms are struggling to keep up with a constantly evolving threat landscape, leaving holes in defense and missing detection of possible attacks. The detection of known threats is one of the most critical challenges: predefined signature-

based methods are used. Snort, for example, are these systems that use a set of rules to identify specific attack patterns. Signature based systems are good at fighting well known threats, but come with a few limitations. They do not often find new, unchanneled threats (zero day attacks) because you are depending on some knowledge of the attack pattern. Additionally, these systems can be hit hard in high traffic environments causing these systems to have performance bottlenecks and take longer to detect. In addition, the rules need to be always managed and updated, which is an ongoing challenge because we need to always maintain the system in effect. Customization of these rule based systems is also difficult to provide organization with sufficient tailoring of detection to meet their specific network environments. The additional shackle of unavailable real-time and friendly interfaces for network monitoring turns out to be worse than the first impediment, as administrators will have no ways of reacting fast to unfolding threats. In order to address these challenges, this project proposes the development of a Network Intrusion Detection system (NIDS) which integrates rule based detection of intrusion with an easy to use web dashboard for real time monitoring of the network. It will be designed such that the network traffic will be handled efficiently, scalable, customizable, and possess improved detection capabilities in terms of known threats. This will be a powerful NIDS allowing the administrators to identify and mitigate potential security risks to improve the entire network security.

1.3. Problem Domain and the Project Solution

This work intends to build a Snort compatible, rule-based Network Intrusion Detection System (NIDS) accompanied with a web based admin dashboard. This solution aims to streamline network security administration, offer meaningful pcap file analysis with automated reporting, and improve the network security posture for the organizations.

Key Components and Features

1. Snort-Compatible NIDS:

Signature-Based Detection: It also takes advantage of the well-established Snort rule set to detect known attack patterns such as DoS attacks, malware infections, and unauthorized access.

Customizable Rules: It provides organizations with flexibility to tailor the NIDS to their own needs, by changing or adding Snort rules.

Effective Threat Response: It ensures a wide range of reliable protection against known threats by making use of predefined attack signatures.

2. Web-Based Admin Dashboard:

Intuitive Interface: The dashboard is designed with real-time monitoring, alert management and incident disposal in mind where both technical and non technical staff can easily use.

Real-Time Alerts and Reports: It offers a quick and informed picture of the network traffic, alerts and ongoing threats.

Simplified Management: It reduces the need to rely on complex command line configurations making the system easier to manage for small to medium security teams alike.

3. PCAP File Analysis and Report Generation:

Automated Analysis: Enables us to perform a detailed examination of network traffic captures (pcap files) to find potential threats, or anomalies.

Comprehensive Reporting: Provides actionable data for threat mitigation and compliance work to security teams, which generates insightful reports.

4. Scalability and Performance:

Handling Large Volumes: Intended to maintain high performance in the presence of large networks traffic.

Critical Alert Prioritization: Filters for fighting false positives and placing priority on critical alerts to keep alert fatigue and improve response diligence.

Scalable Architecture: Grows with the network infrastructures, making sure the system remains effective as such organizational needs grow.

Benefits

Enhanced Threat Detection: The system combines Snort's robust rule based detection with real time monitoring to guarantee accurate and timely threat identification.

Improved Usability: It provides a wide accessibility and manageability across technical and not technical users, in bridging the gap between the two.

Increased Efficiency: Analysis and reporting is automated, removing the burden of analysis and reporting from the security operations team reducing response time to threats and promoting more effective threat response.

Future-Proof Security: The system has been designed to scale; as network demands grow, so does the system's relevance and its effectiveness.

Conclusion

Implementation of this project provides for a thorough, scalable, and accessible solution for network intrusion detection. However, by combining Snort's powerful detection capabilities and a user friendly web interface it essentially solves the problem of securing modern networks in an evolving cyber threat landscape. In addition to improving threat detection with higher accuracy and speed, the system also enhances the overall performance of network security management, thereby improving how organizations can more effectively protect their critical assets.

1.4. Aims and Objectives

1.4.1. Aims

Below are the major aims that this final year project is going to accomplish.

- **Develop a Rule-Based Intrusion Detection System (NIDS):**

One of the major features in this project is to design and implement a Network Intrusion Detection System that uses a rule-based detection module which is similar to Snort, this rule-based detection will be used to identify and log to known network threats such as Denial of Service (DoS), port scanning, malware infections, and unauthorized access attempts.

- **Real-Time Network Traffic Monitoring:**

This feature will enable the system to be able to monitor real-time network traffic, this feature will use python's external networking libraries like Scapy and will be used to implement efficient packet capture techniques to analyze and detect any potential threats that can occur on the network.

- **Create a Customizable Rule Engine:**

One of the major aims is to develop and make a flexible and scalable rule management system that will allow network administrators to customize and update detection rules based on their specific network environments and security requirements, I also aim to make this rule module compatible with the snort's rule sets.

- **Enable PCAP File Analysis:**

Another aim for this project is implement a functionality for uploading and analyzing PCAP files which allows for historical traffic analysis, payload decryption, source and destination IP address viewing, and a detailed pcap analysis report generation which allows administrators to investigate past network activity for potential threats or for any security incidents.

- **Alerting and Reporting System:**

Another important feature of this project is to create an alerting and reporting system that categorizes detected threats by severity and provides a clear information to network administrators which can result in ensuring a timely response to any incidents that may arise.

- **Visualization of Network Devices:**

Another important feature for this project is to implement a visualization on the dashboard that can represent network topology and can also display the devices that are present and active on the network, this feature helps to highlighting any unusual or unauthorized devices or behavior on our network environment.

- **Scalability and Performance Optimization:**

One of the important tasks for this final year project is to optimize the performance of the system and ensure that it can handle high traffic volumes without any performance degradation.

- **Contribute to Enhanced Network Security:**

This project aims to provide a comprehensive, reliable and a scalable solution for enhancing network security by helping organizations to detect and mitigate any cyber threats before they can cause significant damage.

- **Build a User-Friendly Web Dashboard:**

For this project I have to design and develop an intuitive and interactive web-based dashboard using Django which will provide a real-time statistics, alerts and detailed insights into the captured network traffic which can help administrators to make informed decisions quickly.

1.4.2. Objectives

- **Implement Real-Time Packet Capture and Analysis:**

One of the major objectives is to develop a functionality that can continuously capture and analyze the packets in real-time using tools like Scapy. It will enable us to detect suspicious activity and intrusions as they happen on the network.

- **Design and Integrate Rule-Based Detection Mechanism:**

Another important objective for this project is to create a rule-based detection engine that matches the network traffic payloads or signatures against a predefined attack signatures which enables the system to identify known threats such as DoS attacks, malware, port scanning and unauthorized access attempts.

- **Provide Customizable Detection Rules:**

Another one is to build a system that allows administrators to easily customize, add or remove detection rules based on specific organizational needs or emerging threats, ensuring flexibility and adaptability in the system.

- **Develop a Web-Based Dashboard for Monitoring and Reporting:**

The outcome is to design and develop a web-based user interface using Django that allows network administrators to view real-time traffic statistics, alerts and network activity by facilitating better decision-making and timely responses to detected threats.

- **Implement PCAP File Upload and Analysis:**

It enables the system to process and analyze uploaded PCAP files, by providing a comprehensive historical view of network traffic to identify previously undetected threats or to analyze past security incidents.

- **Design an Alerting System with Severity Categorization:**

To create an alerting system that categorizes detected threats by their severity (e.g., critical, high, medium, low) and notifies administrators, accordingly which ensures that urgent issues are prioritized.

- **Visualize Network Topology and Device Activity:**

Another important objective is to make a module that can be used to visualizations all the available devices on the network on the web-dashboard which represent network topology and showing the relationships between devices and identifying any unusual or unauthorized devices or connections.

- **Ensure System Scalability and Performance Optimization:**

To optimize the system's performance to handle high volumes of network traffic without significant degradation, and to consider scalable architecture options for future expansion to larger network environments.

- **Test and Validate System Effectiveness:**

To test the system with various types of network traffic and security scenarios, which ensures its ability to accurately detect threats and provide meaningful insights into network behavior.

- **Provide Export and Reporting Features:**

To implement the ability to export analysis reports, alerts, and traffic data into multiple formats (e.g., CSV, PDF) enabling easy sharing and further analysis of the findings.

- **Enhance Network Security Posture:**

It can be used to ensure that the system contributes effectively to an organization's overall cybersecurity strategy by proactively identifying threats and enabling quick mitigation, thus enhancing network security and reducing the risk of data breaches or service disruption.

2. Background

2.1. About Client

The client for this project can be anyone either they are a individual or a large scale organizations that are looking forward to improve their informational security. They are committed to enhancing their cybersecurity infrastructure to protect sensitive information, prevent unauthorized access, and respond swiftly to potential threats. The client has identified the need for a comprehensive threat detection and analysis system to safeguard their network from evolving cyber threats.

2.2. End Users

Administrators (Admins): They manage the system, configure detection rules, analyze detected threats, generate reports, and optimize system performance. Admins have full access to all the functionalities of the system.

Network operators have the principal responsibility of monitoring network traffic in real-time. They ensure that alerts are meaningful and get to the right people, downloading reports and logs when necessary to keep the network safe. They assist in "threat management," which is just a fancy way of saying they help keep the place from getting hacked. Unlike a sysadmin, an operator has limited access and is focused on "eyes on glass" (aka "real-time")

IT Security Team: A more extensive team that unites with Admins and Operators to use the system for the betterment of overall security and for coordinating responses to incidents.

2.3. Project elaboration

The project plans to create and implement a strong system for detecting threats and analyzing them, called VISTA. This system will allow the client to manage the security of their network in a way that is much more proactive than what was done previously. The project has several key features:

Rule Management for Threat Detection: Enables Admins to create, update, and oversee rules that articulate the patterns and Admin wants to use for detecting threats. Enabling administrators to upload and analyze network traffic data contained in PCAP files to spot possible security breaches.

Real-Time Observation: Offering instantaneous overviews of network movement and producing immediate alerts for any detected dangers. By providing comprehensive reporting capabilities, where users can produce, download, and export detailed documents about the threats that have been detected, the performance of the system, and any security incidents that have occurred.

Analysis of Threats: Providing users the capability to classify and analyze detected threats in a manner that permits working priorities to emerge, based on the severity of the threats.

Management of users: An administration function that allows management of both the addition and removal of operators. Also allows for the management of access permissions to ensure the secure operation of the system.

System Optimization: Concentrating on ongoing enhancements to performance, ensuring that the system manages the large volumes of data with efficiency.

VISTA is designed to integrate into the client's existing infrastructure, augmenting their cybersecurity with advanced detection capabilities, simple user management, and robust reporting tools.

2.4. Similar Projects

2.4.1. Snort

Snort is an opensource and free Network Intrusion Detection System (NIDS) and Network Intrusion Preventions System (NIPS) that monitors traffic and finds alerts if any malicious activities or any signs of intrusion is found. Snort provides a real-time traffic analysis and monitoring features as well as data packet logging meaning it can save all the network packets in a separate pcap file or any database that network admins can configure. Snort is a rule-based IDS/IPS solution that my organizations uses, which its own rule language to perform anomaly, signature and protocol-based inspection modules to find and detect any potential malicious activity in a network environment. Not only that snort is also recognized as an industry standard in detecting, alerting, logging and dropping packets, malicious payloads or any other cyber-attacks not only because of its efficient packet inspection but also because of the larger community of cyber security specialist and enthusiasts which maintains and updates snort rules for ever new cyber-attacks that is reported.

The GitHub repository link for snort is: <https://github.com/snort3/snort3.git>



Figure 1 Snort Logo (www.snort.org, 2025)

The features of Snort are:

- Real time traffic monitoring
- Packet logging
- Analysis of protocol
- Content matching

- OS fingerprinting
- Vast community of rules
- Packet Sniffer
- Custom rule writing and development

```

ubuntu@ubuntu-VirtualBox:~$ sudo ip link set enp0s3 promisc on
[sudo] password for ubuntu:
ubuntu@ubuntu-VirtualBox:~$ sudo snort -q -l /var/log/snort -i enp0s3 -A console -c /etc/snort/snort-copy.conf
04/07-23:42:33.268243  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:33.277144  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:34.270147  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:34.271072  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:35.273905  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:35.275044  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:36.276197  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:36.276825  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:37.278329  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:37.279011  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:38.280237  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:38.280858  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:39.282813  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:39.283381  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:40.285058  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:40.285914  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
04/07-23:42:41.288001  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.8 -> 192.168.1.7
04/07-23:42:41.288931  [**] [1:10001:0] ICMP Ping Detected [**] [Priority: 0] {ICMP} 192.168.1.7 -> 192.168.1.8
^Z

```

Figure 2 Snort Command Line

Snort despite being a free and opensource software that can be deployed by either an individual for testing and learning, researching and purposes or a large organization on their network environment for increasing security, it comes with some predefined rules which is the community ruleset however to get sophisticated rules for detecting sophisticated and complex cyber-attacks the users need to register for Snort Subscriber Ruleset which is managed and is developed, tested, and approved by Cisco Talos this indicates that snort is not completely free as snort offers tiers different tiers for rule subscriptions. The different tiers of snort for comprehensive and sophisticated rules are:

Personal

Personal subscriptions are perfect for users who only need to use Snort in one place. This is only for students or home network environment users.

Business

Business subscriptions are perfect for companies, non-profits, universities, government agencies, etc. that need to deploy Snort across a wide variety of devices and need to

protect a large network. This includes environments where Snort sensors are used in a production or lab environment.

Integrator

Integrator subscriptions are perfect for anyone looking to integrate Snort into their application or server for commercial resale purposes.

Apart to pay for getting the latest and the comprehensive rules, snort is terminal based NIDS/IPS meaning it is completely command line based and doesn't have any Graphical User Interface (GUI) for easier navigation and easy use. To use a GUI for snort the network or system administrators need to do a lot of configuration and need to use external services like SIEM or ELK stack which abstracts and hides a lot of raw data by default.

2.4.2. Suricata

Like snort Suricata is another opensource Network Intrusion Detection System (NIDS) and Network Security Monitoring (NSM) tool which was developed by the Open Information Security Foundations (OISF). Suricata is a high performance and a multi-threading application or a tool which allows for efficient processing of large volume of network traffic. It can also perform deep packet inspection and has several features like packet separation and analysis, file extraction and HTTP logging. Like snort Suricata is also a rule-based detection system and is compatible with the snort rules.

Suricata GitHub link: <https://github.com/OISF/suricata.git>

Suricata is far more than an IDS/IPS



Source: Stamus Networks

Figure 3 Suricata Logo (suricata.io, 2025)

The features of Suricata are:

- Intrusion Detection/Prevention System (IDS/IPS)
- Network Security Monitoring
- Deep Packet Inspection
- Multi-threading processes
- Protocol Specific Parsing
- Custom Rule engine
- Logging and Alert generation
- Opensource and Community Support

Like snort, Suricata also has different tiers for their rulesets which are free tier which includes the community rules and registered user rules and paid tier.

Free Rules

Community Rules: These are contributed by the open-source community and is maintained by the Snort team. They are available for free and provide a basic detection capability.

Registered User Rules: These rules are released by the Snort team and are made available to registered users after a certain delay (usually 30 days from their initial release for paid subscribers).

Paid Rules

Subscription Rules (Snort Subscriber Ruleset): These are the most up-to-date and comprehensive set of rules. Subscribers who pay for this service receive new and updated rules as soon as they are released, without any delay. This subscription provides access to the latest threat intelligence and protection.

The paid rules offer more timely updates and often include advanced threat detection capabilities, which are particularly useful for organizations that require up-to-date security measures.

2.4.3. FortiGate Firewall

FortiGate is a next generation firewall that is developed and is sold by Fortinet. It includes major features like intrusion detection and prevention systems, antivirus, web filtering and even VPN capabilities to secure and restricted network environment from various cyber-attacks. FortiGate devices are known for their high-performance scanning and comprehensive threat detection alongside their ease of management.

Features of FortiGate Firewall

- Deep packet inspection
- Application control
- Intrusion Detection and Prevention System
- Optimizes WAN traffic and it also ensures a secure connectivity channel for remote sites.
- Blocks access to malicious or inappropriate websites.
- Scans and blocks viruses, malware, and other threats in real-time.
- Decrypts and inspects encrypted traffic to detect hidden threats.



Figure 4 FortiGate Dashboard (fortinet.com, 2025)

Despite FortiGate firewalls have a lot of positive side however they also have a lot of drawbacks, the first one is the licensing structure can be overly complicated and costly it also has different tiers for various features. The extensive features that are present in FortiGate firewalls may require a steep learning curve for new users. Some features, like deep packet inspection and SSL inspection can be very computationally demanding and resource-heavy and it may result on impacting the performance on lower-end models. The initial investment and ongoing costs for subscriptions and support can be high, especially for smaller organizations, the user interface can also be overwhelming for beginners due to the many numbers of options and different numbers of configurations. The quality of customer support can be very different depending on the region and specific support plan. When updating the system firmware some users may report issues which can cause unexpected behaviour or require careful management.

FortiGate firewalls are robust and feature-rich, but organizations need to weigh the costs and complexity against their specific security needs and resources.

2.5. Comparisons

S.N.	Features	Snort	Suricata	FortiGate Firewall	VISTA
1.	Free to use	Yes	Yes	No	Yes
2.	Real time packet inspection	Yes	Yes	Yes	Yes
3.	Multithreading	No	Yes	Yes	Yes
4.	Deep Packet Inspection	Yes	Yes	Yes	Yes
5.	Has any GUI Dashboard	No	No	Yes	Yes
6.	Easy to Use	Yes	Yes	No	Yes
7.	Anomaly Based Detection	No	No	Yes	No
8.	Intrusion Detection System	Yes	Yes	Yes	Yes
9.	Multi-vendor compatible rulesets	Yes	Yes	No	Yes

Key Insights of this comparison:

Snort: A highly reliable and widely used tool that provides real time packet and deep packet inspection. But it lacks multithreading support as well as a GUI dashboard, rendering it less friendly to non-technical users.

Suricata: Like Snort but provides multithreading and enhances its performance in dealing with heavy traffic. It does not have the GUI dashboard like Snort.

FortiGate Firewall: Alliance Secures provides real time packet inspection, deep packet inspection, multithreading and with GUI dashboard. It is not free to use, but is less practical for non-technical user.

VISTA: Balanced in multithreading, with a GUI dashboard and easy to use, free, and compatible with multi-vendor rulesets. Nevertheless, it does not feature the anomaly-based detection capabilities.

3. Development

3.1. Considered Methodology

First, let us know what a methodology is. A methodology is a structured arrangement of methods and principles for a particular line of development. A methodology in the field of

software development is proposed to achieve a specific aim. It helps in maintaining an effective and plausible strategy for performing works in education, research, and project design (Badkar, 2024).

For the purpose of my final year project I did a lot of research on software development lifecycles and different methodologies. Some of the considered SDLC methodologies are explained below which can be a suitable alternative for my project.

3.1.1. Scrum

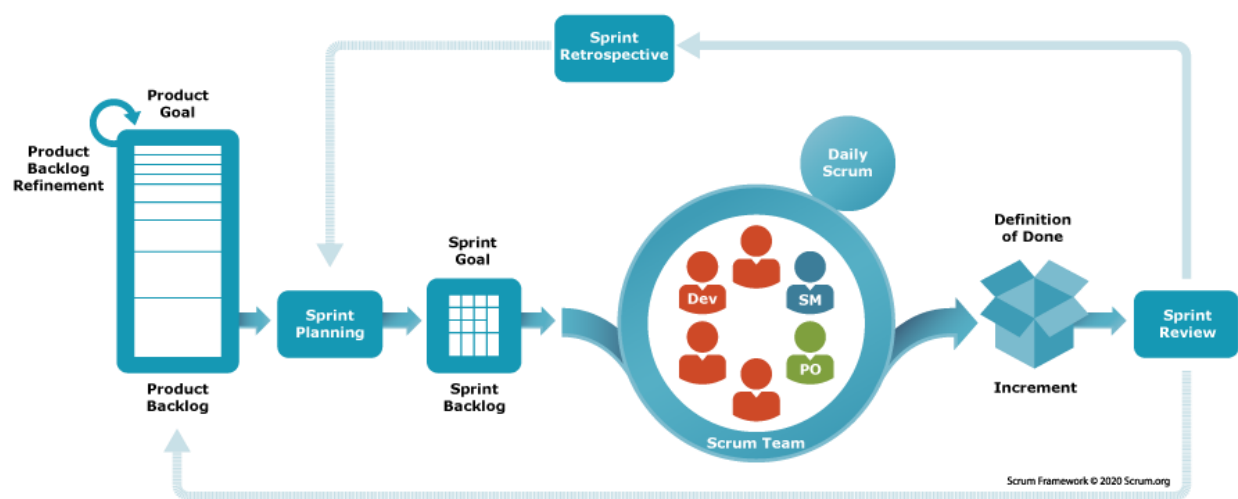


Figure 5 Scrum Methodology (scrum.org, 2025)

Agile is a methodology that Scrum falls under to help teams in developing collaboration and doing work more efficiently. It provides clear rules, roles, and values to help the team keep direction. Firstly, the team creates a list of all activities to be carried out (Product Backlog). Then, they select activities for just a brief period (Sprint) and decide how they are going to achieve them. In everyday meanings, team come together and discuss their work done and further proceedings. At end of Sprint, they show to others what has been done (Sprint Review). Then, the team mentioned what went well and what needs improvement (Sprint Retrospective). Then start next Sprint and keep improving work. Scrum enables the team to operate quickly and learn how to improve during the task completion (scrum.org, 2025).

- **Steps of Scrum Methodology is given below:**
 - Product Backlog
 - Sprint Planning
 - Sprint
 - Sprint Review
 - Sprint Retrospective
 - Repeat
- **The Advantages of Scrum Methodology are:**
 - In Scrum methodology, new Requirements can be easily added.
 - Early errors can be detected very quickly.
 - Choosing Scrum methodology risk factors easily reduced.
 - Scrum methodology helps to do work faster.
 - Scrum methodology always prioritizes customer needs first.
 - Scrum methodology helps in better team collaboration.
- **The disadvantages of the Scrum Methodology are:**
 - Scrum methodology needs an experienced team very much.
 - In Scrum methodology, it is tough to manage big projects.
 - Choosing the Scrum methodology requires full commitment.
 - In Scrum methodology, frequent changes can be confusing while doing work.
 - In Scrum methodology, meetings take lots of time to finalize the project requirements.

- The reason for not choosing scrum methodology for my final year project is given below:

Table 1 Reasons for Scrum methodology

Reason	Details
Single-Person Project	This Final Year Project is a single-person project, so I do not need too many meetings which are time-consuming.
Straightforward Website	My website is a very straightforward website which has well-defined features in which scrum iterative process may not be needed.
No-Frequent Changes Needed	VISTA does not require a lot of frequent features changes and adjustments in development process, since all the functional and nonfunctional requirements are already defined.
Risky for Legalizations	Scrum gives more importance for making software than making documentation of the project, which is very much risky for any software projects legalizations.

3.1.2. Waterfall model

The waterfall method is a traditional process utilized in the production of engineering software and products. Work descends like water and has a pattern that must be followed. Unless and until one step is not over, we cannot jump into the other step. It begins with concern and assessment, followed by processing, checking, developing, installation, evaluation, and sustaining utilization later (lucidchart.com, 2025).

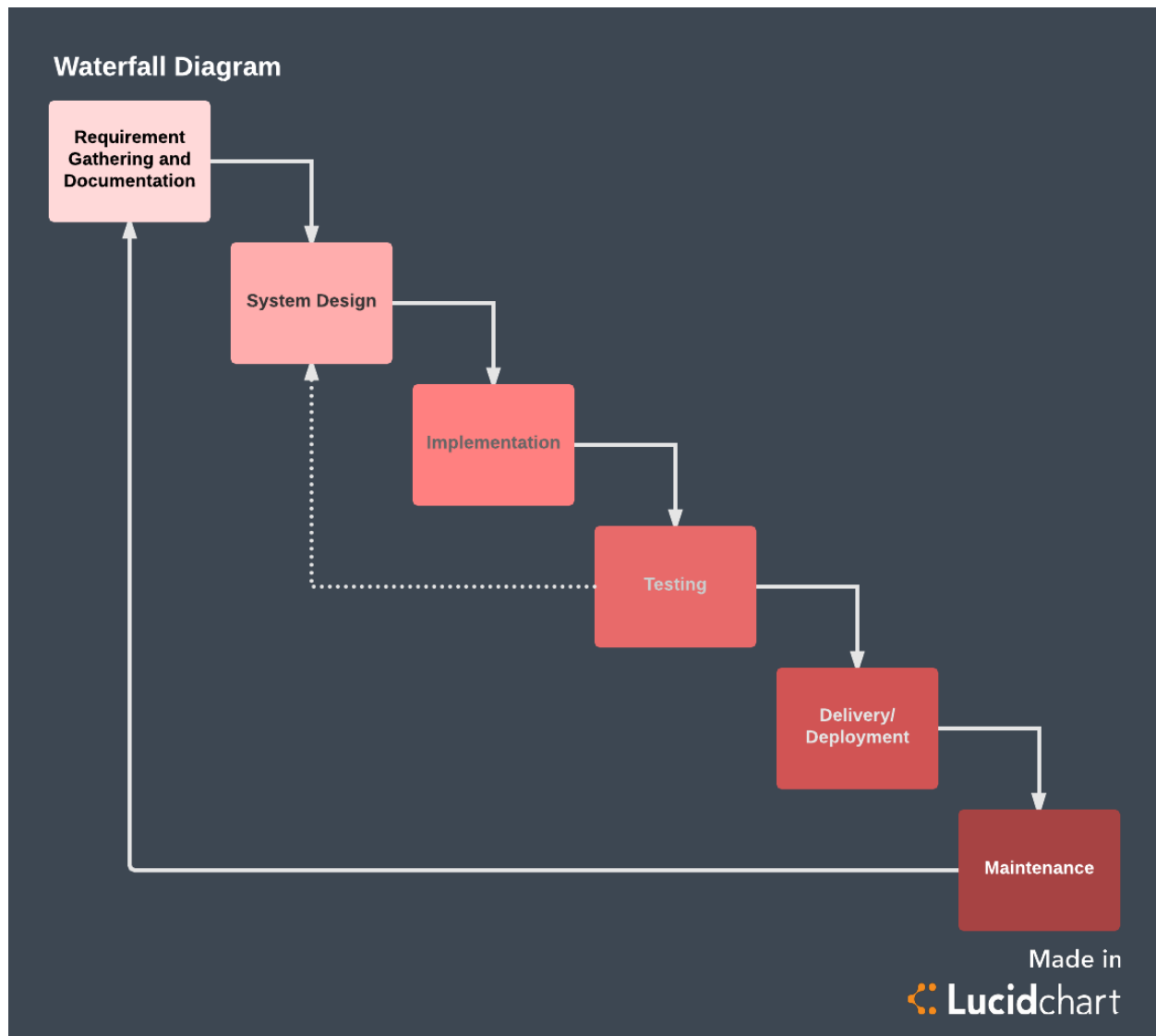


Figure 6 Waterfall model (lucidchart.com, 2025).

- **The steps of the waterfall methodology are given below:**
 - Requirement Gathering
 - System Design
 - Implementation
 - Testing
 - Deployment
 - Maintenance
- **The Advantages of the Waterfall methodology is:**
 - The waterfall is simple and easy to use it has clear steps.

- The waterfall is incredibly good for small projects.
- The waterfall has clear plans and makes it easy to track the progress of the project.
- The waterfall phases have fixed goals.
- The waterfall is time saving because it finishes one phase before starting another.

➤ **The Disadvantages of the Waterfall methodology are listed below:**

- In waterfall methodology once the phases are completed it is difficult to change again.
- The methodology is made for only small-scale projects, it cannot be used in large scale projects.
- The Waterfall methodology is very time consuming because after finishing one phase then only we can move to another phase.
- The methodology is not good for those projects that may change or must change.

Reason for not using the Waterfall Methodology in my project:

Table 2 Reasons for not using Waterfall model

Reason	Details
Hard to Change or Add Features	In this methodology, it is extremely hard to change in case or add new features.
No Early Testing	There will be no early testing, so errors cannot be found until the end.
Time-Consuming Error Fixing	Errors can be detected at the end only and solving them takes lots of time and effort.
Delayed Feedback	Customers cannot give feedback and ideas until the project is complete.
Risky for Web Projects	Web projects need open testing and response, but it does not allow that.

3.1.3. Prototype Methodology

The prototype methodology is making a sample model first before the final product comes. This model is a testing idea, check if works well. People use prototypes to fix problems, improve better, and save time or money before e real things made. Simple way to see what works or does not work.

- **The steps of the Prototype methodology is given below:**
 - Gather Requirements
 - Create Prototype
 - Test Prototype
 - Improve Prototype
 - Final Product
 -
- **The Advantages of the Prototype Methodology are:**
 - The Prototype easily shows ideas, and people understand fast.
 - Good for projects when the user is not sure what wants.
 - Users give feedback early, making a better final product.
 - Find problem fast, fix before too late.
 - Save money, and no big mistake later.
- **The Advantages of the Prototype Methodology are:**
 - Take a long time if the user keeps changing mind.
 - Cost more if make many prototypes.
 - Users think prototypes are real products and get confused.
 - Spend too much time on prototype, real product late.
 - Not good for big or complex projects.

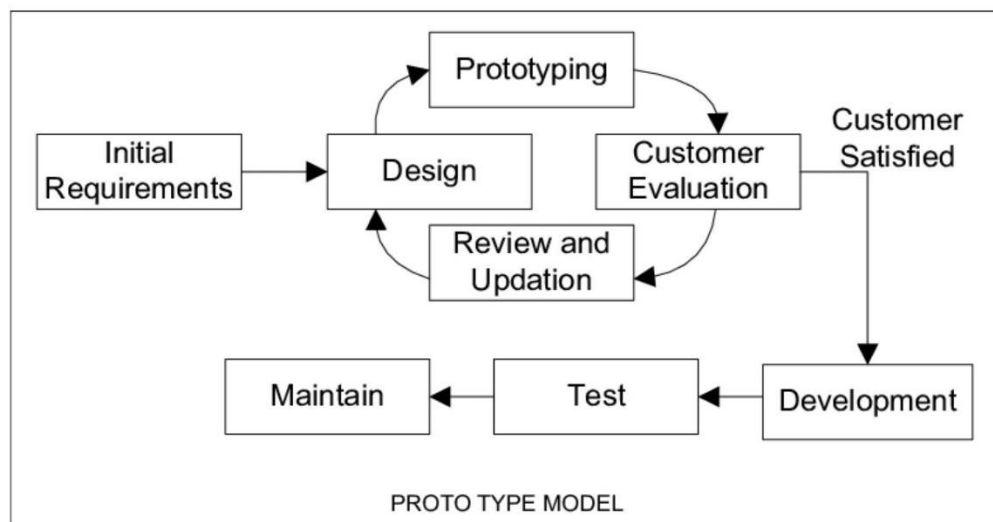


Figure 7 Prototype Model (BoardInfinity.com, 2025)

Reason for not choosing the Prototype Methodology in my project:

Table 3 Reasons for not using Prototype model

Reason	Details
Hard to Manage Changes	Users ask to change many times, very hard to manage work.
Extra Cost for Many Models	Making many prototypes cost too much money and time.
User Confusion	Users think the prototype is a real product and get confused.
Delay in Real Development	Spend too much time on prototypes, real products come late.
Not Good for Big Projects	Prototypes do not work well for big or very hard projects.

3.2. Selected Methodology

For my final year project, I have decided to use the Rational Unified Process, I have explained below the reasons:

3.2.1. RUP

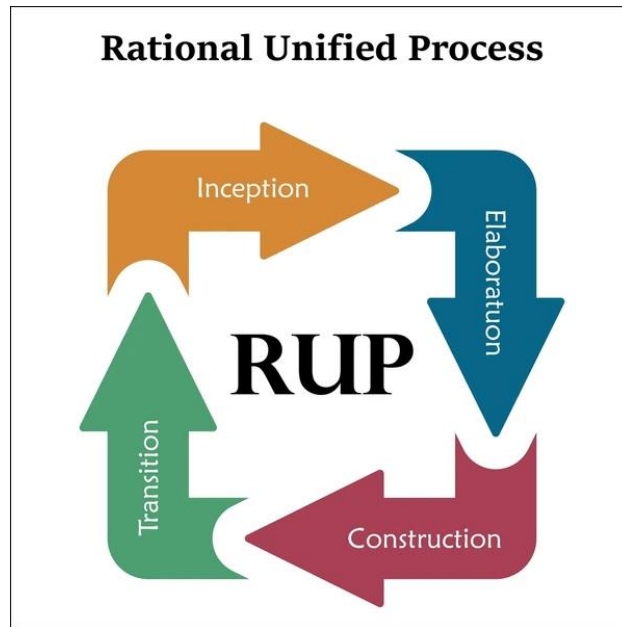


Figure 8 Rational Unified Process (Graphics, 2022)

For the overall development and completion of this project, I would be choosing the RUP (Rational Unified Process) methodology. The Rational Unified Process (RUP) is considered an agile software development methodology because it is iterative. It uses a flexible and adaptive system in software development that involves adjusting and repeating cycles until requirements and objectives are met (Minott, 2023). Since it is iterative and allows for flexible adaptation during the execution and completion of the project. Also, I had previously used the same methodology during my Second Year Project, so I have a better understanding of its procedure.

This methodology is the best fit for projects like these which require continuous feedback, flexibility, and incremental development. This methodology has four stages which are inception, elaboration, construction, and lastly transition. Every stage has its unique responsibility, and every stage is vital for the completion of the project. Since this methodology is performed in an iterative way which made my project's

overall development and completion efficient and convenient, it also enhanced my plans regarding the transition phase.

Steps of the RUP Methodology is:

- Inception
- Elaboration
- Construction
- Transition

- **Inception Phase**

This is the very first phase of this RUP approach required requirements are collected, and the feasibility study of the project is calculated by identifying and defining users along with their roles, and features. In this phase the non-physical

The idea of this project begins. Also, project planning and overall resources and requirements analyses are performed during this phase of RUP methodology. The inception phase of this project is composed of:

- Identify stakeholders and roles
- Define project goals and scope
- Risk analysis
- Requirement gathering
- Prepare Project Chart and Proposal
- Proposal Verification

- **Elaboration Phase**

This is the second phase of this RUP approach major diagrams and SRS documents are designed which are required for the completion of the elaboration phase. This phase involves a more detailed evaluation of the development plan with the aim of further minimizing associated risks that were assumed in the first stage of the development (Minott, 2023). Moreover, planning for development, and designing of the project is executed in this method. This phase consists of the following steps for this project:

- Design System Architecture

- Plan System Components
- Identify and Finalize Tools and Libraries
- Creation of Use Case Diagram
- ERD Development
- Wireframe UI/UX Development
- Complete SRS Documentation
- Create UML Diagrams
- SRS Documentation Verification

- **Construction**

Construction is the third phase of this RUP methodology. This phase of the RUP methodology is associated with the completion of the overall development of this project. In this phase front-end, back-end, database, and prediction tool must be developed along with overall testing of the system.

- Development of PCAP file Analyzer Module
- Development of Network Packet Capture Module
- Development of Packet Parsing and Rule Matching Module
- Development of Alerting and Logging Module
- Development of Network Device Identification and Visualization Module
- Development of Frontend
- Database Integration
- Development of Backend
- Unit Testing
- System Testing
- Complete Working Features

- **Transition**

This is the last phase of this method. In this phase, the deployment of the finalized system is initiated which comes after the construction phase. This phase plays a vital role being the last phase (Watt, 2020). Here, deployment of the platform is performed, and a feedback form is created for required bug fixing and maintenance. The transition phase includes the following steps in this system:

- Final System Testing
- Make Final Documentation
- Ongoing System Monitoring and Performance Optimization
- Deployment
- Bug Fixes and System Maintenance
- Submission of Project
- **The Advantages of the RUP Methodology are:**
 - In RUP methodology work is done in small parts so it is easier to plan.
 - The RUP methodology is flexible because we can change or add features in the middle of the work.
 - In RUP methodology, the supervisor can give feedback in middle of the work and can be changed during the project to make project better.
 - In RUP methodology problems can be found easily because they are divided into each section and can be solved easily.
 - The RUP methodology also saves time and cost by finding the risk which avoids costly fixers later.
 - In RUP methodology testing will be performed in every phase so there will not be any tension of code error or project failure.
- **The Disadvantages of the RUP methodology are:**
 - In RUP, too many steps make it hard for small projects.
 - RUP takes too much time because every phase needs to finish before moving to the next.
 - RUP can be hard to understand if not familiar with the process.
 - Managing RUP is difficult because there are so many phases, it is hard to keep track.
 - RUP can be expensive because a lot of planning and documents are needed.
- **Reason for choosing the RUP Methodology in my project:**

Table 4 Reasons for Choosing RUP

Reason	Details
Better Planning	RUP allows me to plan well because it breaks the work into smaller, manageable parts.
Flexibility	It also gives flexibility to adjust and add new features if needed during the project.
Early Feedback	I can get feedback from the supervisor early in the process, which helps improve the project.
Quick Problem Solving	Since work is divided into smaller parts, finding and fixing problems quickly is easier.
Risk Management	This approach helps to save time and costs, as risks are caught early.
Continuous Testing	With testing happening at every phase, I can ensure the project is always on track.

3.3. Methodology Comparison

A basic justification for comparing each of the chosen approaches with each of the other methodologies that were considered have been noted down below.

Table 5 Methodology Comparison

Features	RUP	Scrum	Waterfall
Flexible to change.	✓	✓	X
Fast Delivery	✓	✓	X

Client Involvement	✓	✓	X
Good for big projects.	✓	✓	X
New Idea addition	✓	✓	X
Huge Team Required	X	✓	✓

4. Progress

As I mentioned earlier in this Final Year Project will be done using the RUP methodology. The project was approved and then started with the VISTA features with the discussion with the supervisor, and finalization. System design charts such as System architecture, Use Case diagram, Sequence, collaboration diagram, ERD was completed before the development process following the RUP methodology, which is available in the appendix.

4.1. Progress Table

The progress table helps to determine the actual progress of the project

Table 6 Progress table

S. N	Task	Status	Progress
1	Identify stake holders and roles	Completed	100%

2	Define project goals and scope	Completed	100%
3	Risk analysis	Completed	100%
4	Requirement Gathering	Completed	100%
5	Prepare project chart and proposal	Completed	100%
6	Proposal verification	Completed	100%
7	Design system architecture	Completed	100%
8	Plan system components	Work in progress	50%
9	Identify and finalize tools and libraries	Completed	100%
10	Creation of Use Case Diagram	Completed	100%
11	ERD Development.	Completed	100
12	Wireframe UI/UX development	Completed	100%

13	Complete SRS Documentation	Completed	100%
14	Create UML Diagrams	Work in Progress	100%
15	SRS Documentation Verification	Completed	100%
i	Feature 1: Development of PCAP file analyzer module	Complete	100%
ii	Feature 2: Development of Network Packet capture module	Work in Progress	25%
iii	Feature 3: Development of Packet Parsing and Rule matching module	Incomplete	0%
iv	Feature 4: Development of Alerting and Logging Module	Incomplete	0%
V	Feature 5: Development of Network Device Identification and Visualization module	Incomplete	0%
Vi	Feature 6: Development of Frontend	Work in progress	20%
Vii	Feature 7: Database Integration	Work in progress	50%

Viii	Development of Backend	Work in progress	15%
16	Unit testing	Incomplete	0%
17	System testing	incomplete	0%
18	Complete working features	Incomplete	0%
19	Final system testing	Incomplete	0%
20	Make final documentation	Incomplete	0%
21	Ongoing system monitorization and performance optimization	Incomplete	0%
22	Deployment	Incomplete	0%
23	Bug fixes and system maintenance	Incomplete	0%
24	Submission of the project	Incomplete	0% ^s

4.2. Development

As mentioned above, this project will follow the RUP Methodology. All the development is divided into phases where tasks must be completed and must be reviewed by the

supervisor. I have done some wireframes and authentication of both customer and worker which I will be showing down below:

4.2.1. UI/UX pages

❖ Login Page

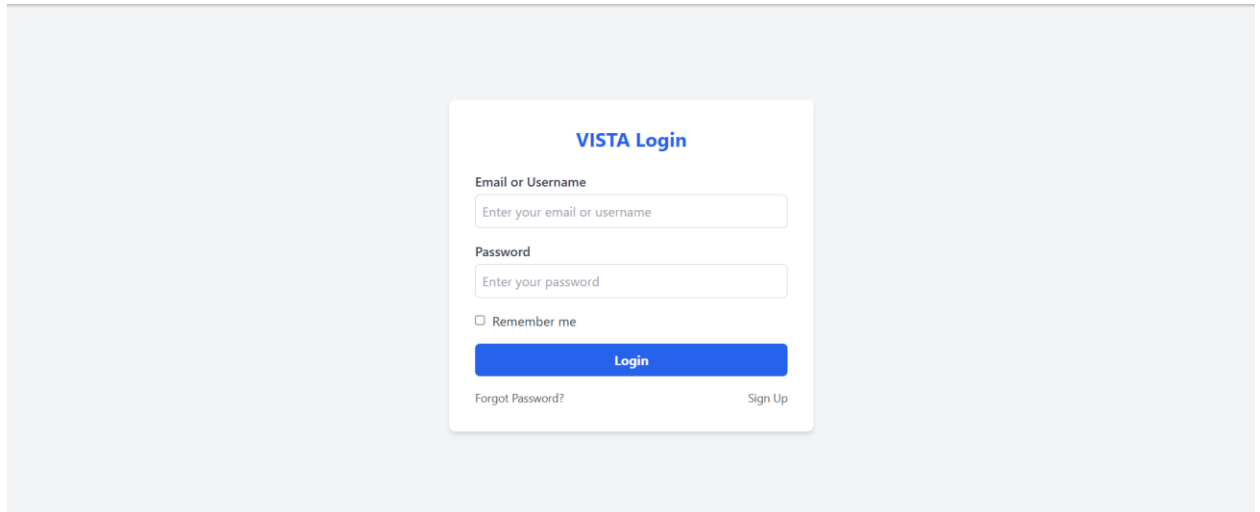


Figure 9 Login UI/UX

❖ User Dashboard

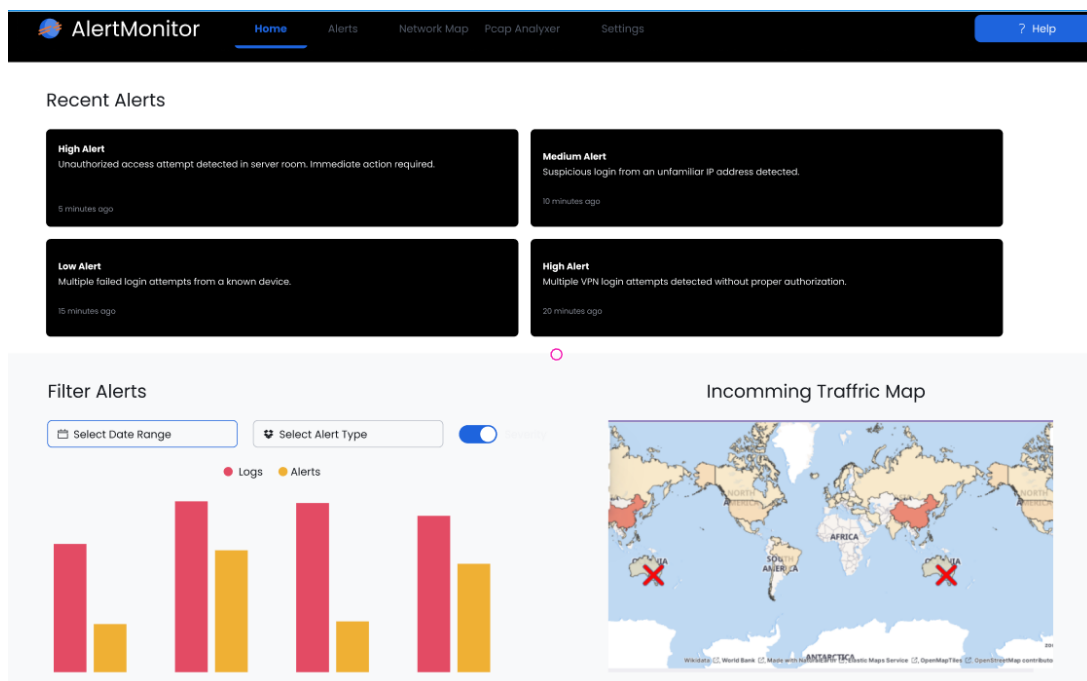


Figure 10 Home Dashboard

❖ Alerts Page

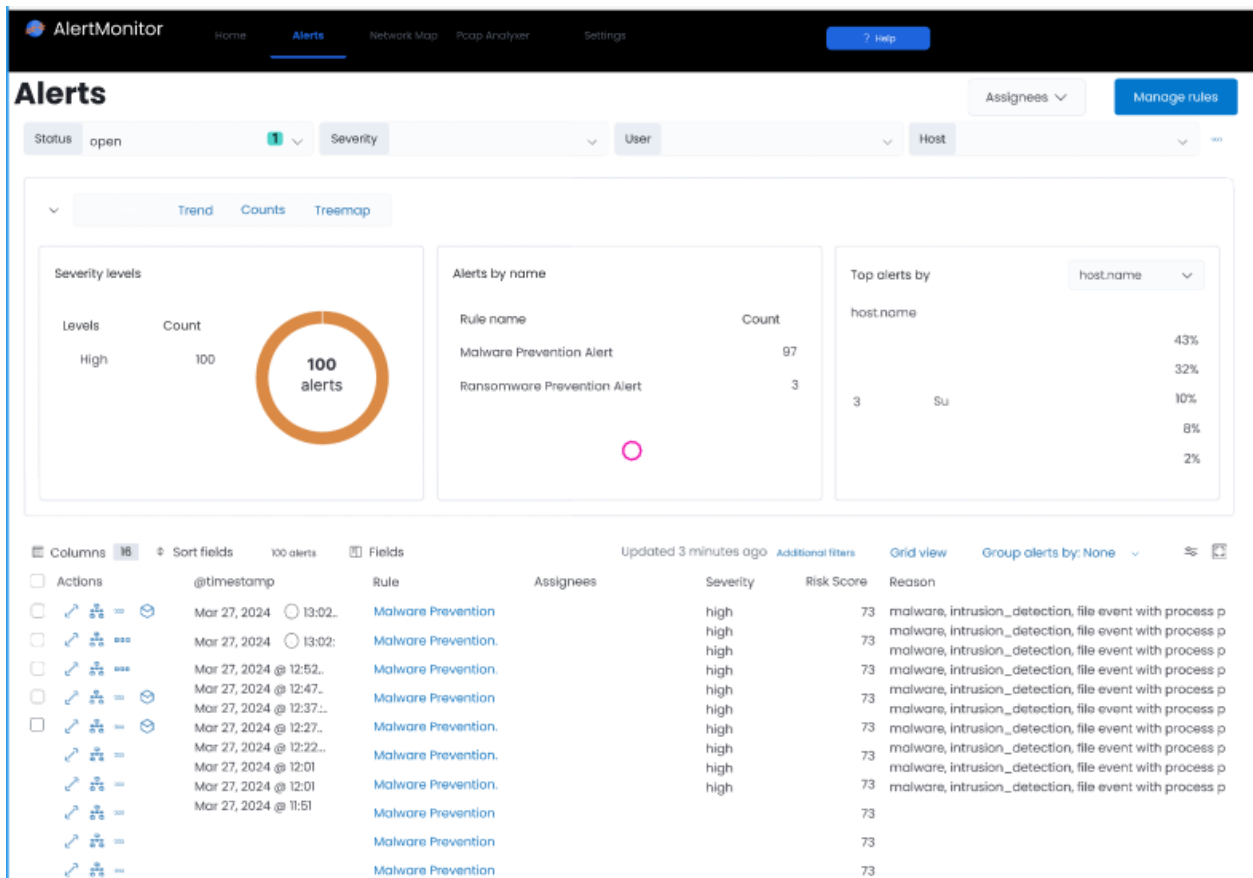


Figure 11 Wireframe of Alerts Page

❖ PCAP Upload Page

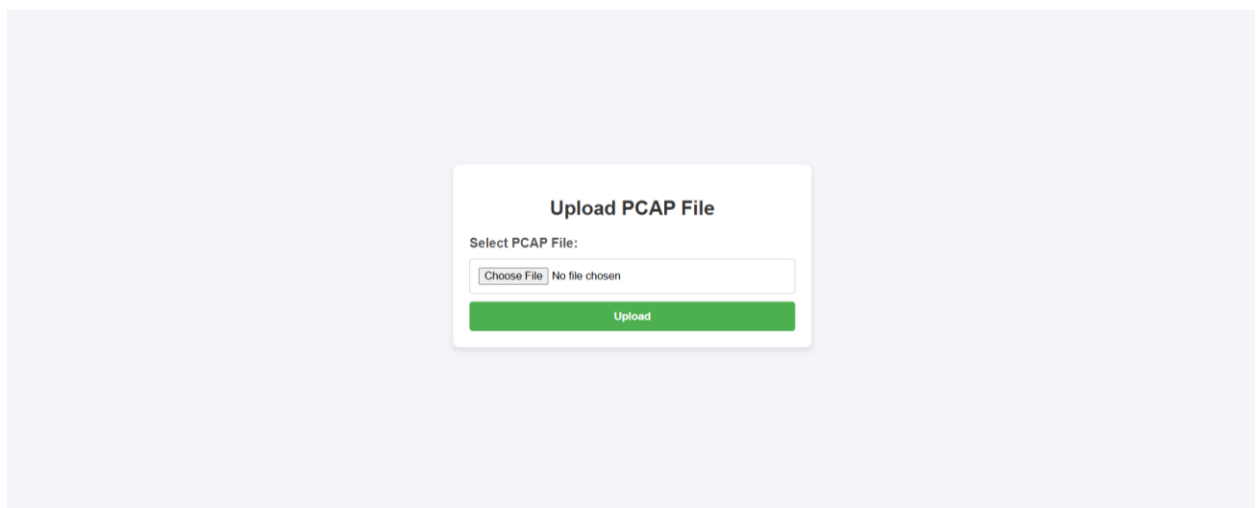


Figure 12 Wireframe of Upload PCAP page

❖ PCAP Result

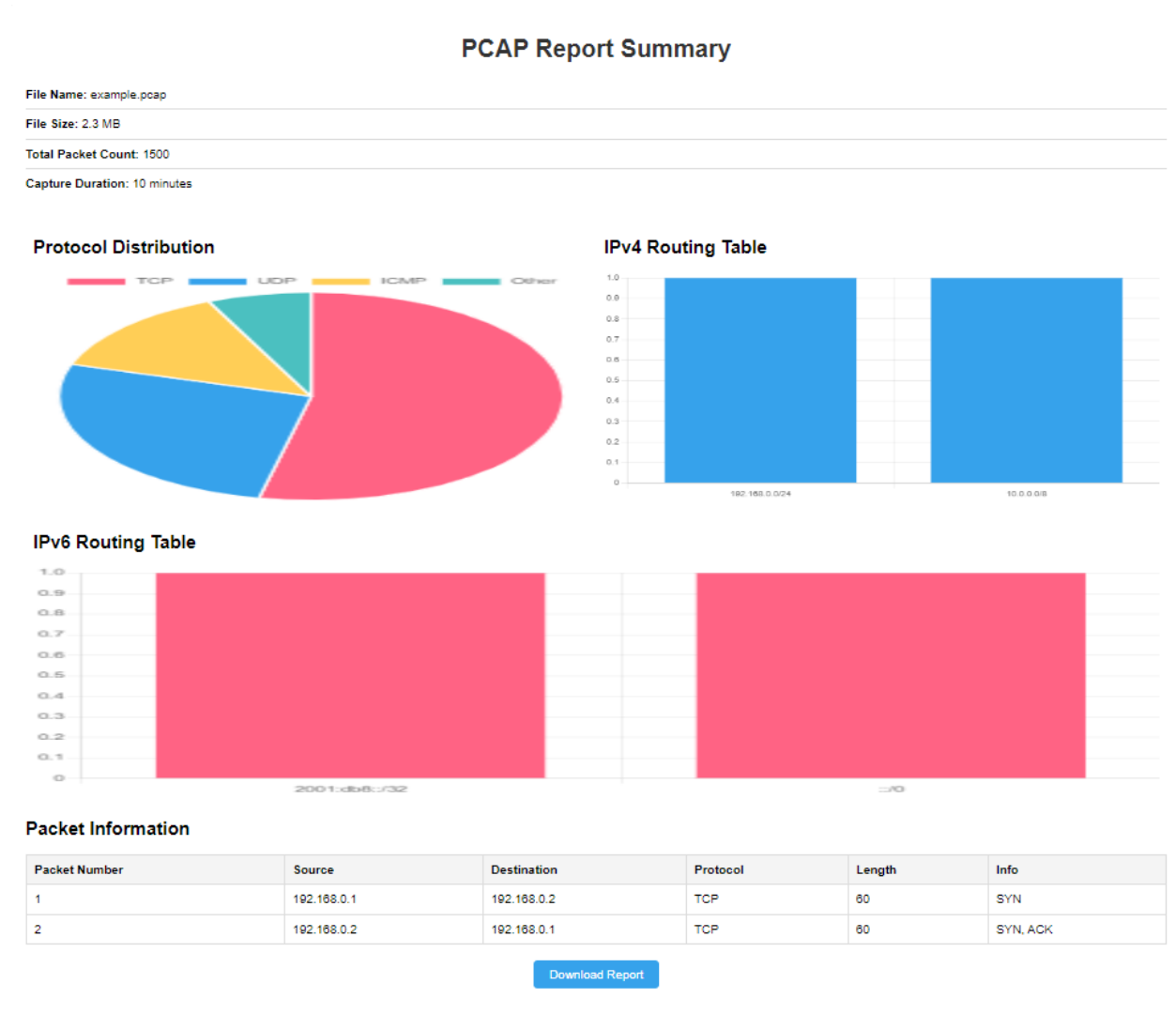


Figure 13 PCAP Summary Report page

❖ Network Source and Destination address Map

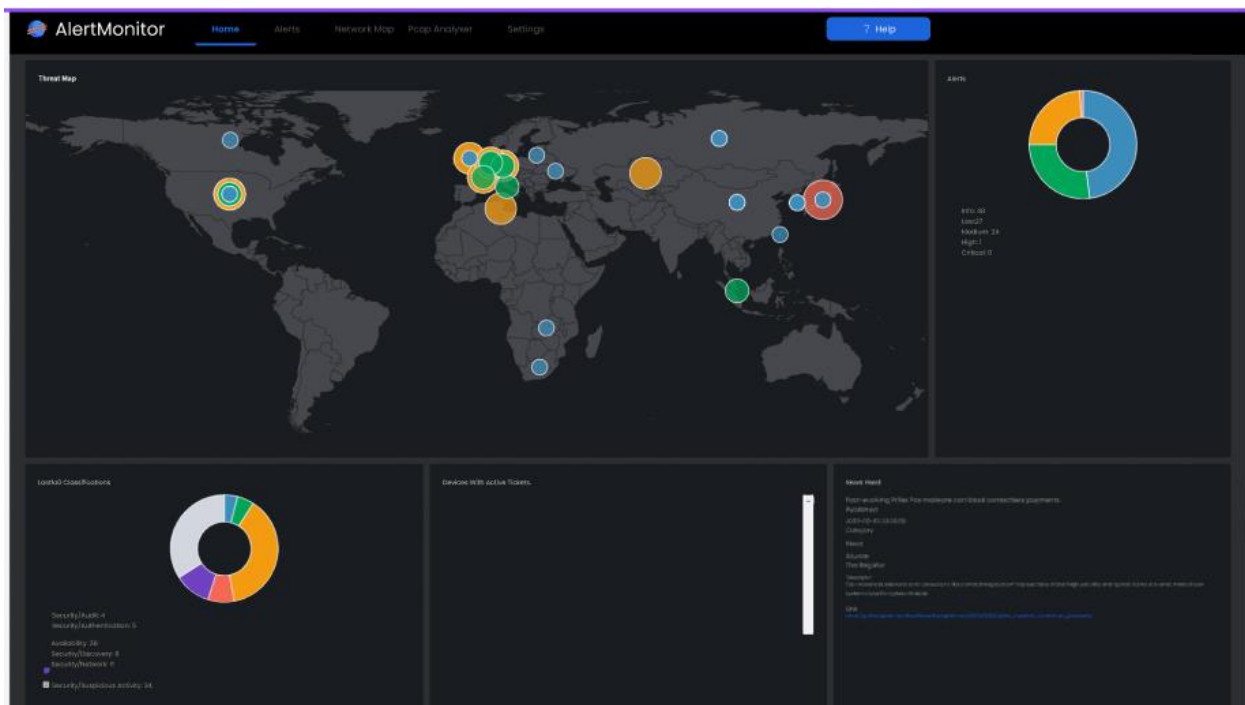


Figure 14 SRC/Destination Address Plotting

4.2.2. Backend Testing

Custom user model with added columns

```

1  from django.db import models
2  from django.contrib.auth.models import AbstractUser
3
4  # Create your models here.
5  class User(AbstractUser):
6      class Role(models.TextChoices):
7          ADMIN = 'ADMIN', 'Admin'
8          OPERATOR = 'OPERATOR', 'Operator'
9
10     #setting a default role for new users
11     role = models.CharField(max_length=15, choices=Role.choices, default=Role.ADMIN)
12     phone_number = models.CharField(max_length=15, blank=True, null=True)
13     profile_picture = models.ImageField(upload_to='profile-images/', null=True, blank=True)
14     emails_verified = models.BooleanField(default=False)
15     def save(self, *args, **kwargs):
16         #If the user is being created and no role is given then set the role to ADMIN
17         if not self.pk:
18             self.role = self.role or self.Role.ADMIN # Ensure role is set if not provided
19         if self.password and not self.password.startswith("pbkdf2_"):
20             self.set_password(self.password)
21         return super().save(*args, **kwargs)
22

```

Figure 15 User Model Django

```
# Create your views here.
def landing(request):#Login function
    #checking if the user is logged in or not
    if request.user.is_authenticated:
        return redirect('home')
    else:
        if request.method == "POST":
            print('trying login')
            userName = request.POST.get("user-name")
            password = request.POST.get("password")
            print(f'{userName} and {password}')
            user = authenticate(request, username=userName, password=password)
            if user is not None:
                print("success")
                login(request, user)
                return redirect(coreView.home)
            else:
                messages.error(request, 'Invalid username or password')
        return render(request, 'login.html')
```

Figure 16 Login View Django

```
def logoutUser(request):
    logout(request)
    #clearing all the session data
    request.session.clear()
    response = HttpResponseRedirect("You have been logged out.")
    #clearing all the user's cookies
    cookies = request.COOKIE
    for cookie in cookies:
        response.delete_cookie(cookie)
    messages.success(request, "You have been logged out.")
    #user lai redirect gareko home page ma after logout
    return redirect('/')

def error404(request, exception): #this function will handle all the 404 errors for this app
    return render(request, 'error-404.html', status=404)
```

Figure 17 Logout View Django


```

68 def registerUser(request):
69     #check if the user is an admin
70     if not request.user.is_staff:
71         messages.error(request, "You do not have permission to register an operator.")
72         return render(request, 'register.html')
73
74     if request.method == 'POST':
75         #get data from the form
76         first_name = request.POST.get('firstname')
77         last_name = request.POST.get('lastname')
78         username = request.POST.get('username')
79         email = request.POST.get('email')
80         phone = request.POST.get('phone')
81         address = request.POST.get('address')
82         password = request.POST.get('password')
83
84         #default role is "OPERATOR"
85         role = "OPERATOR"
86         try:
87             #create the user
88             user = User.objects.create_user(username=username, email=email, first_name=first_name, last_name=last_name, password=password)
89             #you may want to save additional fields like phone_number and address
90             user.profile.phone_number = phone
91             user.profile.address = address
92             user.profile.role = role #assuming you have a Profile model that stores the role
93             user.save()
94             messages.success(request, "Operator Registered Successfully")
95         except Exception as e:
96             messages.error(request, f"User Registration Failed: {str(e)}")
97
98         return render(request, 'register.html')
99     else:
100         return render(request, 'register.html')

```

Figure 18 Admin's Operator Registration View

```

admin.py x
D: > Aayush Wanem Limbu > FYP-Code > web-app > authentication_app > admin.py
1 from django.contrib import admin
2 from django.contrib.auth.admin import UserAdmin as DefaultUserAdmin
3 from .models import User
4
5 #custom User Admin class
6 class UserAdmin(DefaultUserAdmin):
7     # Define fields to display in the list view
8     list_display = (
9         'username', 'email', 'role', 'emails_verified', 'phone_number',
10        'profile_picture', 'is_active', 'date_joined'
11    )
12    list_filter = ('role', 'is_active', 'is_staff', 'date_joined')
13    search_fields = ('username', 'email', 'phone_number')
14    ordering = ('date_joined',)
15
16    #extended fieldsets to include custom fields
17    fieldsets = DefaultUserAdmin.fieldsets + (
18        (None, {'fields': ('role', 'phone_number', 'profile_picture')}),
19    )
20
21    #add a custom fields for the 'Add User' form in admin
22    add_fieldsets = DefaultUserAdmin.add_fieldsets + (
23        (None, {
24            'classes': ('wide',),
25            'fields': ('role', 'email', 'phone_number', 'profile_picture'),
26        }),
27    )
28
29 #registering my custom User model with the admin site
30 admin.site.register(User, UserAdmin)
31

```

Figure 19 Registering Custom User model to admin site

4.3. Gantt Chart

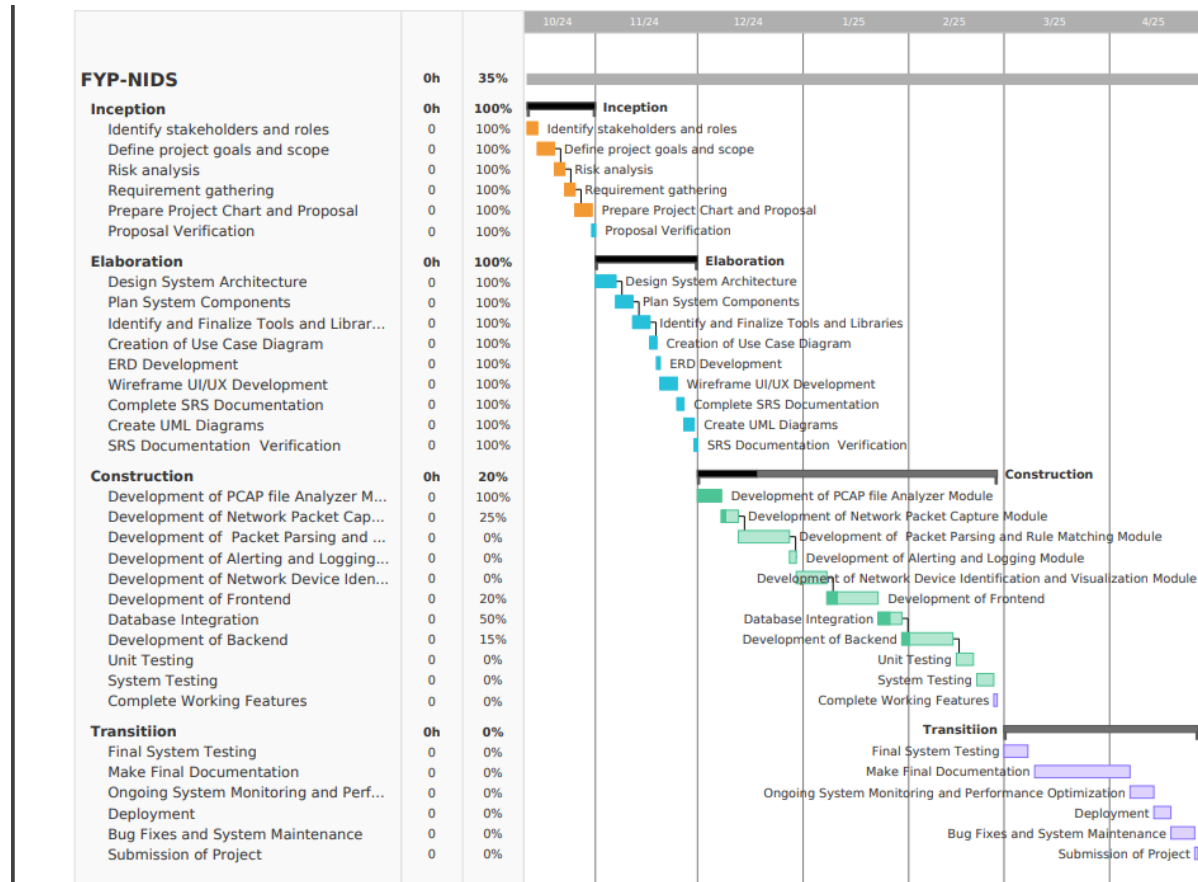


Figure 20 Updated Project Gantt Chart

4.4. Work Breakdown Structure

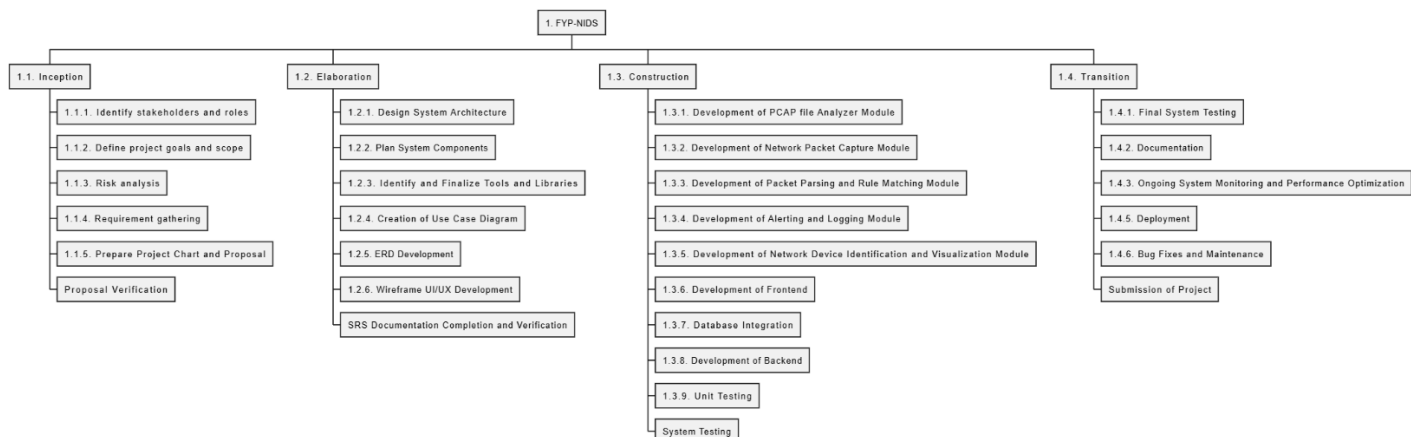


Figure 21 Project WBS

5. Further Work

Each of the initial phases of the project worked had made some good progress – requirements gathering, design and planning. But there are some critical development and testing tasks. The following areas will be the focus of future work:

Development:

Network Packet Capture Module (Task ii): It is currently 25% complete. The remaining work includes:

This tends to settle down to finalizing packet capture mechanisms. It is to develop efficient packet buffering and storage technologies. For testing, we load the module under different network loads.

Packet Parsing and Rule Matching Module (Task iii): On this module 0% is completed. Development will include: I will implement packet parsing logic that will extract all the relevant data. Implementing rule matching engine to match parsed data against defined rules. Accuracy and performance testing on the module.

Alerting and Logging Module (Task iv): This module is also still not complete (0%). Work will focus on:

Deciding on the who, what, when, and how of when and where to trigger the notices.

A good logging mechanism to record the security events. Setting up triggers for alerts and notifications.

Network Device Identification and Visualization Module (Task v): This module comprises 0%. Development will involve: Network discovery protocol implementations.

Generating visualization components to show the network topology and the device information. Accurately and efficiently device identification.

Frontend Development (Task vi): The frontend is 20% complete. Further work includes:
The implementation of the user interface design. Connect frontend with the backend components. Conducting usability testing.

Database Integration (Task vii): It is about 50% completed with respect to database integration. The remaining work includes: Once we finalize the database schema. Providing data access and storage logic. Testing exposure of the database to performance and data integrity problems.

Backend Development (Task viii): The backend is 15% complete. Further work includes: Core business logic and rules. Establishing how to communicate between frontend and backend. Testing backend functionality and performance.

Testing and Deployment:

Unit Testing (Task 16): Once individual modules are developed enough, this will begin. We will test each module individually to be sure it's functioning properly.

System Testing (Task 17): Until then, we just integrate all modules. This will include testing the whole system to make sure that all the components work right.

Completing Working Features (Task 18): This is going to be an on going thing for the development phase and features are going to be implemented and tested incrementally.

Final System Testing (Task 19): All development will be completed first, and this task will be performed after this is complete to be sure the system met all requirements and ready for deployment.

Deployment (Task 22): The system is tested successfully and will be deployed to the target environment.

Bug Fixes and System Maintenance (Task 23): After deployment, we will continue to go in and perform ongoing bug fixes and system maintenance to ensure system stability and performance.

Documentation and Finalization:

Final Documentation (Task 20): After all development and testing is finished this will be completed. It will include materials about user manuals, technical documentation, project reports.

Ongoing System Monitoring and Performance Optimization (Task 21): The system is deployed, and then continuously monitored, and optimized, for performance.

Submission of the Project (Task 24): All the tasks will be completed and documented and attached to the project, which will be formally submitted.

6. References

Badkar, A., 2024. *www.simplelearn.com*. [Online]
Available at: <https://www.simplilearn.com/software-development-methodologies-article#:~:text=Software%20engineering%20methodologies%20are%20important,you%20in%20the%20technology%20field>.

[Accessed 06 01 2025].

BoardInfinity.com, 2025. *www.boardinfinity.com*. [Online]
Available at: <https://www.boardinfinity.com/blog/a-quick-guide-to-prototype-model-in-software-engineering/>

[Accessed 06 01 2025].

fortinet.com, 2025. *www.fortinet.com*. [Online]
Available at: <https://www.fortinet.com/products/next-generation-firewall>

[Accessed 06 01 2025].

Graphics, S., 2022. *Shutterstock*. [Online]
Available at: <https://www.shutterstock.com/image-vector/rational-unified-process-inception-elaboration-construction-2244279291>

[Accessed 22 11 2024].

Korkut, T., 2023. *blog.stackademic.com*. [Online]
Available at: <https://blog.stackademic.com/excelling-in-software-development-with-scrum-methodology-part-1-762a73c6a6ec>

[Accessed 06 01 2025].

lucidchart.com, 2025. *www.lucidchart.ccm*. [Online]
Available at: <https://www.lucidchart.com/blog/pros-and-cons-of-waterfall-methodology>

[Accessed 06 01 2025].

Minott, C., 2023. *Study.com*. [Online]
Available at: <https://study.com/academy/lesson/what-is-the-rational-unified-process->

methodology-tools-examples.html

[Accessed 20 11 2024].

scrum.org, 2025. *www.scrum.org.* [Online]

Available at: <https://www.scrum.org/resources/what-scrum-module>

[Accessed 06 01 2025].

suricata.io, 2025. *suricata.io.* [Online]

Available at: <https://suricata.io/>

[Accessed 06 01 2025].

Watt, A., 2020. *BCcampus.* [Online]

Available at: [https://opentextbc.ca/projectmanagement/chapter/chapter-3-the-project-](https://opentextbc.ca/projectmanagement/chapter/chapter-3-the-project-life-cycle-phases-project-management/)

[life-cycle-phases-project-management/](https://opentextbc.ca/projectmanagement/chapter/chapter-3-the-project-life-cycle-phases-project-management/)

[Accessed 20 11 2024].

www.snort.org, 2025. *www.snort.org.* [Online]

Available at: <https://www.snort.org/>

[Accessed 6 1 2025].

7. Appendix

7.1. SRS Document

Software Requirements Specifications For VISTA (Visual Intrusion Surveillance and Threat Analysis)

Version 1.0

Prepared by Aayush Wanem Limbu

Itahari International College

2025-01-04

7.1.1. Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the detailed requirements for the final year project named VISTA () which is a rule based Network Intrusion Detection System (NIDS) project. This document will serve as a guideline for the development, testing and deployment of the system that will ensure that all stakeholders have a clear understanding of the system's functionality, performance, and constraints.

7.1.2. Scope

The NIDS project aims to develop a rule-based intrusion detection system that monitors network traffic, detects known threats using predefined rules, and provides a user-friendly web dashboard for real-time monitoring and management. The system will offer capabilities such as real-time packet capture, alert generation, customizable rule management, and detailed reporting. The primary users of the system will be network administrators responsible for maintaining network security.

7.1.3. Definitions, Acronyms, and Abbreviations

NIDS: Network Intrusion Detection System

PCAP: Packet Capture

DoS: Denial of Service

UI/UX: User Interface/User Experience

SRS: Software Requirements Specification

DPI: Deep Packet Inspection

7.2.1. Overall Description

7.2.2. Product Perspective

The NIDS will be a standalone system integrating with existing network infrastructures. It will leverage open-source libraries such as Snort for rule-based detection and Django for the web dashboard. The system will be designed to handle high volumes of network traffic efficiently and provide scalability for future expansions.

7.2.3. Product Functions

Real-Time Packet Capture & Inspection: Capture and analyze live network traffic to detect suspicious activities.

Alert Generation: Notify administrators of detected threats with categorized severity levels.

Rule Management: Allow administrators to add, update, and delete detection rules.

Report Generation: Provide detailed analysis reports on network traffic and detected threats.

PCAP File Analysis: Enable historical traffic analysis through uploaded PCAP files.

7.2.4. User Classes and Characteristics

Network Administrators: Primary users responsible for configuring rules, monitoring alerts, and analyzing reports.

IT Security Teams: Users involved in investigating and mitigating security incidents.

7.2.5. Operating Environment

The NIDS will be deployed on a Linux-based server with the following specifications:

OS: Ubuntu 20.04 or later

Processor: Intel i5 11th Gen or equivalent

RAM: Minimum 8 GB (16 GB recommended)

Storage: Minimum 256 GB SSD

2.5 Design and Implementation Constraints

The system must use Python 3.11 or later and Django framework.

The web dashboard must be accessible via modern web browsers (Chrome, Firefox, Edge).

The system should support integration with MySQL for database operations.

7.2.6. Assumptions and Dependencies

Availability of network traffic data for real-time monitoring.

Regular updates to the rule database to address emerging threats.

Reliable internet connection for system updates and remote access.

7.3. Specific Requirements

7.3.1 Functional Requirements

7.3.2 Real-Time Packet Capture

Description: The system shall capture and analyze network packets in real-time using libraries like Scapy.

Inputs: Live network traffic.

Outputs: Analyzed packet data, detection alerts.

7.3.3. Alert Generation

Description: The system shall generate alerts for detected threats and categorize them by severity.

Inputs: Inspection results from packet analysis.

Outputs: Alerts with severity levels (critical, high, medium, low).

7.3.4. Rule Management

Description: The system shall allow administrators to manage detection rules through a web interface.

Inputs: Rule modifications from administrators.

Outputs: Updated rule database.

7.3.5. Report Generation

Description: The system shall generate detailed reports on network activity and detected threats.

Inputs: Network traffic data, detection results.

Outputs: Reports in PDF, CSV formats.

7.3.6. PCAP File Analysis

Description: The system shall support the upload and analysis of PCAP files for historical traffic analysis.

Inputs: Uploaded PCAP files.

Outputs: Analyzed historical traffic data, detection reports.

7.4.1. Non-Functional Requirements

7.4.2. Performance

Description: The system shall process and analyze high volumes of network traffic with minimal latency.

Requirement: Handle up to 1 Gbps traffic without performance degradation.

7.4.3. Scalability

Description: The system shall be scalable to accommodate growing network sizes and traffic volumes.

Requirement: Support horizontal scaling with additional servers.

7.4.4. Usability

Description: The web dashboard shall provide an intuitive interface for administrators.

Requirement: User-friendly UI/UX with real-time updates and interactive visualizations.

7.4.5. Security

Description: The system will ensure a secure access and data protection.

Requirement: Implement authentication, authorization, and data encryption.

7.5.6. Maintainability

Description: The system shall be easy to maintain and update.

Requirement: Modular code structure with comprehensive documentation.

7.2. Data Flow Diagram

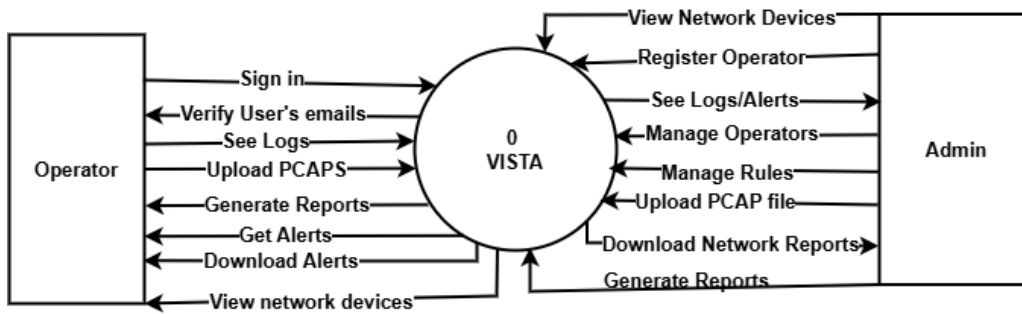


Figure 22 Level 0 DFD

7.3. Use Case

7.3.1. Use Case Diagram of System

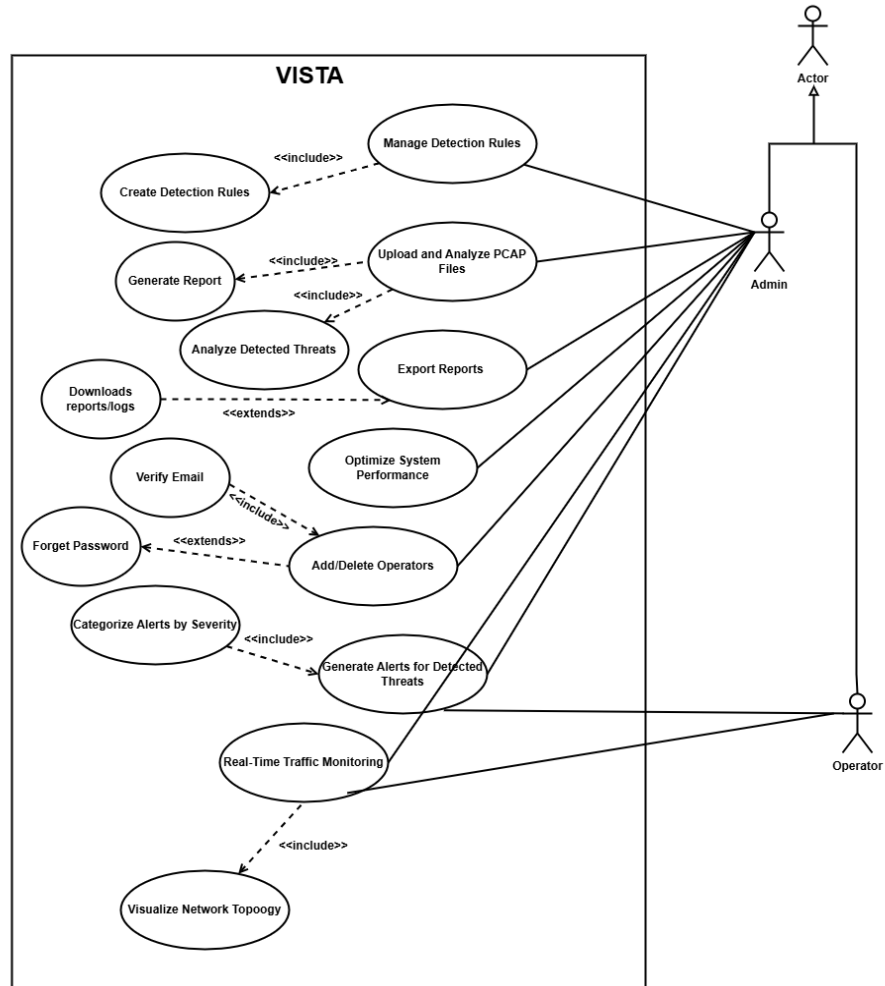


Figure 23 Use Case Diagram

7.3.2. High Level Use Case Description

1. Use case: Manage Detection Rules

Actor: Admin

Description: The Admin creates, edits, and deletes detection rules that help the system identify potential security threats. These rules form the core of the

system's threat detection capabilities, defining patterns and behaviors that trigger alerts.

2. Use case: Upload and Analyze PCAP Files:

Actor: Admin

Description: The Admin uploads Packet Capture (PCAP) files for the system to analyze network traffic and detect anomalies or threats. The analysis provides insights into possible security incidents.

3. Use case: Generate Report:

Actor: Admin

Description: The Admin generates comprehensive reports summarizing the system's analyses, including details on detected threats, system performance, and overall security status.

4. Use case: Analyze Detected Threats:

Actor: Admin

Description: The Admin reviews detailed analyses of detected threats to understand their nature, severity, and potential impact. This helps in formulating appropriate response strategies.

5. Use case: Export Reports:

Actor: Admin

Description: The Admin exports reports in various formats (e.g., PDF, CSV) for external review, documentation, or sharing with other stakeholders.

6. Use case: Downloads Reports/Logs:

Actor: Admin, Operator

Description: Admin or Operator downloads system-generated reports and logs for offline analysis, audits, or backup purposes.

7. Use case: Add/Delete Operators:

Actor: Admin

Description: The Admin manages user accounts by adding new Operators or deleting existing ones, controlling who can access and interact with the system.

8. Use case: Generate Alerts for Detected Threats:

Actor: System

Description: The system generates alerts for detected threats, notifying Admin and Operators of potential security incidents that require immediate attention.

9. Use case: Real-Time Traffic Monitoring:

Actor: Admin, Operator

Description: Admin and Operators monitor network traffic in real-time, gaining immediate insights into the system's security status and detecting any unusual activity.

10. Use case: Visualize Network Topology:

Actor: System

Description: The system provides a visual representation of the network topology, aiding Admin and Operators in understanding the network structure and identifying potential vulnerabilities.

7.4. Collaboration Diagram

Colaboration diagram of Register User

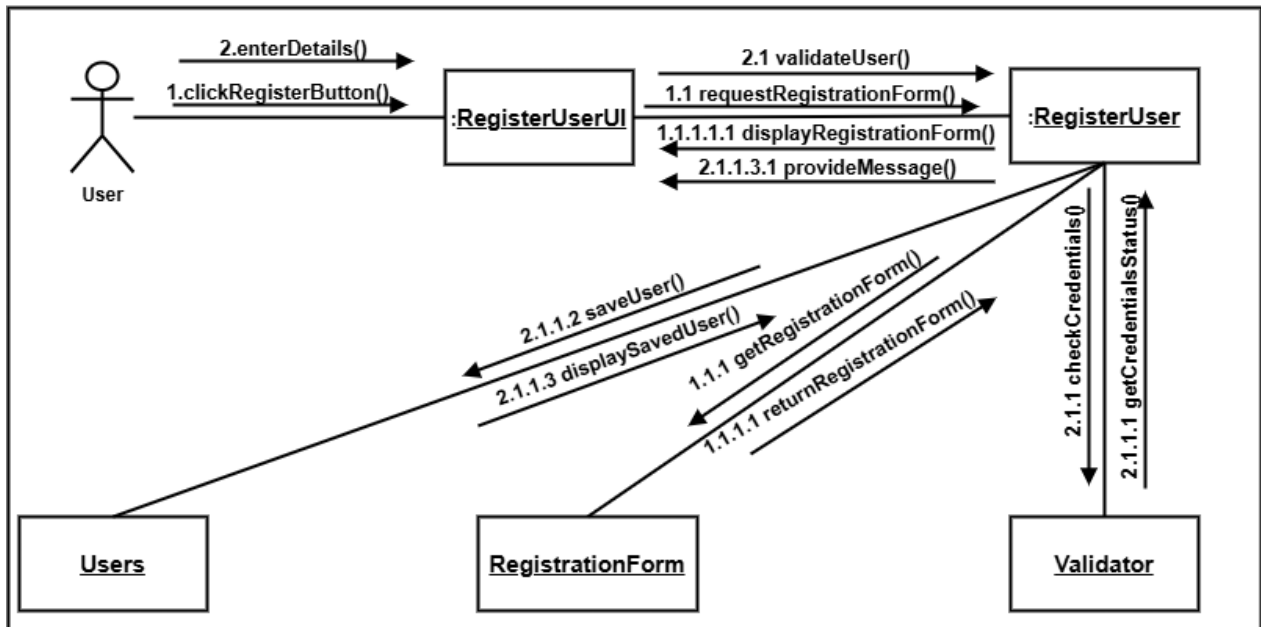


Figure 24 Collaboration Diagram od Register User

Colaboration diagram of Login User

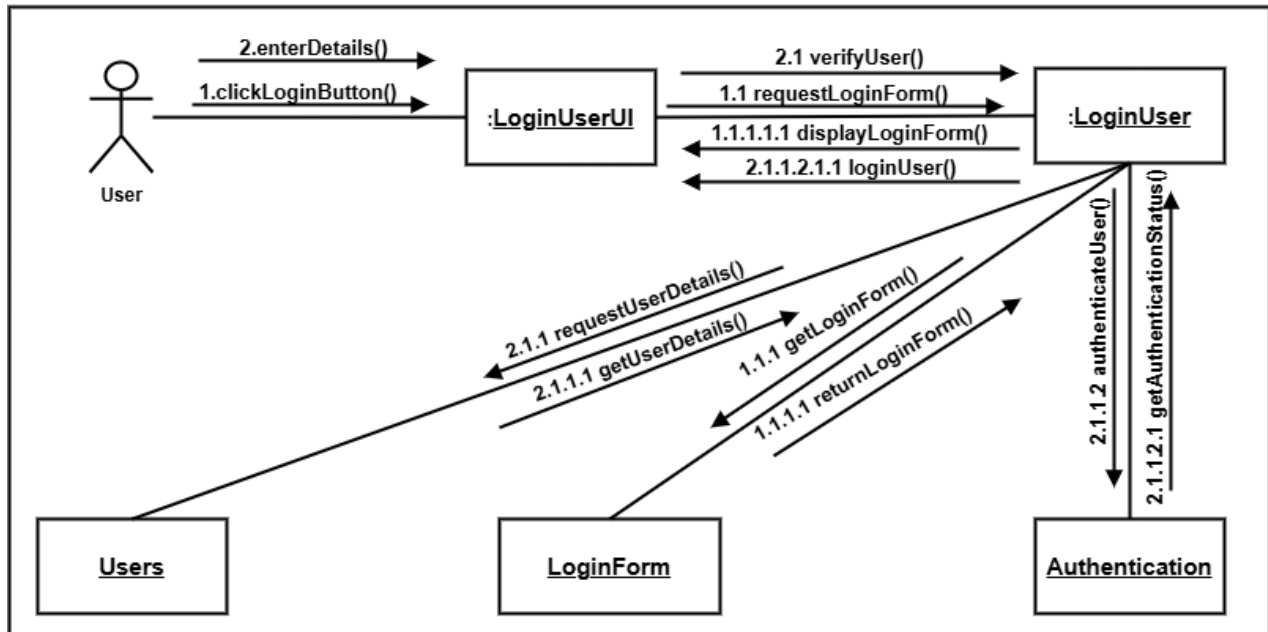


Figure 25 Collaboration Diagram of Login User

Collaboration Diagram for Realtime Packet Capture

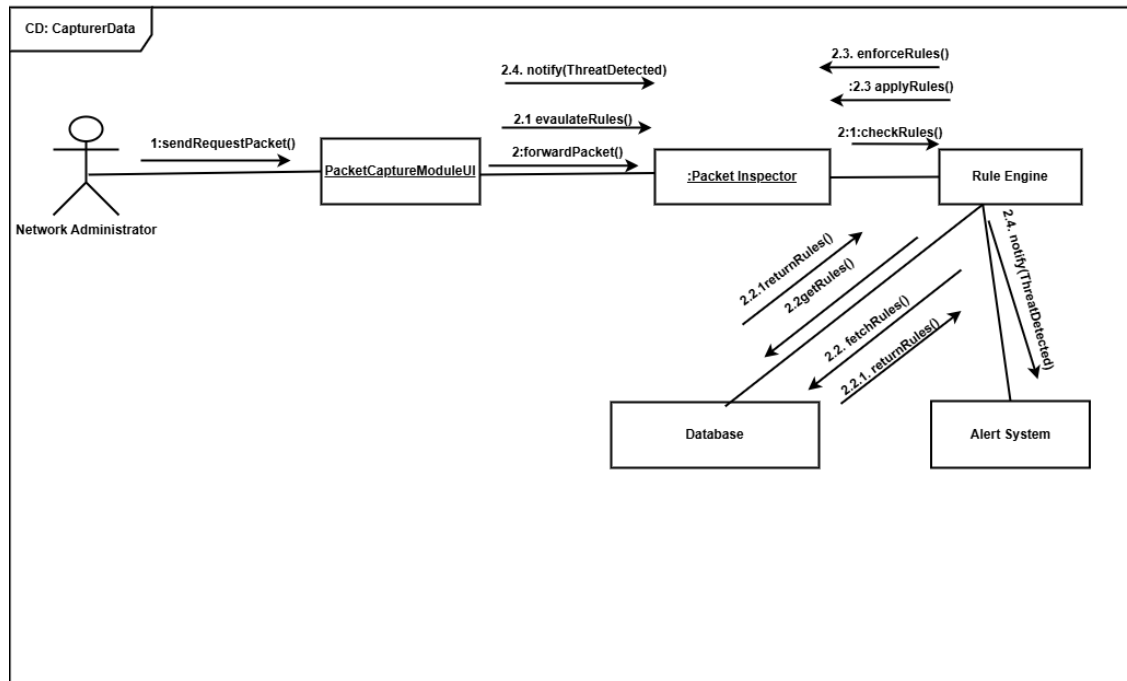


Figure 26 Collaboration Diagram of Realtime Packet Analysis

Collaboration Diagram for PCAP File Analyzer

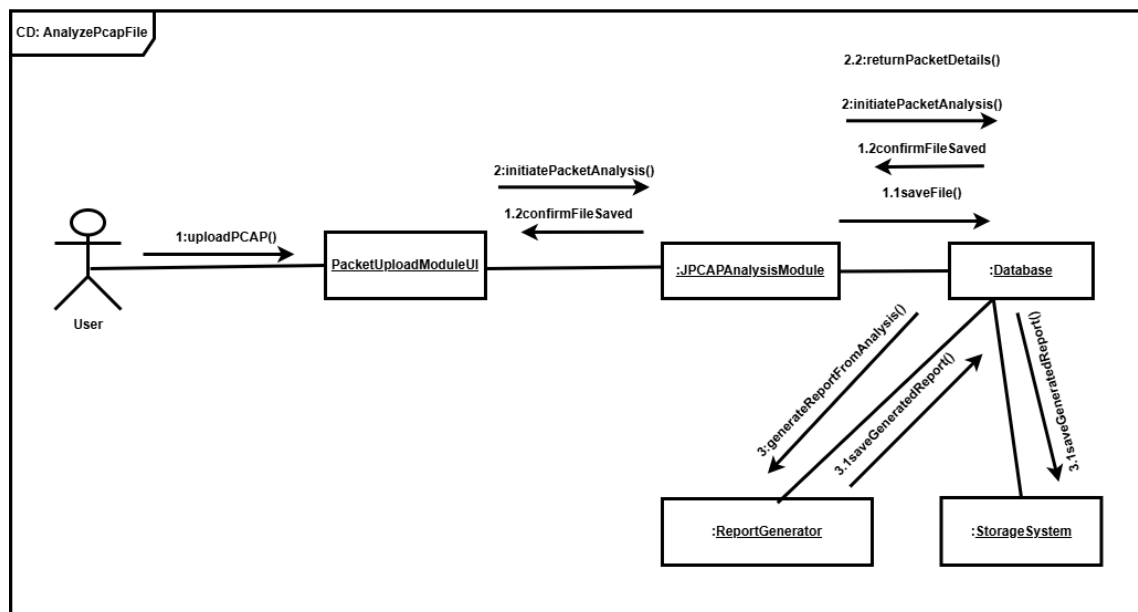


Figure 27 Collaboration Diagram of PCAP Analyzer

7.5. Sequence Diagram

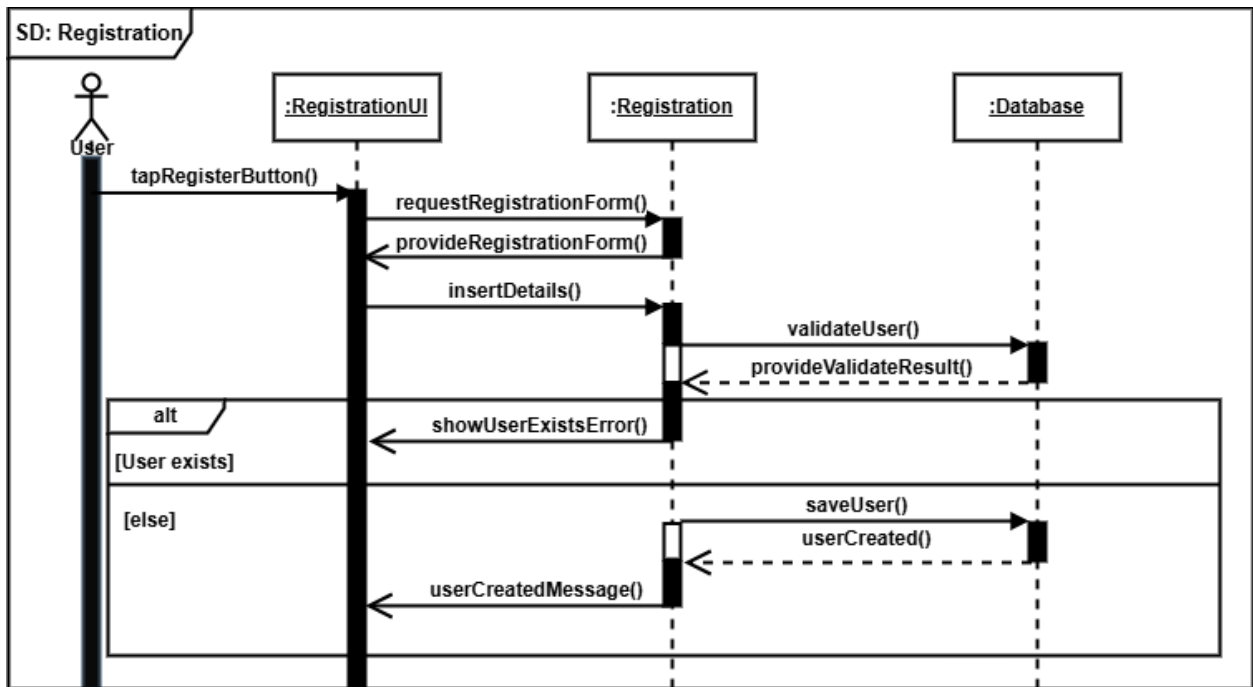


Figure 28 Sequence Diagram of User Registration

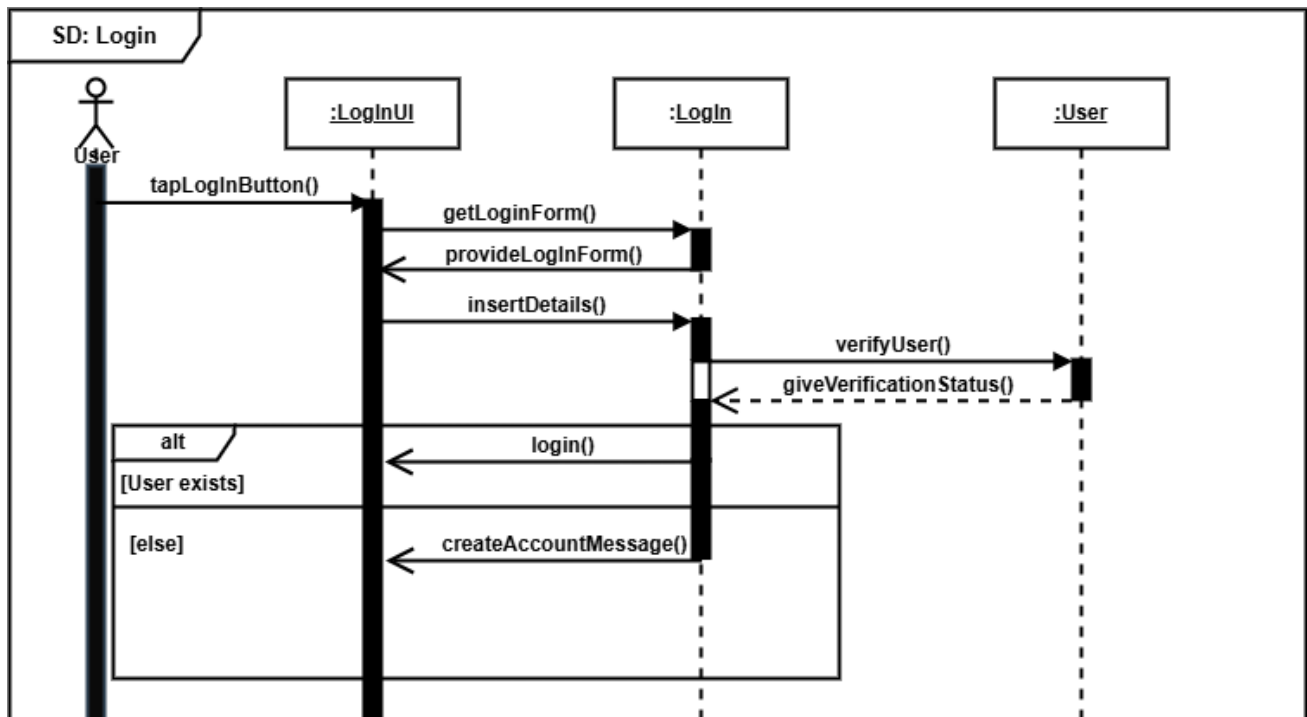


Figure 29 Sequence Diagram of User Login

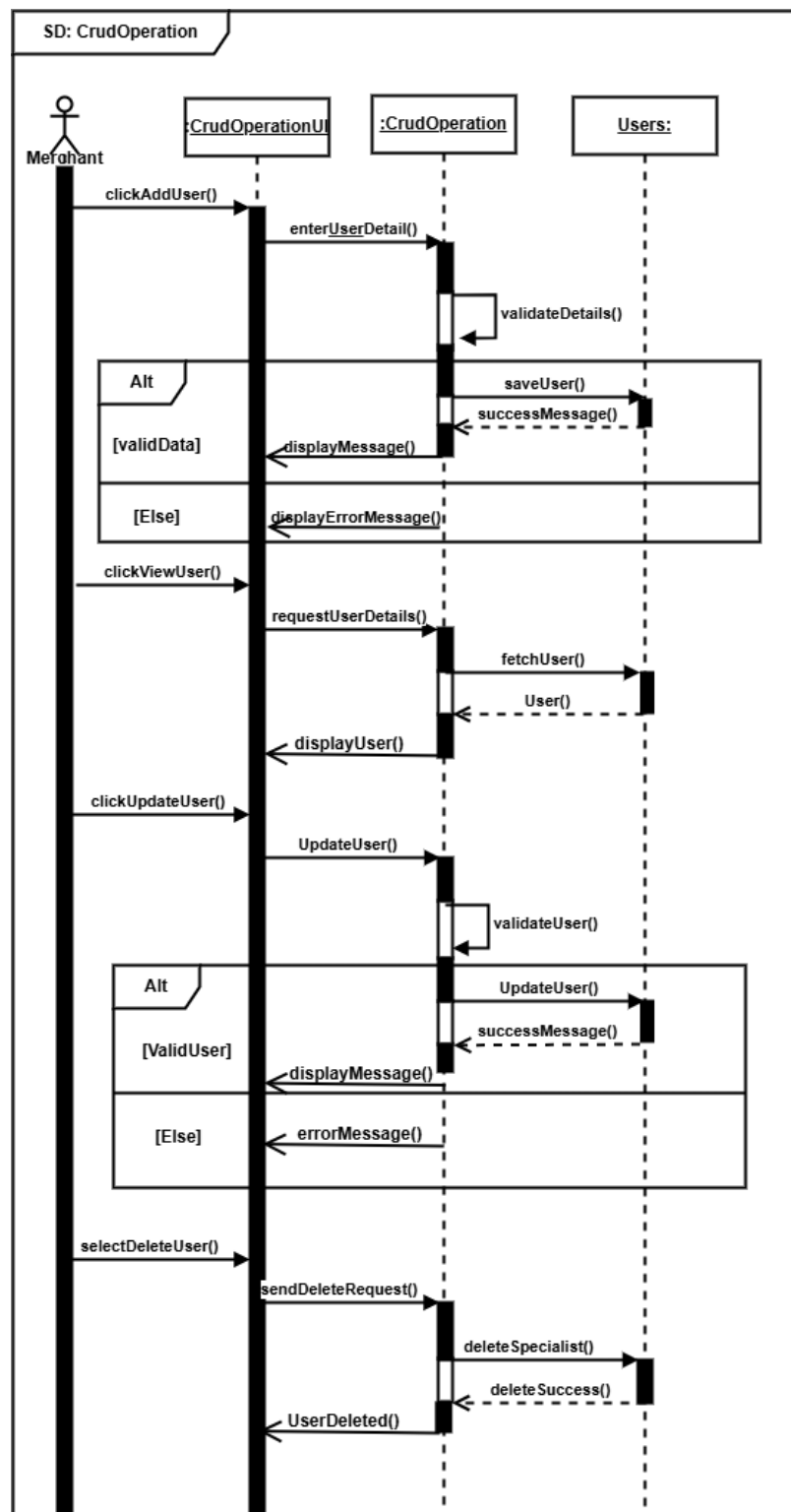


Figure 30 Sequence Diagram of Operator CRUD operation

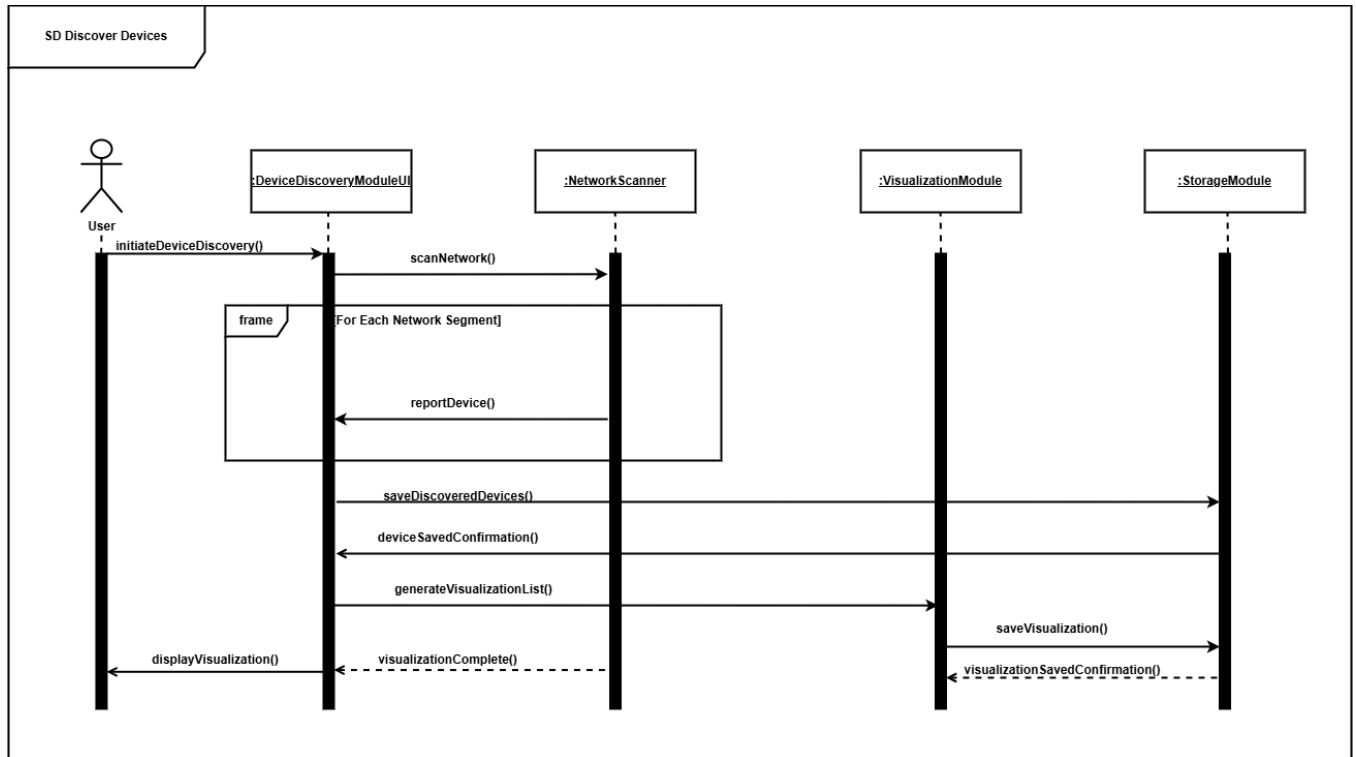


Figure 31 Sequence Diagram of Discover Devices

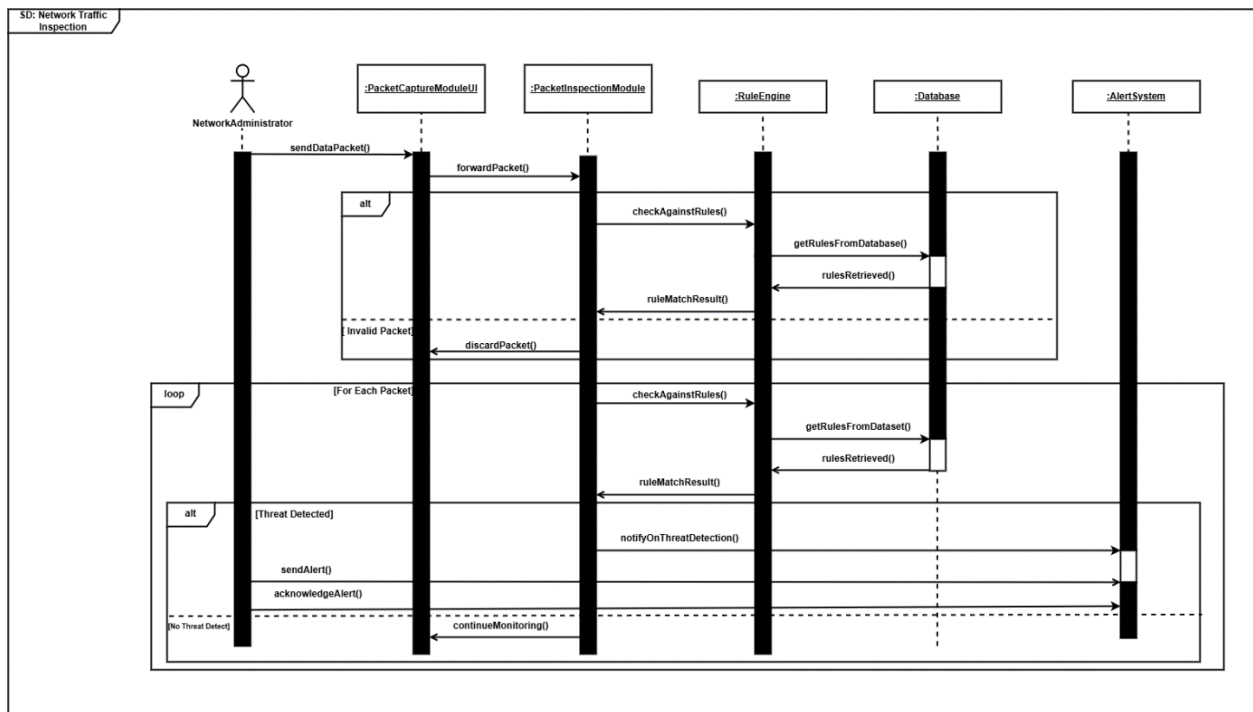


Figure 32 Sequence Diagram of Network Traffic analysis

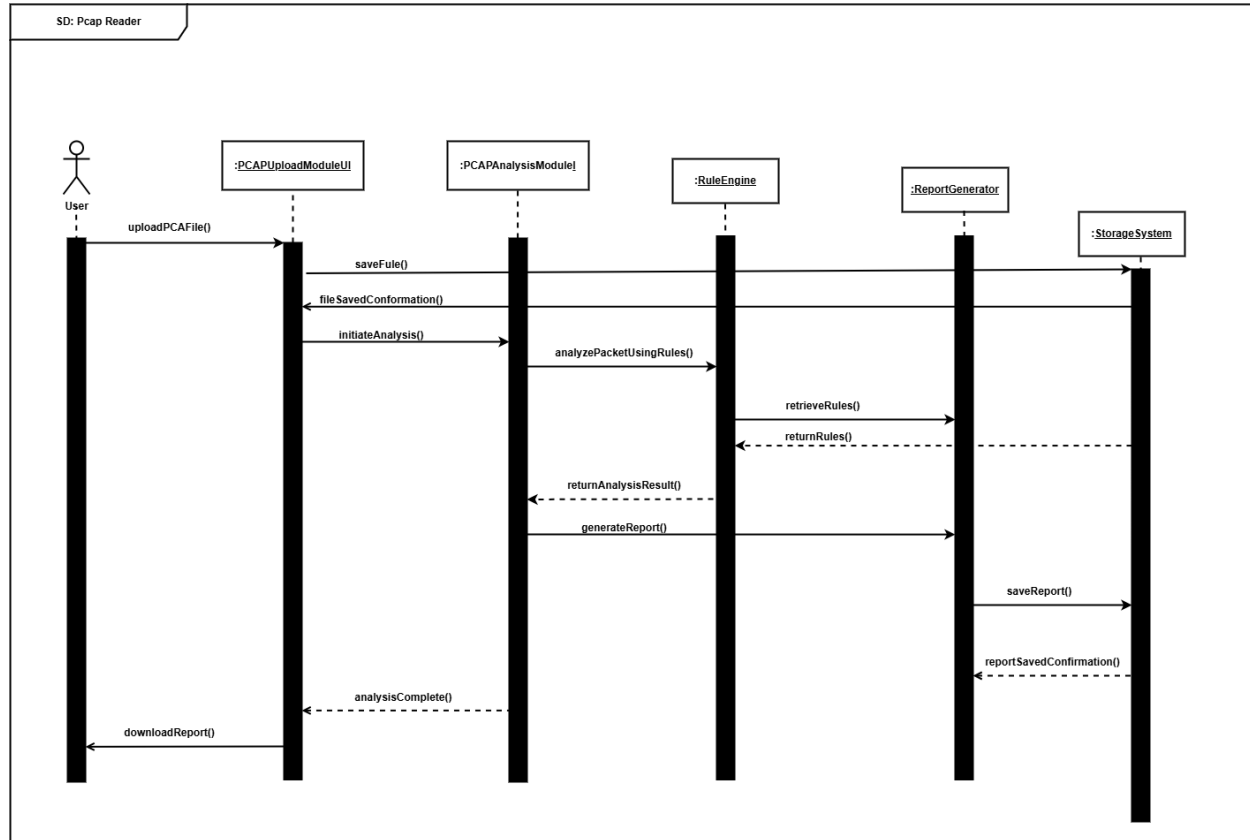


Figure 33 Sequence Diagram of PCAP Reader

7.6. Final ERD

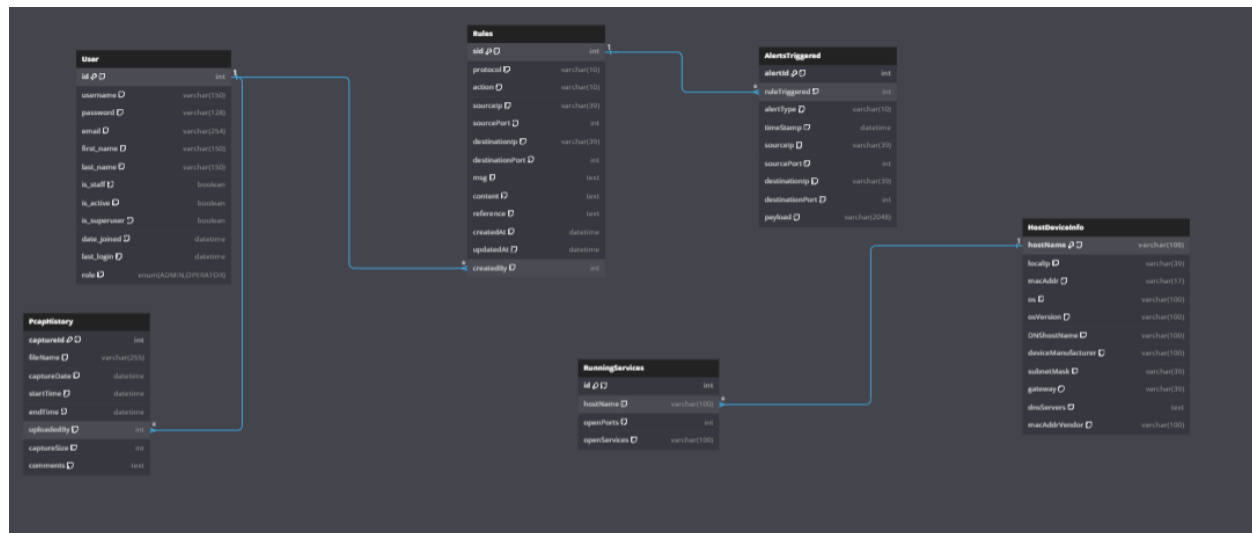


Figure 34 Final ERD

7.6.1. Data Dictionary

Table: User

Table 7 Data Dictionary f User Table

Column Name	Data Type	Notes
id	int	Primary key for the user
username	varchar(150)	Username of the user
password	varchar(128)	Password hash
email	varchar(254)	Email address
first_name	varchar(150)	First name of the user
last_name	varchar(150)	Last name of the user
is_staff	boolean	True if the user is a staff member
is_active	boolean	True if the user account is active
is_superuser	boolean	True if the user is a superuser
date_joined	datetime	Timestamp when the user joined
last_login	datetime	Timestamp of the last login
role	enum('ADMIN', 'OPERATOR')	Role of the user. Can only be 'ADMIN' or 'OPERATOR'

Table: Rules

Table 8 Data Dictionary of Rule Table

Column Name	Data Type	Notes
sid	int	Snort ID for the rule
protocol	varchar(10)	Protocol (e.g., tcp, udp, icmp)

action	varchar(10)	Choices: alert, log
sourceIp	varchar(39)	Source IP address (IPv4/IPv6)
sourcePort	int	Range: 0-65535
destinationIp	varchar(39)	Destination IP address (IPv4/IPv6)
destinationPort	int	Range: 0-65535
msg	text	Message or description of the rule
content	text	Content to match
reference	text	Additional references
createdAt	datetime	Timestamp for creation
updatedAt	datetime	Timestamp for last update
createdBy	int	User who created the rule

Table: AlertsTriggered**Table 9 Data Dictionary of AlertsTriggered Table**

Column Name	Data Type	Notes
alertId	int	Primary key for the alert
ruleTriggered	int	Rule that triggered the alert
alertType	varchar(10)	Type of alert (alert or log)
timeStamp	datetime	Timestamp when the alert was triggered
sourceIp	varchar(39)	Source IP address of the alert
sourcePort	int	Range: 0-65535

destinationIp	varchar(39)	Destination IP address of the alert
destinationPort	int	Range: 0-65535
payload	varchar(2048)	Optional payload data

Table: HostDeviceInfo**Table 10 Data Dictionary of HostDeviceInfo table**

Column Name	Data Type	Notes
hostName	varchar(100)	Primary key - Hostname of the device
localIp	varchar(39)	Local IP address of the device
macAddr	varchar(17)	MAC Address of the device
os	varchar(100)	Operating System of the device
osVersion	varchar(100)	Version of the OS
DNSHostName	varchar(100)	DNS Hostname of the device
deviceManufacturer	varchar(100)	Device manufacturer
subnetMask	varchar(39)	Subnet mask
gateway	varchar(39)	Default gateway
dnsServers	text	List of DNS servers (comma-separated)
macAddrVendor	varchar(100)	Vendor of the MAC Address

Table: RunningServices

Table 11 Data Dictionary of RunningServices table

Column Name	Data Type	Notes
id	int	Primary key for running services
hostName	varchar(100)	Foreign key to HostDeviceInfo
openPorts	int	Number of open ports
openServices	varchar(100)	List of open services

Table: PcapHistory**Table 12 Data Dictionary of PCAPHistory Table**

Column Name	Data Type	Notes
captureId	int	Primary key for the capture
fileName	varchar(255)	Name of the PCAP file
captureDate	datetime	Date and time when the capture was made
startTime	datetime	Start time of the capture
endTime	datetime	End time of the capture
uploadedBy	int	User who uploaded the PCAP file
captureSize	int	Size of the PCAP file in bytes
comments	text	Optional comments or description of the capture

7.7. Survey Result

7.7.1 Post Survey Key Findings

Question number 1:

In this chart, 35 respondents are displayed according to their familiarity with cybersecurity concepts and blue teaming practices.

42. Of those, 9 per cent said they were Slightly Familiar, referring to knowledge of the technology without hands on experience.

34. Moderately Familiar (3%) understand key concepts with some practical experience.

In fact, 20% of respondents are Not Familiar, meaning they know essentially nothing about cybersecurity.

There is no significant representation of a small portion (Highly Familiar and Expert) marked by the absence of green and purple sections. This indicates that most respondents have some basic to moderate knowledge of, or experience in, cybersecurity, and a smaller number have advanced experience or knowledge of the subject.

Question 2

The two largest groups, at 31.4%, are those who are 'Moderately Familiar' and 'Highly Familiar' with NIDS. It indicates that a majority of the respondents have at least a basic understanding of these systems. About 22.9% of respondents confessed, "Not Familiar" to NIDS, which suggests that people do not know about this vital security concept. It does not show us the percentage of "Experts," but if we subtract all the other percentages from 100%, it'll tell us. Here $100 - (22.9 + 14.3 + 31.4 + 31.4) = 0$. That is what makes the Expert category have no respondents.

All in all, the survey results reflect a certain degree of understanding related to NIDS among the respondents in terms of familiarity ranging from moderate to strong familiarity for some portion of the respondents but also a large number lacking even basic knowledge.

Question 3

The survey answered questions regarding the level of importance for modern cybersecurity of Network Intrusion Detection Systems (NIDS), and this pie chart shows their answers. Almost two thirds (68.6%) of respondents think that NIDS are 'Very Important.' A smaller section (28.6%) classifies them as "Optional (Can be implemented

depending on demand)”, a very small fraction (about 2.8%) counts as ‘Not Important’. This is obviously more than most respondents know that plays a central role in the cybersecurity strategies of today.

Question 4

This pie chart shows the results of a survey asking about the importance of Network Intrusion Detection Systems (NIDS) in modern cybersecurity. The results from 35 responses are categorized as follows:

- Very Important: 68.6% of respondents believe NIDS are very important.
- Optional (Can be implemented based on demand): 28.6% of respondents consider NIDS optional.
- Not Important: A small minority (approximately 2.8%) believe NIDS are not important.

The chart clearly indicates that the overwhelming majority of respondents recognize the significant role NIDS play in modern cybersecurity.

Question5

Question6

This pie chart presents the results of a survey asking respondents to compare the effectiveness of signature-based detection systems versus anomaly-based systems. Here's a breakdown:

- Highly Effective (42.9%): This is the largest group, indicating that a significant portion of respondents believe signature-based systems are more reliable and accurate.
- Somewhat Effective (37.1%): This group finds signature-based systems effective for known threats but acknowledges their limitations in detecting new or unknown attacks.
- Equally Effective (20%): This group believes both systems have their strengths and are effective in their respective domains.

- Less Effective (No percentage explicitly shown, but it's the remaining portion): This group finds signature-based systems less effective due to their inability to detect zero-day exploits and variations of known attacks.
- Not Effective (No percentage explicitly shown): This group considers anomaly-based systems far superior.

Key takeaway: While a plurality believes signature-based systems are highly effective, a substantial portion acknowledges their limitations, with a significant combined percentage viewing them as only somewhat effective or less effective. This highlights the ongoing debate and the need for a multi-layered approach to intrusion detection, often combining both signature-based and anomaly-based methods.

Question7

Real-time monitoring is the most critical feature of a NIDS, followed by alerting and reporting, customizable rules, and deep packet inspection. Features related to historical analysis, visualization, and scalability are considered less critical but still important for a comprehensive NIDS solution.

Question8

The survey clearly shows that the majority of respondents place a high value on the ability to customize detection rules for their NIDS. This highlights the importance of flexibility and adaptability in modern intrusion detection systems.

Question9

Results from a survey regarding the usefulness of a web based admin dashboard to control NIDS are presented in the form of this pie chart. Thirteen out of 20 found it "Very useful," adding they believed it would simplify management and monitoring (65.7%). Just a smaller portion (22.9%) thought it was 'Somewhat useful,' meaning 'It helps, but it's not vital.'

Question10

While there's some debate, the majority of respondents prioritize detecting known threats, either exclusively or with a contextual preference. This suggests that while zero-day

attack detection is important, most organizations focus on mitigating the more common and easily detectable threats first.

Question11

Most people think that analyzing of captured network traffic (PCAP files) is very important to locate threats that may have slipped through their defenses. They use these types of analysis to learn about past security events and increase their overall security position.

Question12

Overwhelmingly, respondents agree that the functionality of a NIDS would be improved by visualizing network topology. Therefore, if this feature is facilitated in NIDS dashboards, it will be highly accepted by users and will therefore assist them to understand and manage network security.

Question13

The responses of the majority identify scalability as extremely or at least important, indicating that a majority of organizations put a lot of weight on scaling when choosing a network monitoring solution. It is made clear that as network traffic increases so does this necessitates NIDS solutions to cope.

Question14

In fact, the majority of respondents (combining "Strongly prefer" and "Prefer") clearly supports web-based interface for NIDS management. While it may sound like the obvious and commonplace solution, this is telling of the importance of user friendly interfaces for broad availability as well as ease of management, even in the more technical territories of network security.

Question15

The majority of respondents (combined "Extremely intuitive" and "Somewhat intuitive") believe that a NIDS interface should be at least somewhat intuitive for non-technical users. This emphasizes the importance of user-friendly design and accessibility in NIDS, even for users without deep technical expertise.

Question16

A NIDS dashboard should provide the overwhelming majority of the users (60% of them to be precise) the ability to navigate traffic statistics, detailed reports, and visual graphs. Data visualization and thorough reporting plays an important role in effective network security monitoring and analysis, which this demonstrates.

7.7.2. Visual Representation of Survey Result

How would you rate your familiarity with cybersecurity concepts and blue teaming practices (e.g., threat detection, incident response, and network defense)?

35 responses

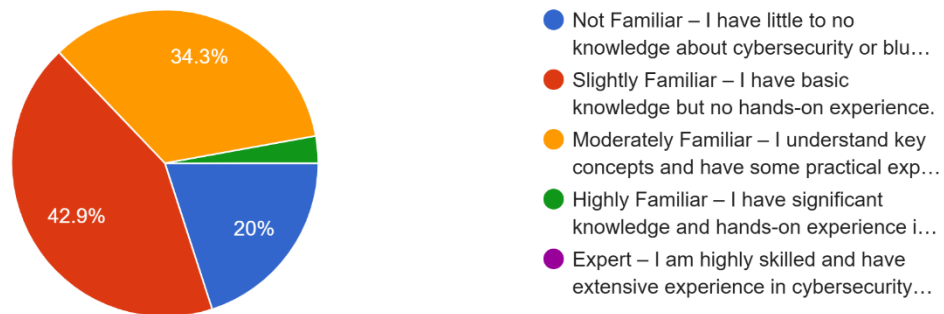


Figure 35 Survey Question 1

Are you familiar with the concept of Network Intrusion Detection Systems (NIDS)?

35 responses

[Copy chart](#)

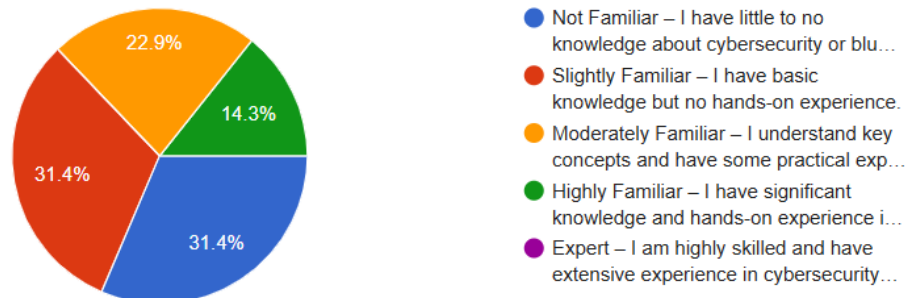


Figure 36 Survey Question 2

How important do you think NIDS are for modern cybersecurity?

 [Copy chart](#)

35 responses

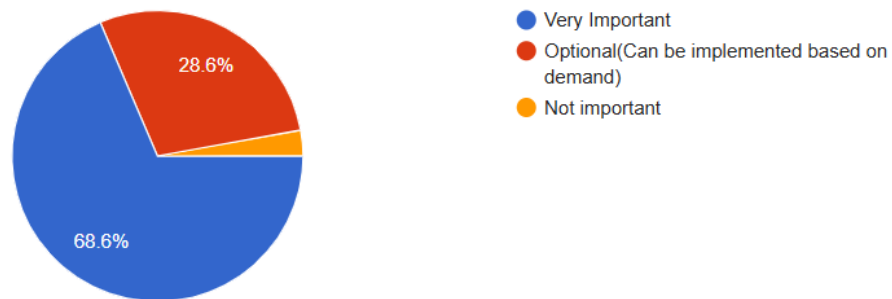


Figure 37 Survey Question 3

How important do you think NIDS are for modern cybersecurity?

 [Copy chart](#)

35 responses

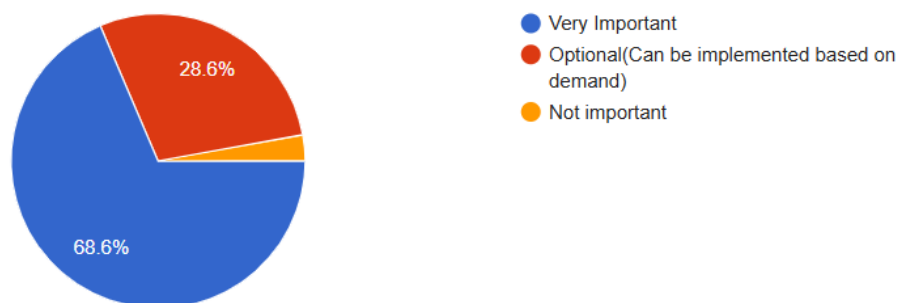


Figure 38 Survey Question 4

Have you used or implemented a rule-based NIDS like Snort before? If yes, please share your experience.

25 responses

Nop

I have experimented with Snort in a lab environment. I implemented basic rules for monitoring traffic and detecting common threats like port scans and brute force attempts. It was a great introduction to understanding how signature-based intrusion detection works.

no i have not implemented any

No I haven't, but I would love to.

I haven't used it in real life.

I have not used any rule-based NIDS

Good

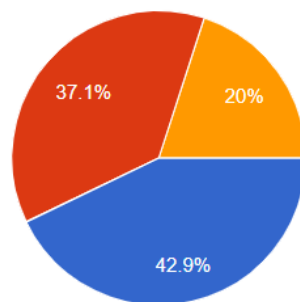
No

Figure 39 Survey Question 5

How effective do you find signature-based detection systems compared to anomaly-based systems?

 Copy chart

35 responses



- Highly Effective – Signature-based systems are more reliable and accurat...
- Somewhat Effective – Signature-based systems work well for known threats b...
- Equally Effective – Both systems are effective in their respective domains a...
- Less Effective – Signature-based systems are less effective due to their...
- Not Effective – I find anomaly-based systems far superior in identifying and...

Figure 40 Survey Question 6

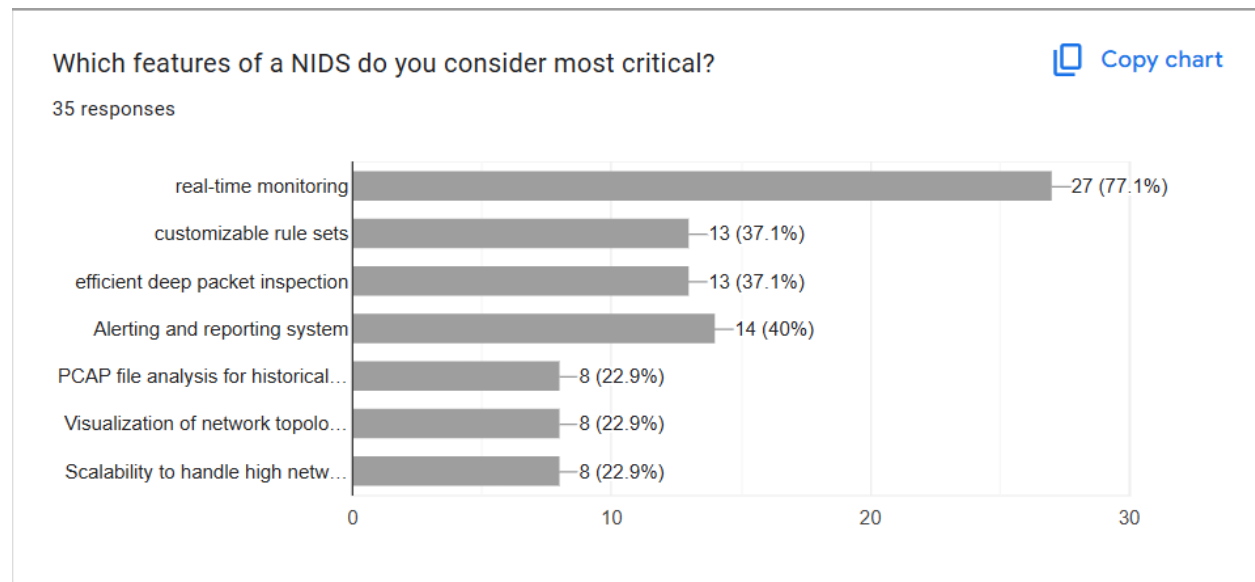


Figure 41 Survey Question 7

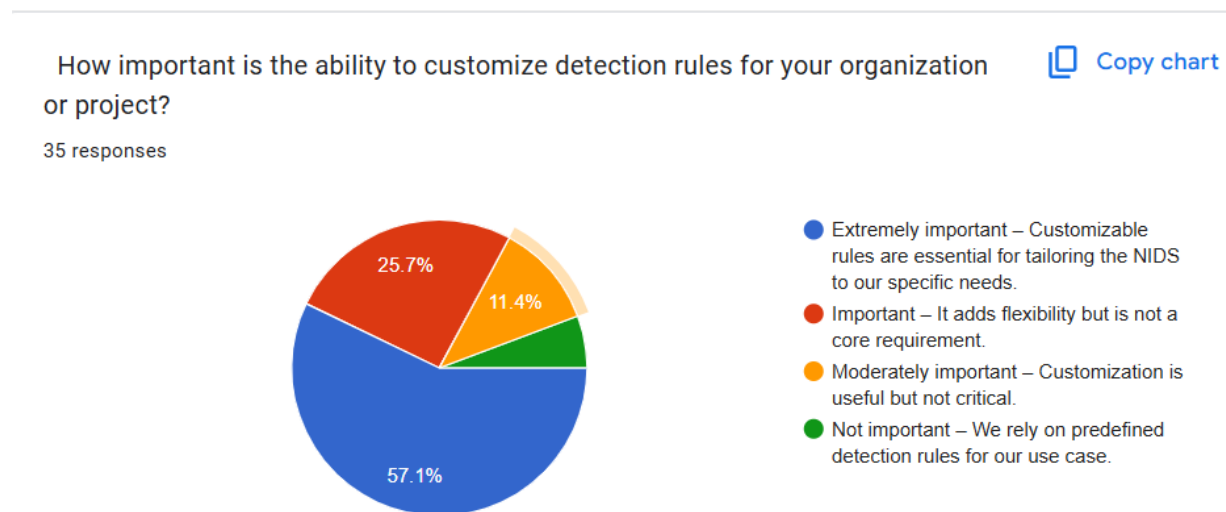


Figure 42 Survey Question 8

How useful would it be to have a web-based admin dashboard for managing NIDS?

 [Copy chart](#)

35 responses

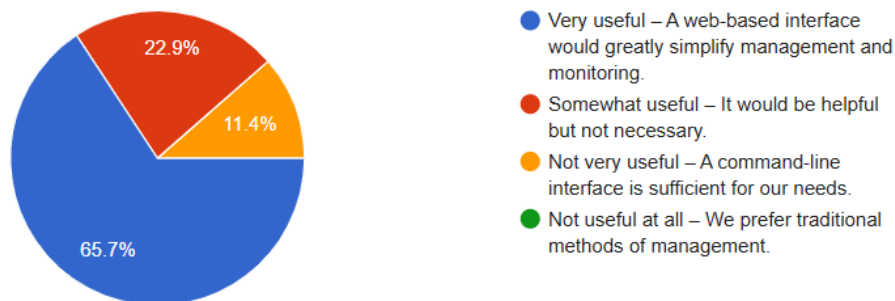


Figure 43 Survey Question 9

Would you prioritize detecting known threats over the ability to identify zero-day attacks?

 [Copy chart](#)

35 responses

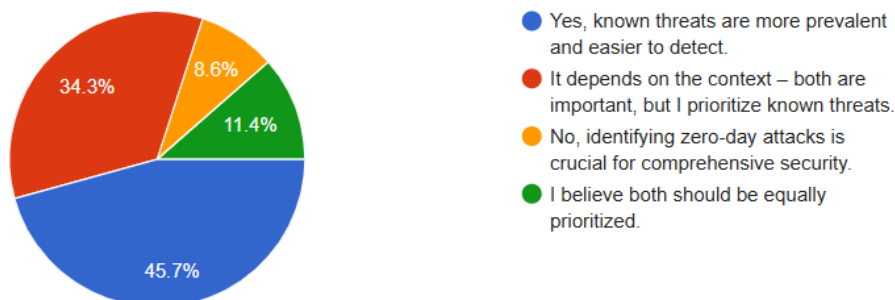


Figure 44 Survey Question 10

How significant is PCAP file analysis for retrospective threat detection in your use case?

[Copy chart](#)

35 responses

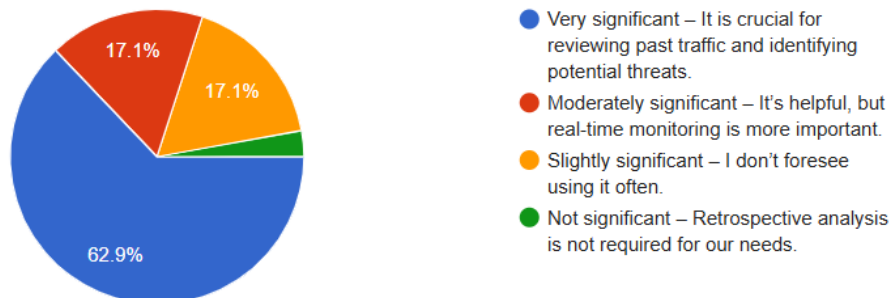


Figure 45 Survey Question 11

Do you think a visualization of network topology would enhance the usability of a NIDS?

[Copy chart](#)

35 responses

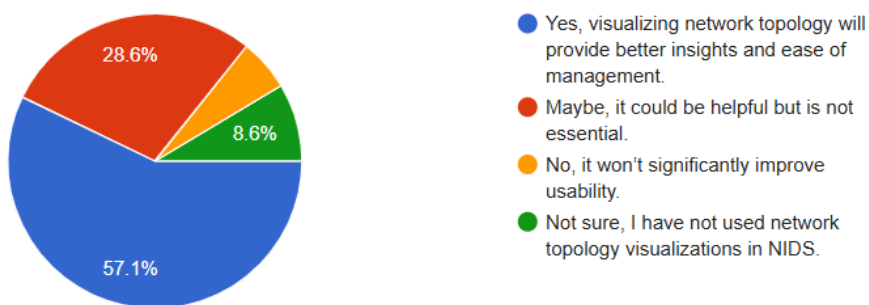


Figure 46 Survey Question 12

How important is scalability for handling high network traffic in your environment?

35 responses

 [Copy chart](#)

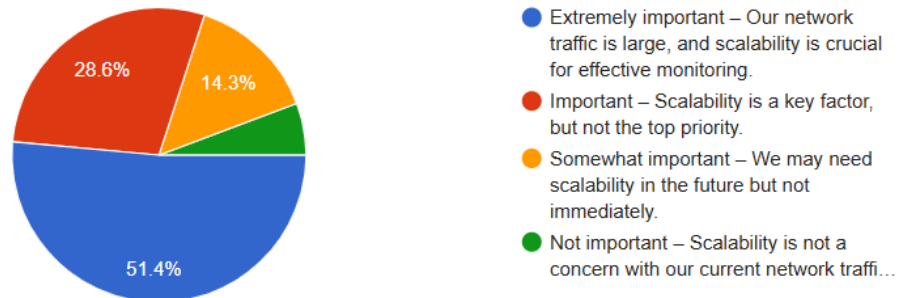


Figure 47 Survey Question 13

Do you prefer a web-based interface over a command-line interface for managing NIDS?

35 responses

 [Copy chart](#)

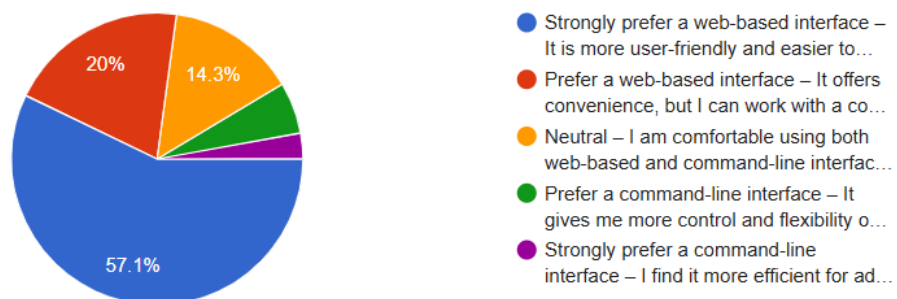


Figure 48 Survey Question 14

How intuitive should the user interface of a NIDS be for non-technical users?

 [Copy chart](#)

35 responses

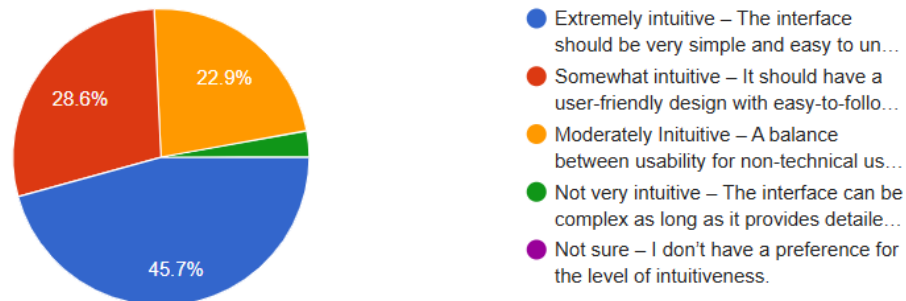


Figure 49 Survey Question 15

Would you value features like traffic statistics, detailed reports, and visual graphs in the dashboard?

 [Copy chart](#)

35 responses

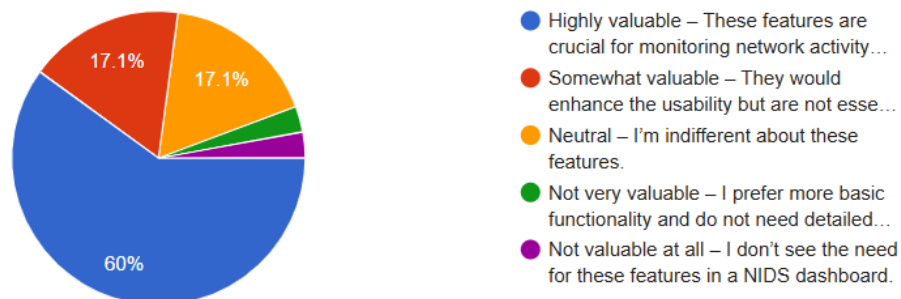


Figure 50 Survey Question 16

