Contents

List of Figures

List of Tables

DEDICATION

First and foremost,
I would like to express my deepest gratitude to Allah, who has allowed me to have the knowledge and pursue this path. All praise and thanks to Allah—Alhamdulillah—for granting me this opportunity to acquire such valuable experience through out my career.

To my dear mother Latifa,
who has always believed in me and supported me unconditionally, who always was there for me. No words can describe my love and gratitude.

To my father Habib,
It's been a challenging journey without you, my father. I deeply wish you were here today to witness my graduation

To my brothers,
Especially Mohamed, whose constant encouragement and technical companionship have been invaluable. And also Malek and Bilel, who have provided me with all kind of support throughout my journey.

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude and appreciation to the following individuals and organizations who have played a crucial role in the completion of this project:

First and foremost, I am greatly indebted to **Mr. Nejib Khalfaoui**, Network Security Professor at the Higher Institute of Computer Science. I express my sincere thanks for supervising the work, and providing expert advice throughout the whole process.

I would also like to express my deep gratitude to **Mr. Mohamed Jelalia**, CEO and founder of PixelJ, for his consistent support and encouragement throughout this project.His trust and opportunities to work on meaningful projects have enriched my experience and contributed significantly to my knowledge in the field.

Abstract

This project aims to develop a user-friendly network monitoring and intrusion detection system that enables users to inspect and monitor hosts connected to their network, analyze their connections to the internet, and evaluate internet protocol metrics. The system focuses on detecting and preventing intrusions, with a specific emphasis on mitigating Man-in-the-Middle (MITM) attacks through Address Resolution Protocol (ARP) spoofing. By leveraging advanced monitoring techniques and machine learning algorithms, the system can detect suspicious network activities and provide real-time alerts to administrators. The project aims to enhance network security, empower administrators with insights into network traffic, and enable timely responses to potential threats.

General Introduction

In our modern era, where our personal information is stored online, cybersecurity is not just a choice; it is a necessity. The rise of malicious activities and cyber threats has made cybersecurity one of the fastest-growing industries. Protecting our data from breaches and attacks has become a top priority.

Cyber threats encompass a wide range of malicious activities, such as unauthorized access, identity theft, malware infections, and much more. These threats can have severe consequences, including financial loss, reputational damage, and compromised personal information. With the increasing frequency and sophistication of cyber attacks, the need for robust detection mechanisms has become vital for organization, business and even individuals.

Detecting cyber threats involves carefully observing and analyzing activities happening on computer networks, systems, and user actions. Keeping an eye out for anything strange or suspicious that could suggest a possible attack. By noticing these signs early on, we can act swiftly to prevent or reduce the harm caused by cyber incidents.
While traditional methods like static signature detection have been widely used to identify known threats, they have limitations. Cybercriminals are constantly evolving their tactics, employing new techniques to bypass static signature-based defenses. As a result, the cybersecurity community has turned to advanced technologies like machine learning and artificial intelligence (AI) for more effective threat detection.

Machine learning algorithms can analyze vast amounts of data and identify patterns that indicate anomalous behavior. By learning from historical data, these algorithms can adapt and improve their detection capabilities over time. This enables them to identify emerging threats and previously unseen attack patterns, providing a proactive defense against cyber attacks. The integration of machine learning and AI in anomaly detection has revolutionized cybersecurity. These technologies offer the ability to detect unknown threats, zero-day attacks, and sophisticated hacking attempts. They enable security professionals to stay one step ahead of cybercriminals and respond swiftly to emerging threats.

Chapter 1

Project context presentation

I Introduction

In this chapter, we will provide an overview of the host company for this project, including their background, services, and areas of expertise. We will then delve into an analysis of the current state of their network infrastructure, highlighting any shortcomings or vulnerabilities that may exist. Based on these findings, we will discuss the critical aspects that need to be addressed to enhance the company's network security. Finally, we will propose effective and tailored solutions to mitigate the identified issues and improve the overall security posture of the organization.

II Host organization

Figure 1.1: PixelJ logo

PixelJ is an independent, global code creative agency where unique people come together to do the best work of their lives. Our goal is to make work that influences culture and

builds business value. We partner with ambitious startups and global brands striving to make authentic connections. At PixelJ, we believe in collaboration, communication, and trusting each other to create great things.

II.1 Journey

Founded in 2005 in Tunis, PixelJ has continuously inspired the development and multimedia ecosystem. Over two decades, we have always evolved with time, market development, and societal changes. What has remained constant is our focus on building people. We have distinguished ourselves through innovation, rigor, open-mindedness, and adaptability. Through our working methodologies and collaborations with renowned international brands, we have made a name for ourselves in the industry.

II.2 Expertise

At PixelJ, our mission is to push the boundaries of creativity and innovation to deliver exceptional experiences. We take pride in working with iconic brands such as Ford, Nintendo, LG, Prada, and many others. Our talented team consists of passionate developers, designers, and creatives who constantly seek new ideas and technologies to create tailored solutions that meet our clients' specific needs.

II.3 Customer-centric approach

We believe that each project is an opportunity to push the limits of innovation. Whether it's creating interactive mobile applications, web applications, immersive virtual reality experiences, or supporting startups in their projects, we are here to turn your ideas into reality. Our customer-centric approach and attention to detail enable us to deliver outstanding results. We work closely with our clients to understand their goals, values, and brand identity, in order to create solutions that uniquely represent them.

II.4 Innovation and quality

Innovation is ingrained in our DNA. We stay constantly updated with the latest trends and technological advancements to offer cutting-edge solutions to our clients. Our passion for excellence and commitment to quality have earned us the trust of renowned brands worldwide.

II.5 Next generation technology

Emerging digital technologies continue to transform businesses, and at PixelJ, we excel in harnessing Next Generation Technology to enhance business systems. Our expertise includes blockchain development, AR and VR app development, big data analytics, NFT development, and chatbot development.

II.6 Mobile app development

We understand the importance of speed and ease-of-use in today's mobile-driven world. Our team of smart app developers specializes in iOS, Android, React Native, and Flutter app development, ensuring we deliver the best mobile applications using the latest technologies.

II.7 Web development

Our software development methodology centers around innovative implementation of Agile processes. We offer expertise in prototyping, frontend and backend development, API integration, database architecture, CMS customization, and enterprise custom development.

II.8 Ecommerce & CMS

We create memorable websites that sell. Our services include CMS opensource development and customization, ERP integration, ecommerce performance optimization, and payment gateway integration.

II.9 Data analytics

Understanding user interaction and leveraging data is crucial for business success. Our data analytics services empower you with insights and opportunities for growth. We offer expertise in data engineering, AI and ML, automation, visualization and analytics.

II.10 Startup business

We provide a holistic approach to startup businesses, offering services such as technology partnership, product technology strategy and advice, clickable prototypes, proof-of-concept, influencer marketing, and app workflow simulation.

II.11 What we do

At PixelJ, we deliver world-class digital solutions that are on-time and in-budget. We believe that bleeding-edge companies drive deeper, more meaningful connections with their consumers. We help our partners realize the full potential of being a modern brand that moves at the pace of culture by exploiting the last unfair competitive advantage any brand has: craft, code, and creativity. Success depends on the solutions you can deliver in the digital world. For more than 15 years, global brands continue to choose PixelJ as their preferred custom development company. We proudly sketch, architect, and develop elegant products & solutions used by millions of people. Your data is a goldmine of growth, just waiting to be expertly mined. Our mission is how to turn data into information, and information into knowledge and knowledge into decision. We uncover hidden opportunities for business enhancement and expansion with our data analytics consulting tools and Design Thinking and Agile-led approaches. With a focus on delivering new and improved ways to satisfy your vision, our expert data engineers empower you to monitor key metrics, discover patterns, and access revelatory insights. Our robust data analytics company arms you with all you need to make better decisions.

III Project Presentation

III.1 Current State

The network infrastructure of the company consists workstations connected through a local network. The company uses a basic firewall to protect the network from external threats. The network topology is currently a simple star network, with the router acting as the center point and the workstations connected to it. However, the current network infrastructure is vulnerable to various types of cyber attacks.

III.2 Critics

- **Lack of Network Monitoring** One of the primary issues I discovered during my assessment is the absence of monitoring systems. Without monitoring, it is impossible to detect any anomalies in the network or potential security breaches. The company is unable to monitor the state of the network, connected devices, protocols in use, throughput, and other metrics puts it at a significant disadvantage and vulnerable to internal attacks.

- **MITM Threats** Man-in-the-middle (MITM) attacks are one of the most common types of network attacks that can compromise the security of a company's internal network. These attacks involve an attacker intercepting communications between two parties to steal sen-

sitive information, such as login credentials. The company is vulnerable to MITM attacks, and its network is not configured to detect or prevent these types of attacks. Indeed, one might argue that with the widespread use of encryption and secure communication protocols, MITM attacks would be ineffective. However, advanced MITM techniques can be used to redirect user traffic to unintended destinations, which can have severe consequences. For example, attackers can use Domaine name system (DNS) spoofing to pose themselves as a legitimate DNS server to modify the IP address of the website that the user intends to visit. As a result, the user may end up on a completely different website, which can be malicious and designed to steal sensitive information.

In this case, the attacker can host a malicious website with a valid SSL certificate, making it difficult for the user client browser to detect that the website is fraudulent. The website can be designed to look and feel like a legitimate website, and the attacker can then steal sensitive information such as login credentials, credit card information, and personal data.

Overall, the company's network infrastructure is vulnerable to various internal threats. The absence of monitoring and logging systems, along with the lack of security measures against MITM attacks, leaves the network open to potential security breaches. The implementation of appropriate security measures, such as network monitoring, security protocols is essential to safeguard the company's network and prevent data breaches.

III.3 Proposed Solutions

III.3.1 Open-Source Solutions

The first proposed solution is to implement open-source security solutions, which offer a cost-effective approach to network security. By leveraging the collective expertise of the open-source community, these solutions are reliable and well-established for safeguarding networks.

- **Snort:**

  A is a free open source network intrusion detection system and intrusion prevention system (NIDPS) that can monitor network traffic for any suspicious activity. Snort uses rules and signatures to identify and prevent malicious traffic from entering the network. This solution can detect various types of attacks, including port scans, malicious payloads and buffer overflows.

- **Suricata:**

  An open-source intrusion detection and prevention system (IDPS) that can detect and prevent network attacks. Suricata uses a combination of signature-based and anomaly-

based detection to identify threats. This solution can identify various types of attacks, including malware, network exploits, and DoS attacks.

- **ELK stack with Snort:**

  The ELK stack (Elasticsearch, Logstash, and Kibana) is a powerful open source tool for log management and data visualization. By integrating Snort with ELK, network administrators can monitor network traffic, detect potential threats, and visualize data in real time. Additionally, the ELK stack can be used for advanced analytics and machine learning to identify patterns and anomalies in network traffic.

III.3.2 Paid Solutions

The second proposed solution involves implementing paid security systems, which offer advanced features and comprehensive support for network security. These paid solutions provide cutting-edge technologies and robust functionalities that go beyond what open-source solutions can offer.

- **LogRhythm Siem and NDR:**

  LogRhythm offers a SIEM (Security Information and Event Management) solution that provides real-time threat detection and response capabilities. It also includes NDR (Network Detection and Response) functionality, which can detect network threats and anomalies. This solution offers a high level of customization and advanced analytics for threat hunting.

- **Pulseway:**

  Pulseway is an IT management software that allows users to monitor, manage, and troubleshoot workstations, servers, and network devices in real-time. It provides customizable alerts, built-in commands and scripts, and a mobile app that allows users to manage their IT network from anywhere. Pulseway is available for MSPs and internal IT teams, and it offers features such as automation, remote control, and patch management.

  Paid solutions typically offer more advanced features and technical support compared to open-source solutions, but they come with a higher cost. The specific solution chosen will depend on the budget, specific security needs, and technical expertise of the organization.

III.3.3 Custom Solution

- **Custom Network Monitoring and IDS:**

  The third proposed solution is to develop a custom monitoring dashboard that provides real-time visibility into the state of the network, and mobile applications for alerts and

responses. A custom solution can be tailored to the specific needs and budget of the organization. Furthermore, this allows for greater control and flexibility, and can potentially provide a more cost-effective long-term solution.

III.3.4 Adopted solution

We have considered the available options and have decided to adopt a custom development solution for our monitoring needs. This decision was based on the following factors:

- Control and ownership
  we can modify, update, and maintain the solution according to our requirements and timelines, without relying on external vendors for updates or support.
- Integration and Compatibility
  Custom solutions can be designed to seamlessly integrate with your existing network infrastructure and other security systems. We can leverage our existing technology stack and ensure compatibility with our data sources, APIs, and internal processes

III.4 Attack anatomy

III.4.1 What is ARP ?

> The Address Resolution Protocol (ARP) is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given internet layer address, typically an IPv4 address. This mapping is a critical function in the Internet protocol suite. — Wikipedia

When a device needs to send data to another device on the same network, it first checks its ARP cache to determine if it already knows the MAC address of the destination IP. If the MAC address is not found in the cache, an ARP request is broadcasted to the network, asking the device with the corresponding IP address to respond with its MAC address. The ARP response contains the MAC address, allowing the sender to create a direct link and send the data. The following is an example of an ARP cache registry in a windows system.

```
C:\Users\thinkpad>arp -a

Interface : 192.168.1.4 --- 0x5
  Adresse Internet      Adresse physique      Type
  192.168.1.1           d8-b6-b7-85-76-7e     dynamique
  192.168.1.3           e0-1f-88-9c-19-c2     dynamique
  192.168.1.6           00-56-cd-06-89-15     dynamique
  192.168.1.7           08-11-96-83-55-e0     dynamique
```

Figure 1.2: ARP cache

To better understand the protocol and how it works here is a diagram that illustrate an ARP communication

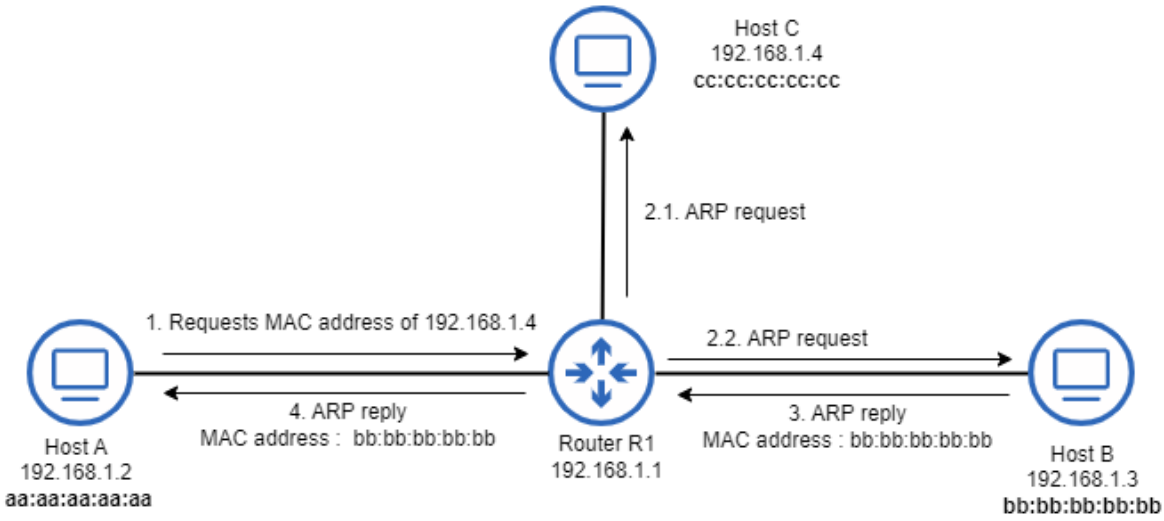


Figure 1.3: Example of ARP communication

In the figure 1.3 Host A was trying to send a packet to host B, Host A consulted his ARP registery and didn't found a MAC address associated with the IP address that the packet is intended to be sent to. so Host A initiate an ARP request to the broadcast (meaning to all the network), the packet it self will be sended to the MAC address of the router as it is the default gateway of the network, the router in his turn receives the broadcast message and transmit it to all available branches, The machine whose IP address matches with the destination ip address mentioned in the received packet will send "ARP response" back to the host-A. ARP reply containing MAC address with corresponding IP address, then the router receives the response containing the mac address of the host, and transmit it to the original requester.

III.4.2 ARP scan

One possible use of the ARP protocol is when we want to legitimately scan our local network for connected devices. the arp-scan tool works exactly like shown in the figure 1.3,

it first determines the local network subnet mask, then determines the range of ip address in the network, then starts sending a broadcast to all of the possible ip addresses, this will lead to only the connected and available devices in the local network to reply with their ip and mac addresses.

III.4.3 ARP spoofing

This ARP attack type is used to attack the network in the middle of the communication link. This is achieved with the help of fake ARP replies. The key factor in the vulnerability is that ARP protocol assumes that ARP response comes from the right terminal whose IP address matches with the one contained in ARP request message. Moreover there is no way to validate that ARP response was sent from the correct device. This design flow in the ARP protocol is being exploited by hackers. In an ARP spoofing attack, the attacker manipulates the Address Resolution Protocol (ARP) responses sent across a network. By sending spoofed or modified ARP responses, the attacker tricks computers on the network into associating a specific IP address with a different MAC address. This manipulation corrupts the ARP cache or ARP table, which is responsible for maintaining the mapping between IP addresses and MAC addresses. The following is an example of how would the ARP registry look like after the attacker has executed the attack, we can see that the MAC address ending in 55-e0 is duplicated across two different IP addresses, this is an indicator of ARP spoofing attack:

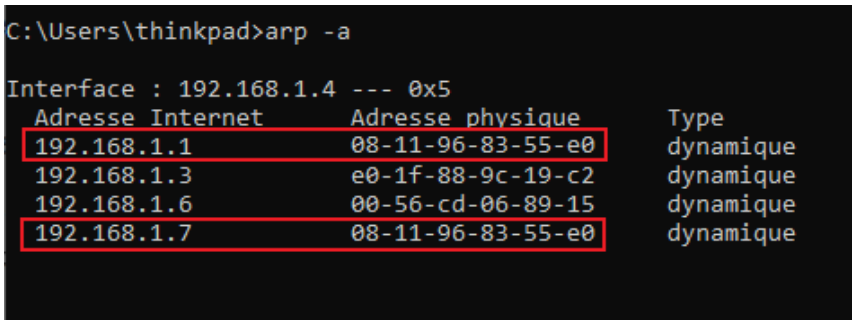


Figure 1.4: Spoofed ARP cache

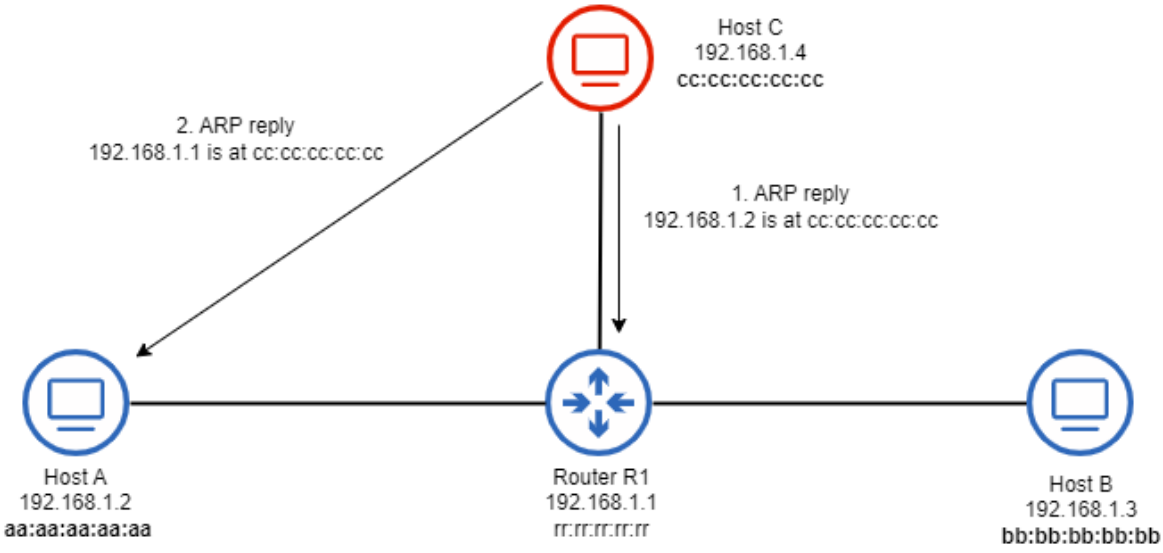


Figure 1.5: Example of ARP attack

By poisoning the ARP cache, the attacker gains control over the network traffic flowing between the victim (Client) and the router. In the diagram, Host C represents the attacker who intercepts the traffic between Host B (the victim) and the router.

To execute the ARP spoofing attack, the hacker sends two ARP responses. The first response is directed to the router, misleading it into believing that the hacker's MAC address is associated with the victim's IP address. The second response is sent to the victim, tricking it into associating the hacker's MAC address with the IP address of the router.

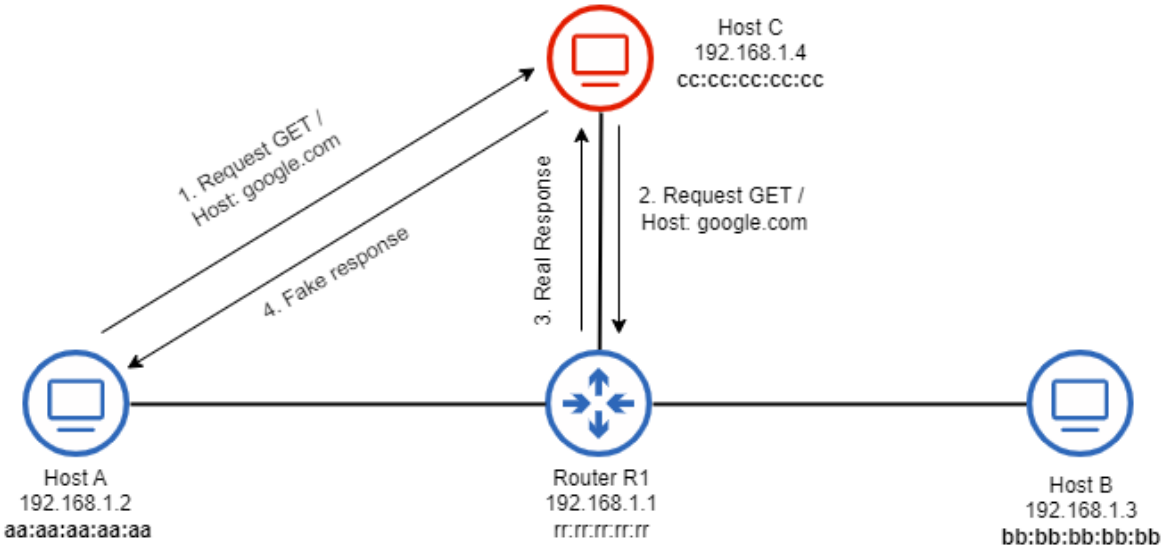


Figure 1.6: Example of ARP attack

In this diagram the Host A had already been tricked into associating the ip address of 192.168.1.1 which is the default gateway ip address to the attacker MAC address "cc:cc:cc:cc:cc".

When Host A tries to communicate with a destination, such as google.com, it sends the packet to what it believes is the default gateway (attacker's MAC address). Unaware of the manipulation, Host A assumes it is sending the packet to the legitimate router.

The attacker, upon receiving the packet from Host A, then forwards it to the actual router. The router processes the packet and sends a response back to the attacker. Once the response is received, the malicious Host C re-sends it to Host A, completing the bidirectional communication. The term "bidirectional" in this context refers to the fact that the attacker has successfully intercepted and manipulated the traffic in both directions: from Host A to the router and from the router back to Host A.

This manipulation of the ARP process allows the attacker to capture and analyze the network traffic between the victim and the router. By eavesdropping on this traffic, the hacker can intercept sensitive information, perform man-in-the-middle attacks, or redirect the traffic to a malicious server.

In order to execute a successful ARP spoofing/Poisoning attack in a real-life scenario, the attacker will do the following steps :

Scan the network: The attacker scan the whole subnet with ARP broadcast requests, this will give the attacker the list of connected devices.

Select targets: The attacker then selects from the discovered devices the target or targets. and because all devices in a local area network will communicate with the default gateway or the router when trying to send packets to an external destination. it is intuitive that the router is a target.

Send the ARP Reply packets: The attacker then is ready to spoof the ARP registry tables of the targets, he starts to sends continuously ARP reply packets to the selected hosts, trick them into thinking he is the default gateway. this could be done using graphical user interface ettercap, which makes it relatively very easy, or using terminal tool such arpspoof cli tool.

Intercept traffic: The victims will then try to communicate to the internet and sends their packets to the attacker mac address, the attacker can redirect that traffic to the real router in order to have legit requests. This make it very hard for the victim to notice anything at all. The attack can use tools such as wireshark to capture and analyse the victim's traffic, and potentially find sensitive information

III.5 Conclusion

This chapter we provides an overview of the project's context, explored the network environment at PixelJ and understood the need for strong security measures to protect against

potential threats.The proposed measures aim to enhance PixelJ's security by creating a way to monitor and analyse it's internal network leading minimizing the risks of internal threats. By creating a safe network environment, PixelJ can ensure that its data and systems remain secure and available.

As our project focuses particularly on Adress resolution protocol (ARP), we provided also a detailed coverage of the protocol, it's legitimate use, and the malicious intent use, we explain how ARP spoofing attack are performed and what could lead to.

Chapter 2

Requirements specification

I Introduction

This chapter outlines the requirements for this project. The system's primary objective is to provide comprehensive coverage of the network. And detect potential security threats as early as possible. The requirements are divided into functional and non-functional requirements, which present the specific features, behaviors, and performance characteristics that the system must include to meet the user's needs.

II Context

Our company has recognized the critical need to ensure proper network utilization and to define usage policies among connected users. It is imperative for us to maintain a secure network environment and protect sensitive data.
We recognizes the pressing need to fortify our network defenses against internal attacks. These types of attacks refers to the actions of **insiders** within an organization, such as employees, contractors, or partners, who intentionally or unintentionally compromise the security of the network. Internal threats can be particularly challenging to detect and prevent because insiders already have authorized access to the network.

III Functional requirements

The system aims to enable users to monitor their network, detect potential security threats, and prevent them. in this section we will list the detailed list of features required to achieve our goals in term of network monitoring needs.

III.1 Network monitoring

The system should allow users to monitor their network and see who is connected to it. The following features are required:

- **Device discovery**:
  The system should be able to discover and identify devices that are connected to the network.

- **Connection details**:
  The system should display the connection details of each device on the network, such as IP address, MAC address, hostname and quantity of data transmitted.

- **Conversations**:
  The system should allow users to view the conversations that are taking place on the network, including external hosts that are being communicated with.

- **Filtering**:
  The system should allow users to filter for a specific address or range of addresses.

- **Network speed**:
  The system should display the network download and upload speed, as well as the latency (ping) of the network.

III.2 Protocol analysis

The system should allow users to inspect the protocols used in the network, providing insights into network traffic patterns and potential security threats. The following features are required:

- **Ethernet protocol distribution**: The system should display a chart showing the distribution of Ethernet protocols used in the network, including IPv4, IPv6, and ARP. This feature allows users to gain a better understanding of the network's traffic composition, identify any unusual protocol usage, and detect potential security threats. For example, a sudden increase in the usage of a particular protocol may indicate an ongoing attack.

By analyzing the distribution of Ethernet protocols, the system can help identify potential security vulnerabilities and help prevent data breaches, data theft, and other malicious activities.

- **IPv4/IPv6 transport layer protocol distribution**: The system should display a chart showing the distribution of transport layer protocols used in the network, for both IPv4 and IPv6. This feature helps users identify the most commonly used transport layer protocols, such as TCP, UDP, and others, and provides insights into potential security threats.

- **IPv4/IPv6 Application layer protocol distribution**: The system should display a chart showing the distribution of application layer protocols used in the network, for both IPv4 and IPv6. These are the application layer protocols that the system should be able to recognize the following protocols :

1. TCP
   a) HTTP b) HTTPS c) SMTP d) SSH e) TELNET
   f) FTP g) DNS h) RDP
2. UDP
   a) HTTP b) HTTPS c) DNS d) DHCP e) DHCPv6
   f) NTP g) OpenVPN h) RDP

III.3 Intrusion detection and prevention

The system should be able to detect and prevent potential security threats. The following features are required:

- **ARP spoofing attack detection**: The system should use machine learning algorithms to be able to detect ARP spoofing attacks, which are a type of man-in-the-middle attack. The system must be trained with a dataset of normal and malicious ARP traffic to accurately detect and classify ARP spoofing attacks in real-time.

- **Attack response**: In addition to detecting the attack, the system should also be able to respond to such attacks in real-time. Upon detection of an attack, the system should automatically initiate a response mechanism to mitigate the attack. One possible response mechanism is to de-authenticate the malicious MAC address, which would effectively terminate the attack. The system should alert the system administrator or security team about the attack to confirm the action execution and provide relevant information for further investigation and remediation. It is important that the attack response mechanism is efficient and effective to minimize the potential impact of the attack on the network and its users.

III.4 Mobile application

The system should have a mobile application that will receive notifications in case of detection and enable users to take action. The mobile application should provide a convenient way for users to monitor the network and respond to security incidents in real-time, even when they are not physically present in the office. The following features are required:

- **Notification :**
  The mobile application should receive a notification when an attack is detected. Notifications can be customized to provide details about the type of attack, the affected device or network segment, and the recommended course of action. Users can choose to receive push notifications, SMS messages, or email alerts, depending on their preference and availability.

- **Action :**
  The mobile application should enable the user to take action when an attack is detected. The user can choose to de-authenticate the malicious MAC address or block traffic from the affected device or network segment. The mobile application should provide a simple and intuitive interface for users to perform these actions, without requiring advanced technical knowledge or expertise. In addition, the application should log all actions taken by users, to ensure accountability and facilitate auditing and reporting.

IV Non-functional requirements

The non-functional requirements describe the quality attributes of the system. The following is a list of the non-functional requirements for the system:

- **Performance**: The system should have a fast response time when detecting an attack, with a goal of less than 5 minutes. This requirement ensures that potential threats are identified and addressed in a timely manner, reducing the risk of damage to the network and data.
- **Modularity:**

  The system should be designed in a way that enables future development to add **handlers** independently from previous ones. For example, if a new type of attack is discovered, a new packet handler can be created and added to the system without affecting the existing handlers.
- **Security**: The system should be secure and protect user data from unauthorized access or disclosure. This includes measures such as encryption of sensitive information, secure storage of user data, and access control mechanisms to prevent unauthorized use of the system.
- **Usability**: The system should be designed to be easy to use and intuitive, with a clear and user-friendly interface. This is important to ensure that the system can be easily adopted by users with varying levels of technical expertise, and that users can quickly and easily navigate the system to perform necessary tasks.
- **Reliability**: The system should be reliable and perform consistently over time, with minimal downtime or interruptions. This is essential to ensure that the system is always available to detect and respond to potential threats, and that users can rely on the system to provide accurate and up-to-date information.
- **Maintainability**: The system should be designed to be easy to maintain and upgrade, with clear documentation and well-structured code. This is important to ensure that the system can be easily updated and improved over time, and that any issues or bugs can be quickly identified and addressed. Additionally, the system should be designed to minimize the impact of any changes or upgrades on user workflows and processes.

V Use cases

V.1 Actors presentation

The system has two types of users: a super administrator and a network administrator. The super administrator has full permissions to access and manage all the features of the system, including the ability to create, modify, or delete user accounts. On the other hand, the network administrator has access to all the features except for the ability to manage users accounts.

V.2 Global use case diagram

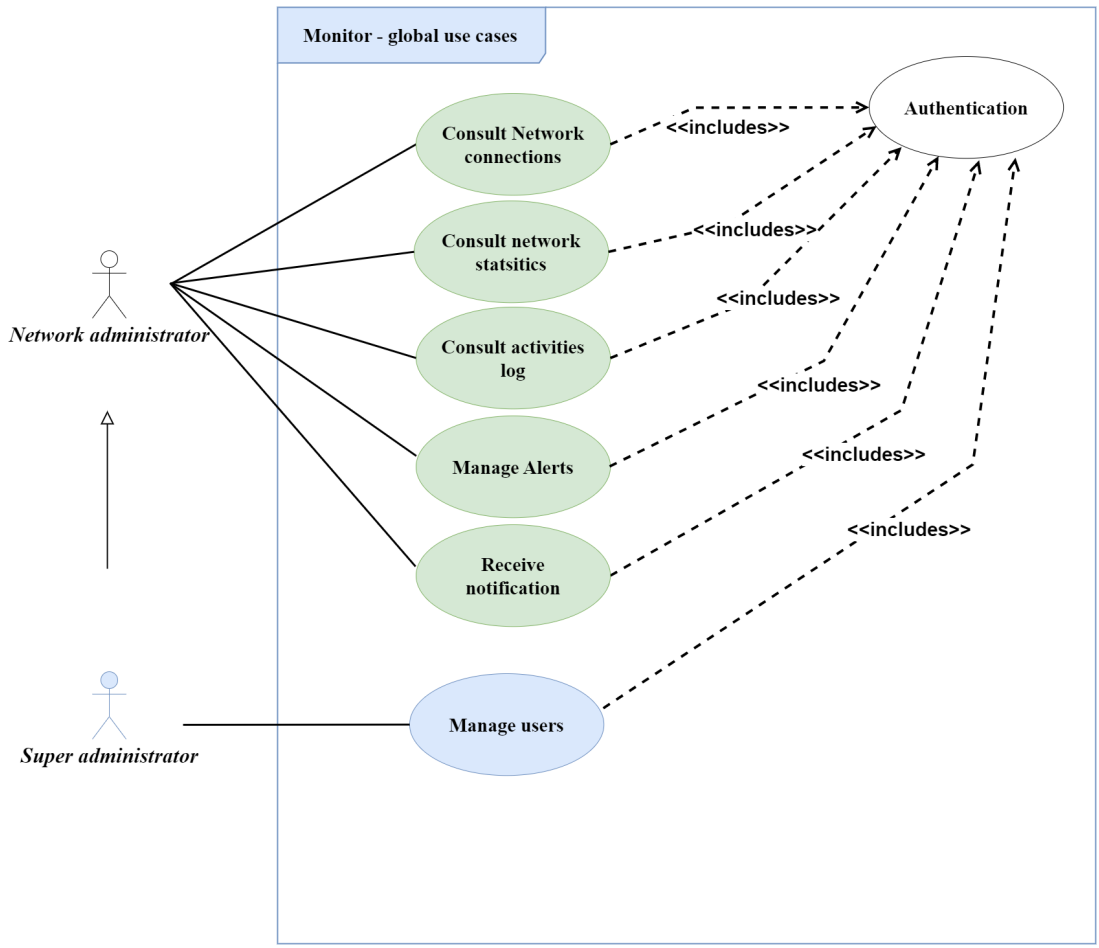


Figure 2.1: Global Use Case Diagram

The use case diagram shown in Figure 2.1 illustrates the functionality of a network administration system. The use cases encompass activities such as retrieving network connection details, consulting statistical charts for analysis, reviewing activity logs for troubleshooting purposes, and receiving real-time alerts for critical events. All these use cases serve to provide the

network administrator with a comprehensive toolkit for efficient network administration and maintenance.

V.3 Use Case 1: Users Management

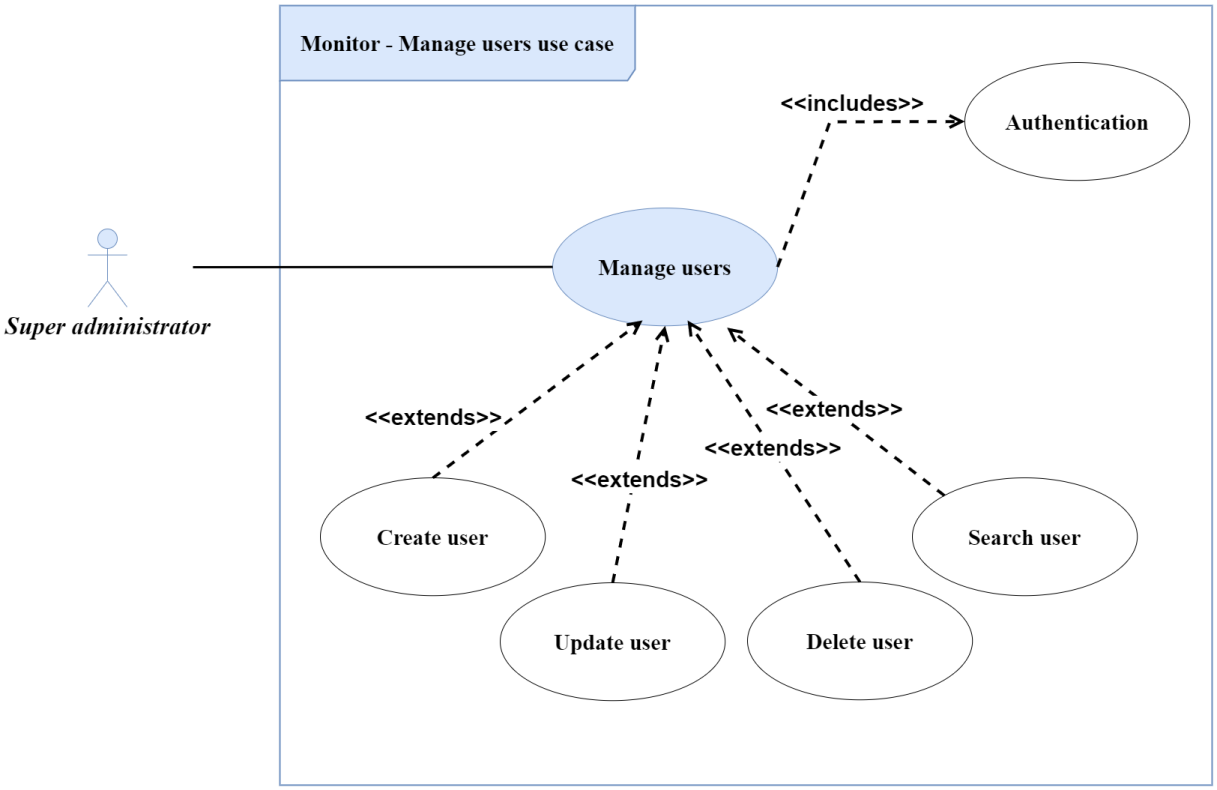


Figure 2.2: Manage users Use Case Diagram.

The ”User Management” use case in the Network Administration System involves the Super Administrator's exclusive ability to create, update, and delete user accounts. As the system's highest-level authority, the Super Administrator holds the responsibility of managing user accounts and ensuring appropriate access and permissions. The use case begins with the creation of new user accounts. Additionally, the Super Administrator is responsible for updating existing user accounts.

Due to the similarity between the ”Update User” and ”Delete User” and the ”Create User” scenarios, we will focus on providing a detailed description specifically for the ”Create User” use case. The ”Manage Users” use case, as shown in Figure 2.2, covers the actions related to creating, updating, and deleting user accounts within the Network Administration System.

Table 2.1: ”User creation” description

Actors	Super Administrator
Goal	To create a new user account for the Network Administration System
Precondition	The system is running and accessible to the Super Administrator.
Postcondition	A new user account has been created and added to the system
Main Scenario	1. The Super Administrator logs in to the Network Administration System. 2. The Super Administrator selects the ”Add Admin User” option from the users page. 3. The system displays a form for the Super Administrator to enter the new user’s information, such as username, password, and email. 4. The Super Administrator fills in the form with the new user’s information and submits it. 5. The system verifies the input and creates a new user account with the provided information.
Alternate Scenarios	1. The Super Administrator enters invalid input in the form. 2. The system displays an error message and prompts the Super Administrator to correct the input.

V.4 Use Case 2: Network connections

The ”Network connections” use case involves the process of identifying devices connected to the network, whether wired or wireless. The system allows the user to visualize the physical or logical structure of the network, including the relationships between devices and their connections. This helps identify and diagnose problems that may impact network performance. The user can interact with the topology graph to filter, search and explore specific devices and connections in greater detail, enabling more efficient troubleshooting and management of the network.

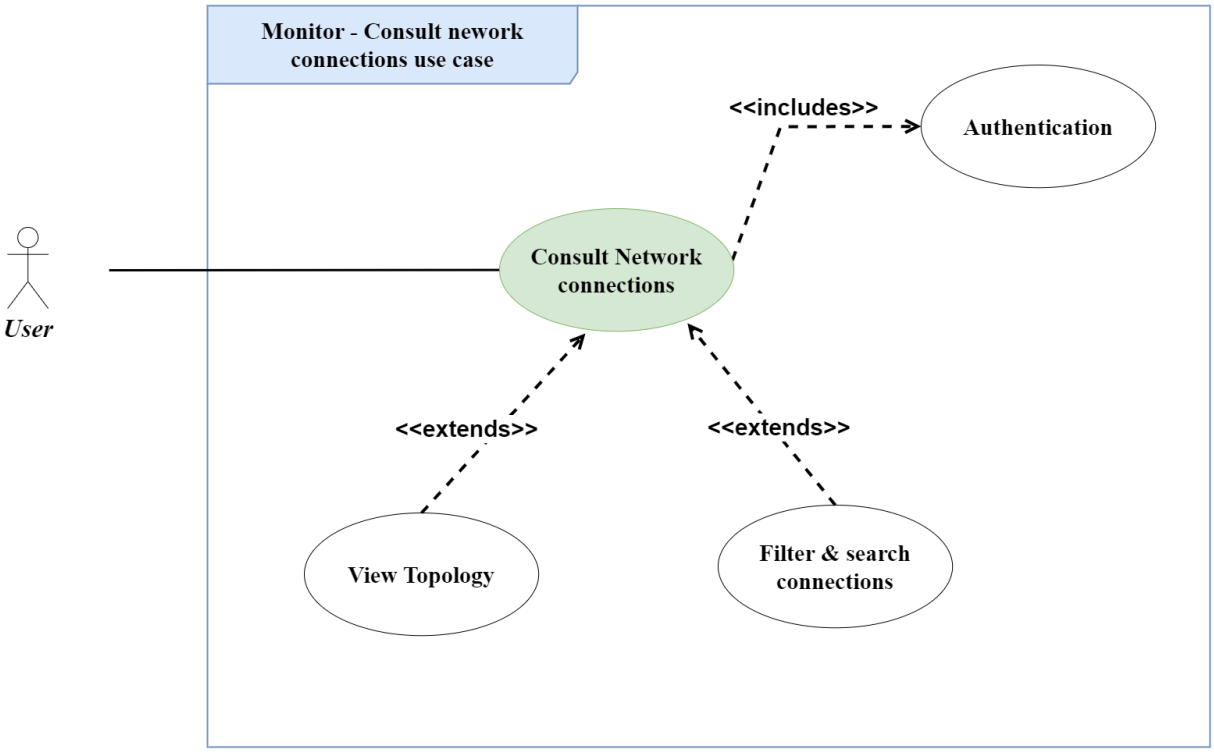


Figure 2.3: Network connections use case diagram.

1. View network topology

The ”View Network Topology” use case is an extension of the ”Network Connections” use case. It allows the Network Administrator to access and visualize the network topology graph, which represents the interconnected devices and their connections within the network.

Table 2.2: ”Consult network connections” description

Actors	Super Administrator, Network Administrator
Goal	To discover all active devices on the network and record their basic information
Precondition	1. The system is running and connected to the network 2. The network administrator is logged in to the system.
Postcondition	1. A list of all active devices on the network has been generated and stored in the system 2. The network topology is represented in graphical and tabular formats in the web dashboard in the conversations tab

Main Scenario	1. The Network Administrator initiates the system starting process. 2. The system scans the local network 3. The system stores the list of active devices and their basic information in its database. 4. The system notifies the Network Administrator that the device discovery process is complete and that the traffic capturing processes has started. 5. The system stores the captured traffic conversations data and displays a topology graph
Alternate Scenarios	1. The System doesn't find any connected local devices 2. The system notifies the Network Administrator that the device discovery process is complete and that the traffic capturing processes has started. 3. The system stays in stale mode until new traffic has been generated

2. Filter connections

The "Filter connections" use case enables the user to quickly search for specific devices or sets of devices on the network based on IP addresses or ranges. This use case helps to streamline the process of identifying and isolating devices for troubleshooting or management purposes. By filtering the list of devices, the user can focus their attention on specific devices or sets of devices, making the network management process more efficient and effective.

Table 2.3: ”Filter connections” description

Actors	Super Administrator, Network Administrator
Goal	To filter the list of devices on the network based on a specific IP or MAC address or host name
Precondition	1. The system is running and connected to the network 2. The network administrator is logged in to the system. 3. The list of connections and devices on the network has been generated.
Postcondition	A filtered list of devices based on the filter has been generated and displayed to the user
Main Scenario	1. The Network Administrator requests the connections page. 2. The Network Administrator enters either the MAC or IP address to filter for in the search input. 3. The system filters the list of connections on the network to only display the conversations that contain match the specified address. 4. The system displays the filtered list of connections to the Network Administrator.
Alternate Scenarios	1. If no devices are found that match the specified address, the system displays a message indicating that no devices were found.

V.5 Use Case 3: Network statistics

The ”Network statistics” use case involves the process of allowing users to view the protocols used in the network. This includes displaying the distribution of Ethernet protocols, transport layer protocols, and application layer protocols used in the network, for both IPv4 and IPv6.

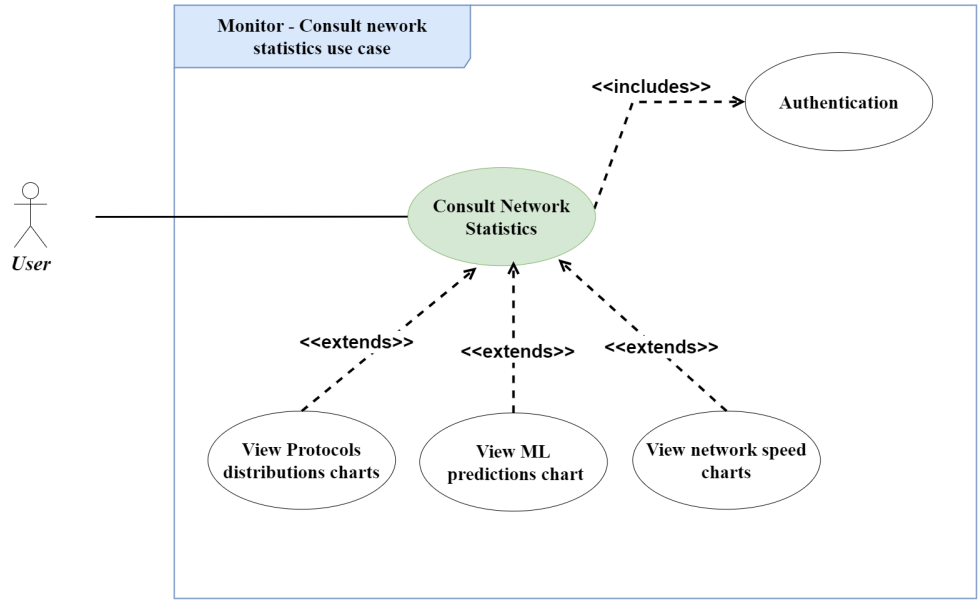


Figure 2.4: Network statistics use case diagram.

1. Protocols distribution chart

Table 2.4: ”Protocol distribution” description

Actors	Super Administrator, Network Administrator
Goal	To view the distribution of Ethernet protocols, transport layer protocols, and application layer protocols used in the network for both IPv4 and IPv6
Precondition	1. The system is running and connected to the network 2. The network administrator is logged in to the system.
Postcondition	The user is presented with charts showing the distribution of Ethernet protocols, transport layer protocols, and application layer protocols used in the network for both IPv4 and IPv6
Main Scenario	1. The user selects the dashboard page from the network monitoring menu 2. The system generates charts showing the distribution of Ethernet protocols, transport layer protocols, and application layer protocols used in the network for both IPv4 and IPv6. 3. The system displays the charts to the user.
Alternate Scenarios	None

2. Machine learning chart

This chart is an intuitive visual way of detection internal threat, in particular the one that is the interest of this project (ARP poisoning attacks). the chart must display a time-series multi-line chart that indicate the probability of the packet being malicious for each conversation. when the attack occurs the line that represent the particular conversation that is involved in the attack will raise to higher probabilities of attack classification, triggering an alert raise.

Table 2.5: "Machine learning chart" description

Actors	Super Administrator, Network Administrator
Goal	To view in real time the classification machine learning algorithm in action, the classification output and the probability of it are displayed for each unique conversation
Precondition	1. The system is running and connected to the network 2. The network administrator is logged in to the system.
Postcondition	The user is presented with multi-line chart showing the diffrent values of the ML predictions probabilities in time, allowing to visually identify the anomalie when it occurs
Main Scenario	1. The user selects the dashboard page from the network monitoring menu 2. The system generates a chart showing the timeseries values off the machine learning classification probabilities 3. The system displays the charts to the user.
Alternate Scenarios	None

V.6 Use Case 4: Activity logs

The "Activity Logs" use case allows the user to access and review the recorded activities within the system. These activities encompass a variety of events, including system status updates such as the start and stop of captures, as well as the logging of newly discovered local hosts through the host discovery process. Additionally, any user actions related to creating, updating, or deleting user accounts are also logged for reference. Note that the system doesn't provide the ability to delete or update activities for history integrity purposes.

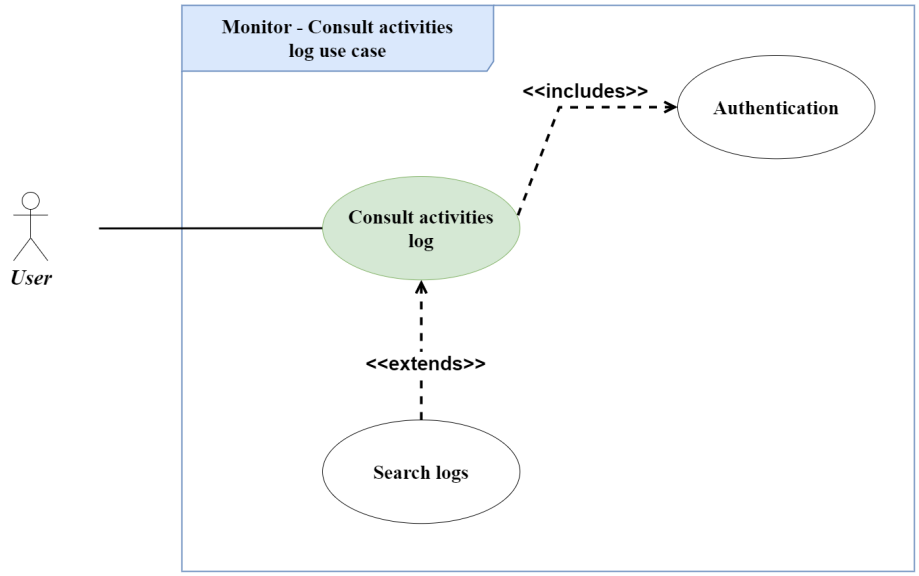


Figure 2.5: Network activities log use case diagram.

1. Search logs

The ”Search Logs” use case allows users to search and retrieve specific logs based on their requirements. This functionality enables them to locate and analyze particular activities or events recorded within the system.

Table 2.6: ”Search logs” description

Actors	Super Administrator, Network Administrator
Goal	To search and retrieve specific logs based on user-defined criteria
Precondition	1. The system is running and connected to the network. 2. The user (either the Super Administrator or Network Administrator) is logged into the system.
Postcondition	The user is presented with the requested logs based on the search criteria
Main Scenario	1. The user enter the search term, in the input field in the activities page. 2. The system performs the search operation based on the provided criteria. 3. The system retrieves and present the relevant logs that match the search criteria.

V.7 Use Case 5: Alert management

The ”Alerts management” use case allows the user to access alert, execute actions if available. Additionally the user can create a new alert and choose whether to notify other admins, finally a user can choose to delete an alert.

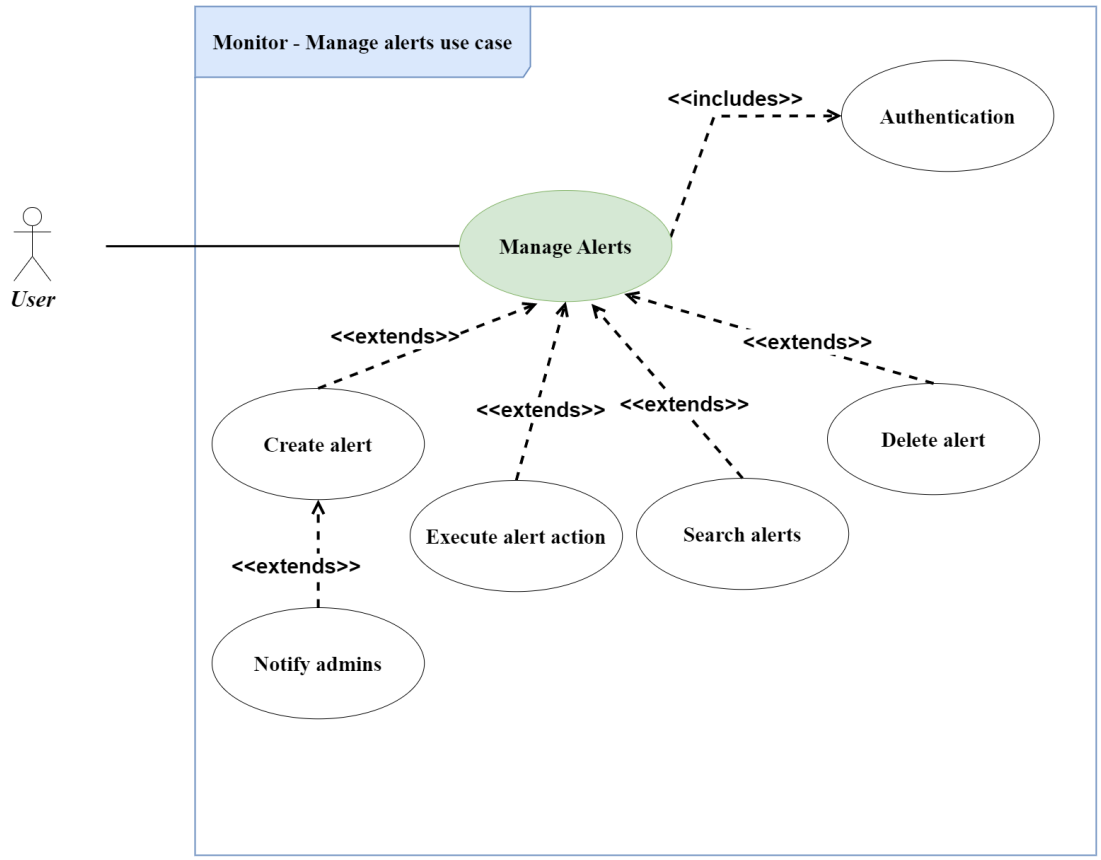


Figure 2.6: Alerts management use case diagram.

1. Alert creation

The ”Alert creation” use case allows users to create new alerts in the system, assign actions to alerts (currently limited to de-authenticating a MAC address), and choose whether or not to notify other users. This functionality provides users with a flexible and customizable approach to monitoring and responding to critical events within the network administration system.

Table 2.7: ”Create Alert” Description

Actors	Super Administrator, Network Administrator
Goal	To create a new alert with specific criteria and optionally assign an action (deauthentication of a MAC address) while also optionally notifying other users
Precondition	1. The system is running and connected to the network. 2. The user (either the Super Administrator or Network Administrator) is logged into the system.
Postcondition	The user successfully creates a new alert within the system with the specified title and description, action, and notification settings
Main Scenario	1. The user navigates to the alert management section of the system. 2. The user selects the option to create a new alert. 3. The user specifies the title, description for the alert. 4. The user assigns the action of deauthenticating a MAC address to the alert. 5. If the user chooses a action he must also provide a mac address to be assigned to the action. 6. The user chooses whether or not to notify other users for the alert. 7. The system validates the provided information and saves the new alert configuration. 8. The new alert show in the list of alerts with the status ”active”
Alternate Scenarios	None

V.8 Use Case 6: Receive notifications

The ”Receive Notifications” use case enables users to receive notifications from the system. These notifications can serve different purposes, including providing basic information, alert notifications, or custom-created notifications. By implementing this functionality, the system ensures that users stay informed and up-to-date with relevant events and updates.

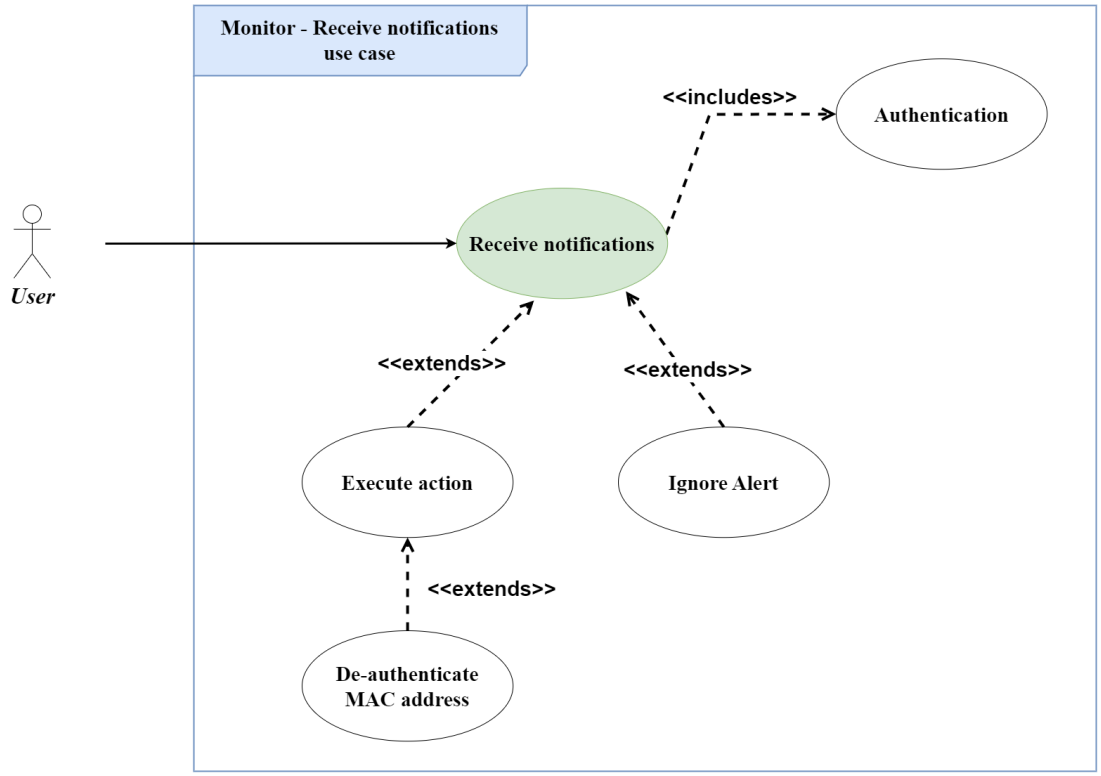


Figure 2.7: Notifications use case diagram.

1. Receive alert notifications description

The ”Receive alert notifications” use case allows users to receive notifications from the system regarding critical events or system conditions that require immediate attention or action. These notifications ensure that users are promptly informed about any urgent issues that may impact the system’s functionality or security. in our project scope we will be interested particularly on ARP poisoning detection alerts.

ARP attacks can be a serious security threat to a network, allowing an attacker to intercept and manipulate network traffic. Detecting and responding to these attacks quickly is essential to prevent further damage. This use case outlines how the system can detect and notify the user of an ARP poisoning attack using machine learning, the model must be trained to accurately detect malicious ARP activities. The notification should contain sufficient information to help the user understand the attack and what actions they can take to address it.

Table 2.8: Receive alert notifications use case description

Actors	Super Administrator, Network Administrator
Goal	To detect and notify the user of an ARP poisoning attack on the network using machine learning rather than static signatures.

Precondition	1. The system is running and connected to the network 2. The network administrator is logged in to the system.
Postcondition	The user is notified of the attack and the MAC address associated with the attack.
Main Scenario	1. the system inspect each packets in the network in real time, extracts the relevant data and feeds it to the machine learning model 2. The machine learning model for ARP spoofing detection detects an attack. 3. The system saves the information about the occurrence, including the time and MAC address that performed the attack. 4. The system sends a notification to the the mobile/web application users. 5. The notification contains the MAC address associated with the attack and an explanation of the attack, including the potential consequences and recommended actions. 6. The user receives the notification on their mobile device or in the web dashboard and can take appropriate action to address the attack, such as blocking the offending MAC address.
Alternate Scenarios	1. If the machine learning model is not able to detect the ARP spoofing attack, the user will not receive any notification. 2. If the user does not have a mobile application and he is not logged in to the dashboard, the notification is saved to activity logs and the user will be notified via email.

2. De-authenticating a MAC address description

De-authenticating a MAC address from network can be necessary for security purposes, such as preventing unauthorized access to the network or removing a device that has been com-

promised. This use case outlines the steps a network administrator can take to de-authenticate a MAC address from the network. The system should provide clear prompts and confirmations to help prevent accidental de-authentication and ensure the user is entering valid MAC addresses. Additionally, the system should log the action for future reference and update network monitoring data to reflect the change.

Table 2.9: De-authenticating a MAC address use case description

Actors	Super Administrator, Network Administrator
Goal	To de-authenticate a MAC address from the network.
Precondition	1. The system is running and connected to the network. 2. The user is logged in to the system. 3. An alert has been raised and the MAC address to be de-authenticate is identified.
Postcondition	The specified MAC address is de-authenticated from the network.
Main Scenario	1. The user receives an alert either by mobile push notification or in the web application. 2. The user selects the de-authendication action from the alert. 3. The user confirms the de-authentication. 4. The system de-authenticates the MAC address associated with the alert.
Alternate Scenarios	1. The user attempts to confirm the de-authentication, but the system is unable to complete the action. 2. The system sends an error message to the user, indicating the reason for the failure (e.g., the MAC address is not currently connected to the network).

VI Conclusion

In this chapter we have presented a comprehensive listing of the system's functional and non-functional requirements. These requirements serve as the foundation for the development

and implementation of the project. To ensure a clear understanding of the system's functionality, we have included a use case global diagram. This diagram provides an overview of the various interactions between the system and its actors.

Furthermore, we have provided a detailed description and explanation individual use cases. By doing so, we aim to provide a clear understanding of the specific actions and functionalities that the system should support. Each use case description outlines the steps involved, the involved actors, and the expected outcomes.

Chapter 3

System Design

I Introduction

In this chapter, we provide a detailed overview of our system's architecture, organization, and behavior. In the dynamic modeling section, we use UML sequence diagrams to explain the system's behavior in different scenarios.

In the static modeling section, we show the system's class diagram understand its architecture and component organization. The class diagram displays the relationships and interactions between the different classes within the system.

II Dynamic modeling

II.1 Sequence diagrams

The sequence diagrams illustrate the interactions between the different components of our system in specific scenarios. Each sequence diagram shows a step-by-step process of how the components communicate and collaborate to achieve a specific task.

It is a type of interaction diagram that shows the exchange of messages between objects or components over time. It depicts the objects and their interactions with each other and the order in which these interactions occur. Sequence diagrams are often used to represent complex scenarios and are particularly useful in modeling scenarios that involve multiple objects or components.

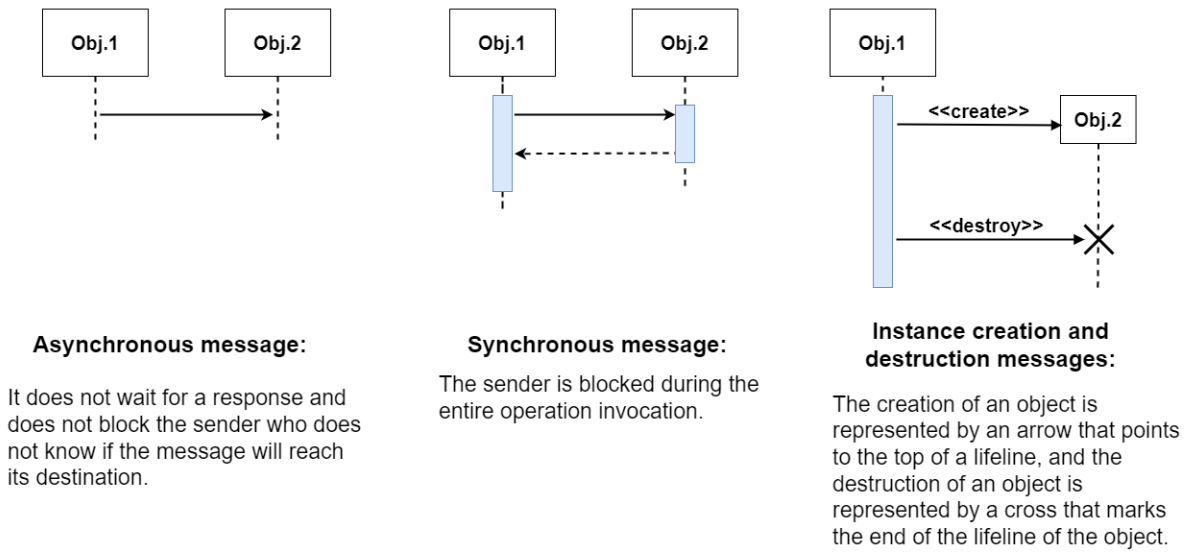


Figure 3.1: Messages types.

The figure in this section displays the different types of messages that can be represented in UML sequence diagrams, including asynchronous messages, synchronous messages, instance creation messages, and instance destruction messages. In our system design, we will be using these message notations to explain the interactions between various components and objects of the system.

II.1.1 Authentication sequence

The first sequence diagram we present is the Authentication sequence. This sequence diagram showcases the process of authenticating a user on the system.

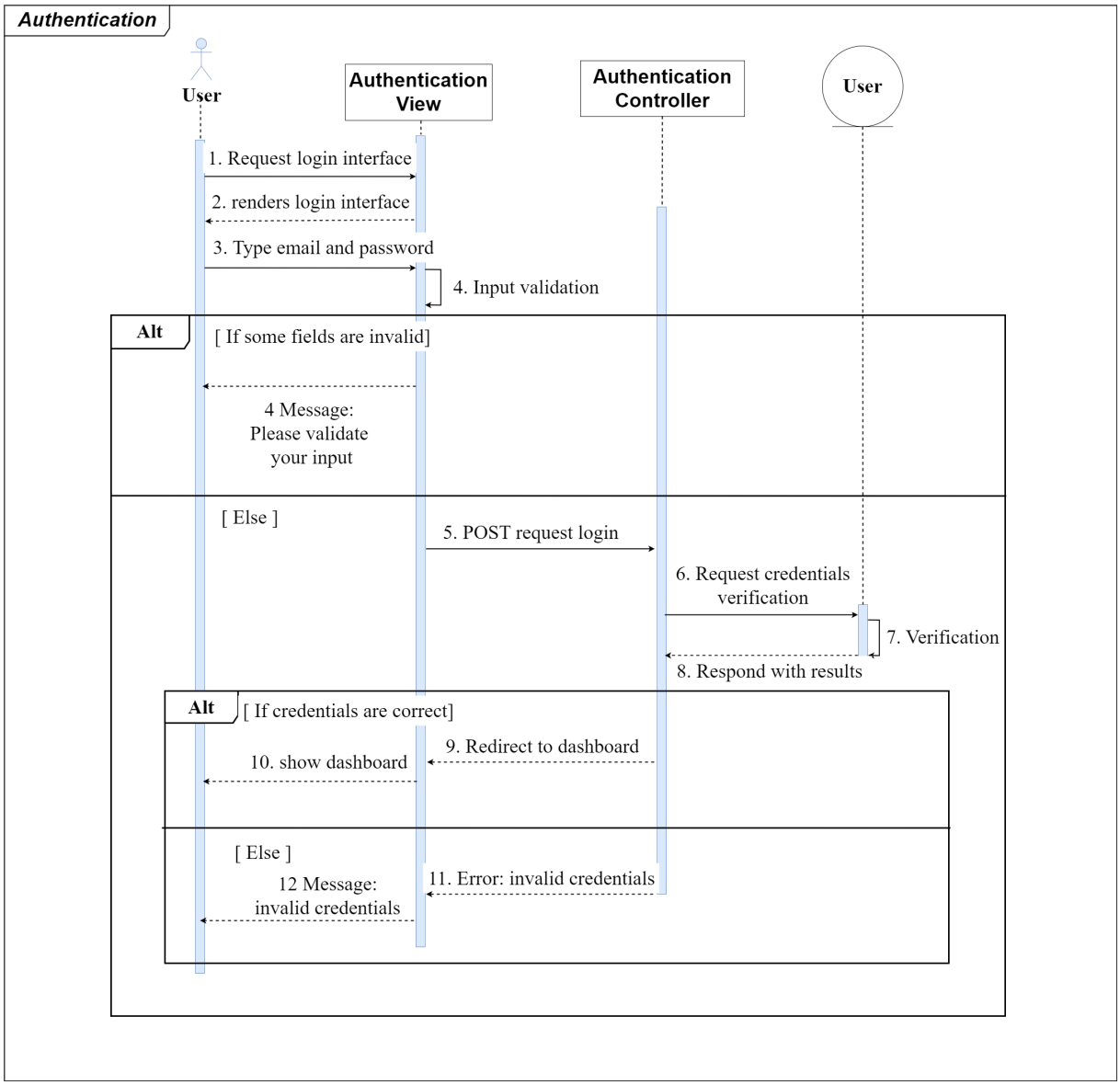


Figure 3.2: Sequence diagram for Authentication use case.

The sequence begins with the user sending their login credentials to the system, the front-end will validate the email before making the request, once the request is been sent, the server verifies the credentials (Authentication controller requests the database) and sends a response back to the user. If the credentials are valid, the system grants the user access to the requested resources and redirect them to the dashboard page. However, if the credentials are invalid, the system denies access and prompts the user to re-enter their credentials.

II.1.2 Consult network topology sequence

The next sequence diagram we present is the ”Consult network topology” sequence. This sequence diagram demonstrates the process of consulting the network topology and visualizing the connections between nodes.

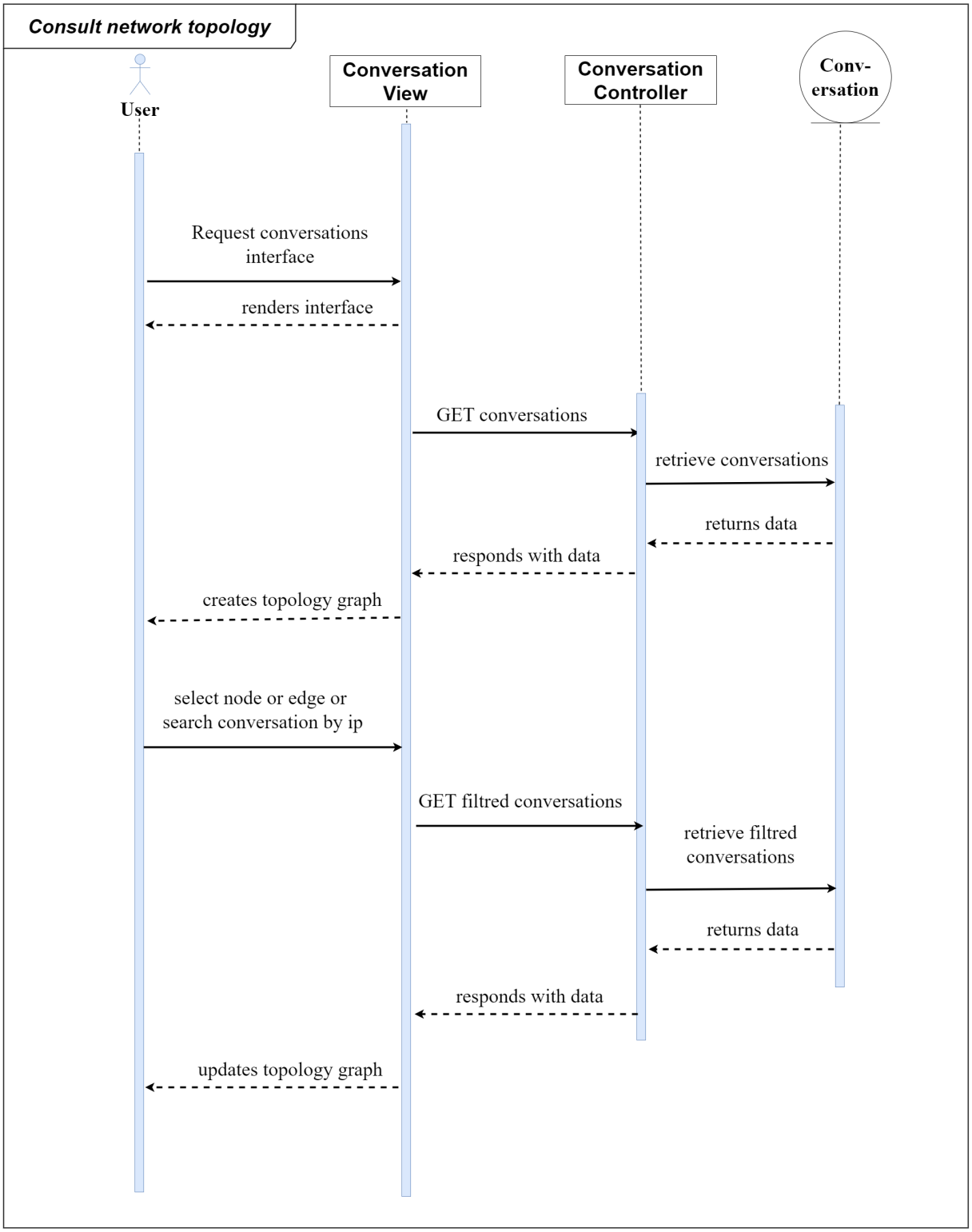


Figure 3.3: Sequence diagram for Consult network topology use case.

The sequence begins with the user requesting the network topology page. The view initializes and sends a request to the controller to retrieve the conversation data. The controller then retrieves the necessary data from the model and returns it as a response. Once the response is received, the view presents the user with a graph and nodes chart, displaying information

such as IP addresses and the number of transmitted packets. This visualization allows users to visually understand the connections between nodes in the network.

II.1.3 Start capture sequence

The process of starting the network monitoring and intrusion detection involves a series of steps. It begins with the network administrator initiating the process and then a user accessing the web application to send a ”start-capture” request. This request serves as a trigger for the network traffic capturing process. The web application then checks if the monitoring system is already capturing network traffic. If it is, the user receives a response indicating that the process is already running. However, if the monitoring system is not currently capturing network traffic, the server responds with a ”200 OK” status, signifying a successful request. Along with this response, the server also initializes the packet handlers.

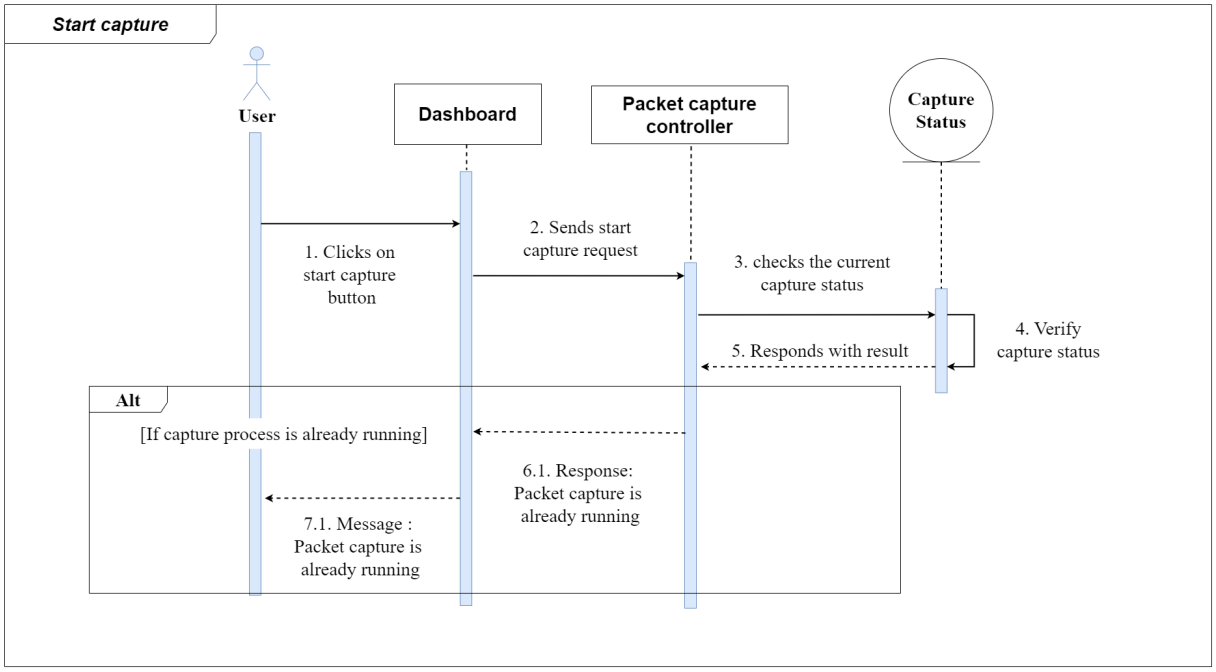


Figure 3.4: Sequence diagram for network capture process. (1)

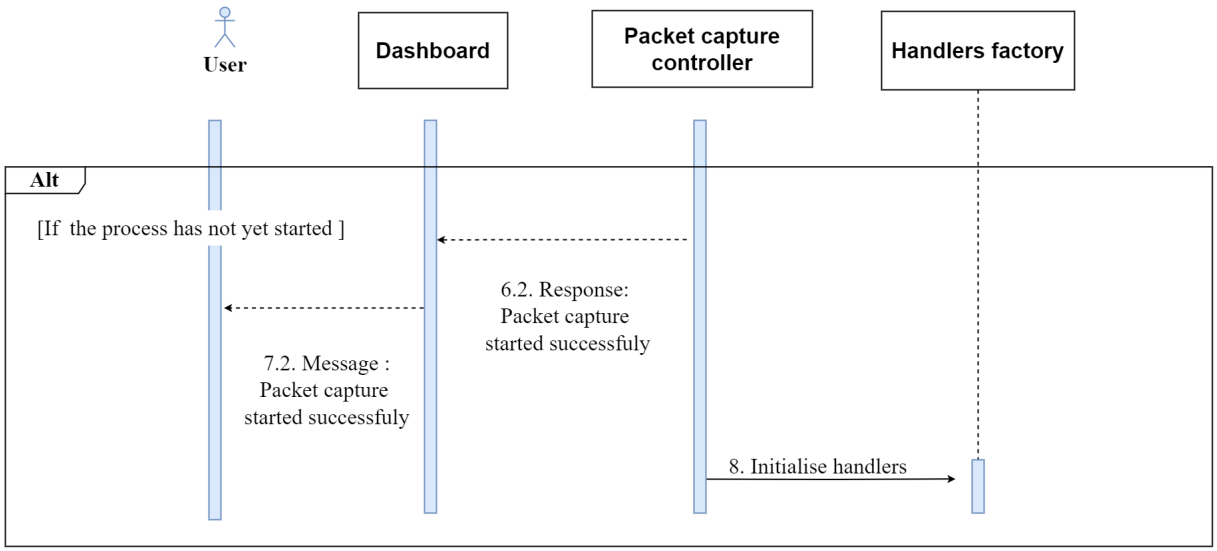


Figure 3.5: Sequence diagram for network capture process. (2)

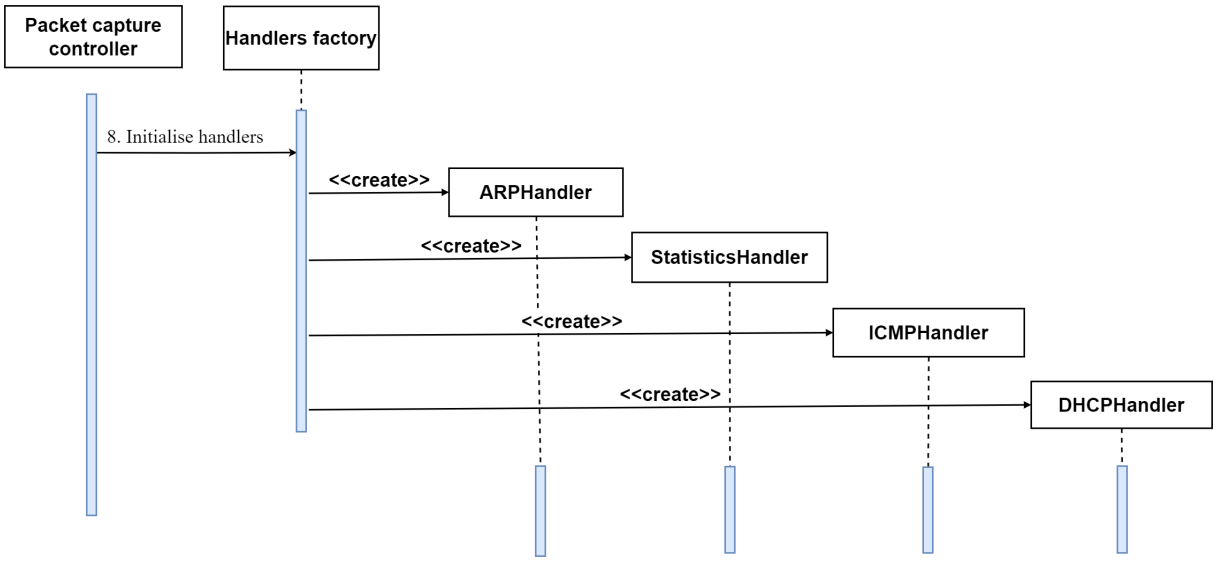


Figure 3.6: Sequence diagram for network capture process. (3)

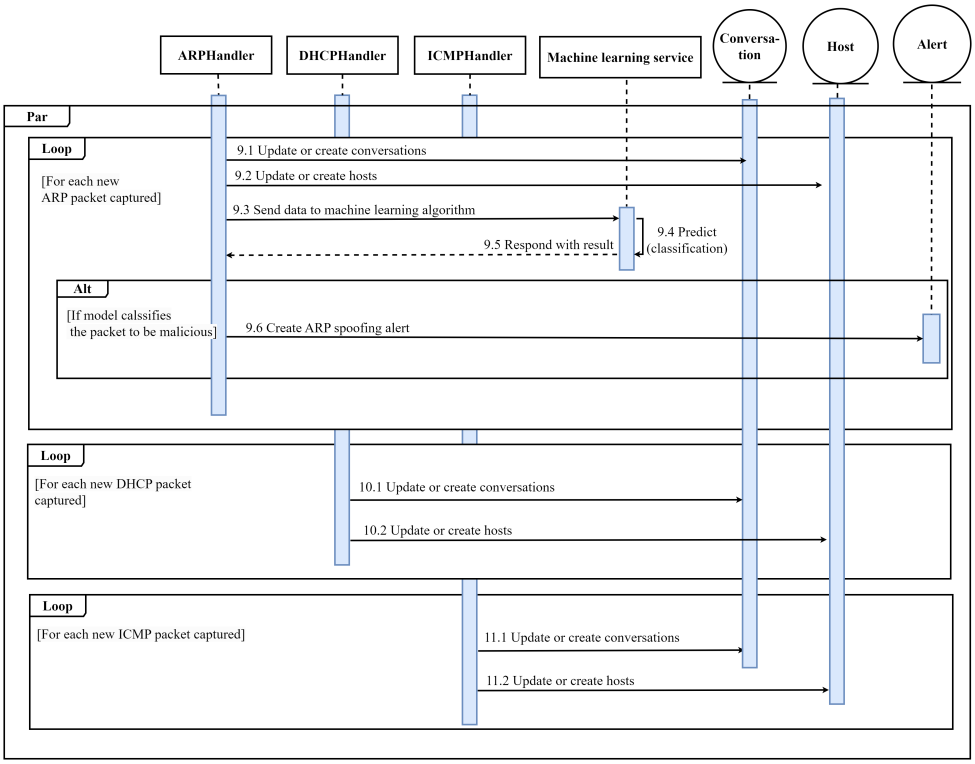


Figure 3.7: Sequence diagram for network capture process. (4)

The DHCP handler is responsible for optimizing response time in discovering new connected devices to the network. To achieve this, the handler has its own traffic capture process that only listens to DHCP packets.

This ensures that the system can discover new devices significantly faster than if it only had one capture process for all the traffic, as the volume could be much greater than the speed at which the system could process packets. By dedicating a separate capture process to DHCP traffic, the system can gain valuable time in response time.

The ARP handler, similarly, listens only to ARP traffic. It extracts information from each packet captured and updates the hosts and conversations tables in the database. The handler then passes the data to a machine learning trained model for ARP spoofing attack classification.

In addition to the DHCP and ARP handlers, the system also includes a statistics handler. The statistics handler listens to all traffic and is responsible for calculating metrics and saving protocol information such as the ethernet type , the transport layer protocol and the application layer protocol. The handler is responsible for extracting and saving information such as destination , protocol, packet sizes, and other statistics to provide insights into network usage patterns and identify anomalies.

II.1.4 Create alert sequence

The following sequence diagram illustrates the process of creating an alert in the system.

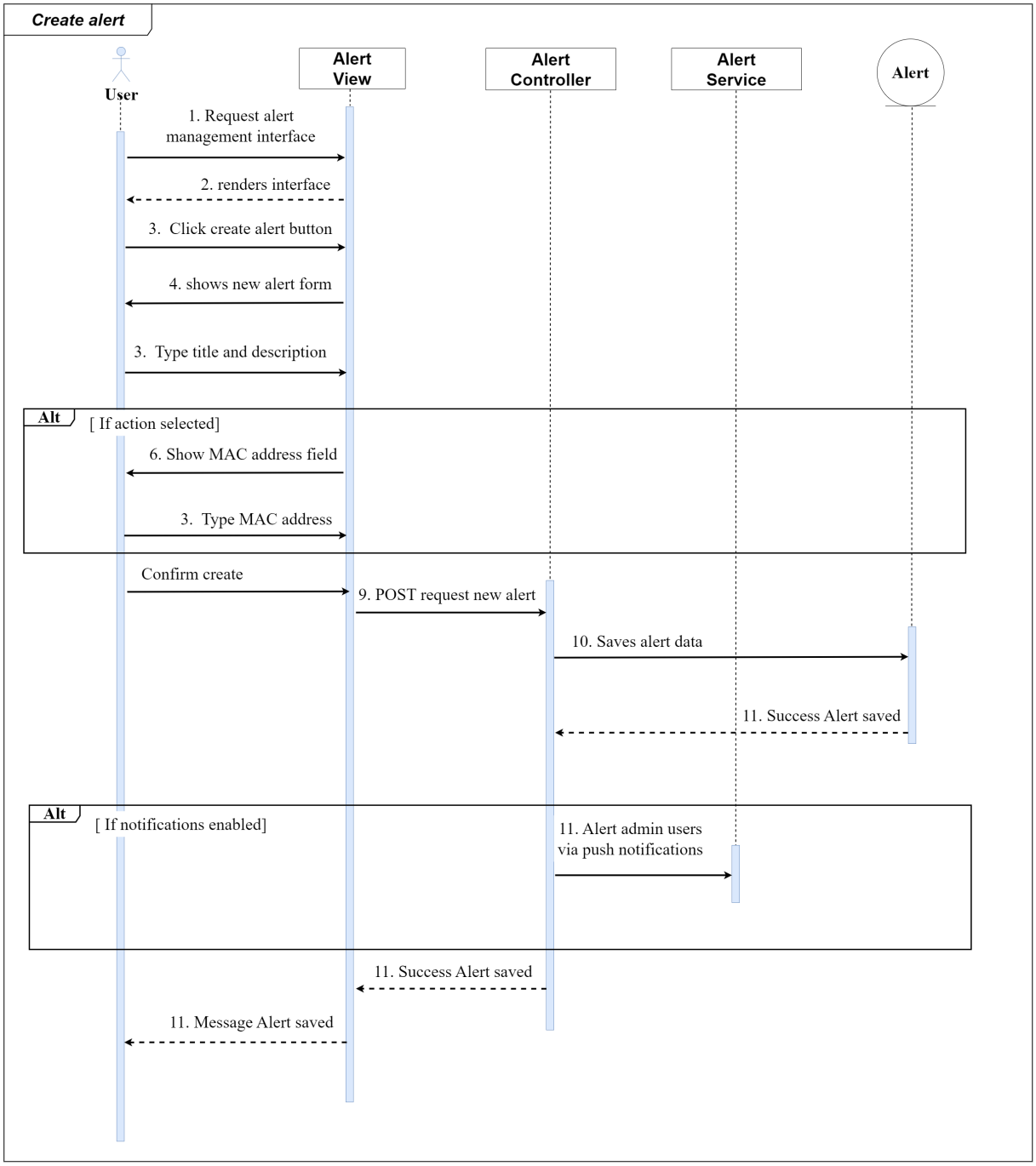


Figure 3.8: Create alert sequence diagram

The sequence begins when the user requests the create alert page. The view then presents a form to the user, where they can enter the title and description for the alert. Additionally, the user has the option to associate an action with the alert. If the user chooses to associate an action, they must also provide a MAC address. The view includes a checkbox for notifying other users via push notification. Once the user fills in the form and submits it, the view sends the data to the controller. The controller receives the alert data and proceeds to save it in the system. If the user has selected the option to send notifications, the controller communicates with the Alert service to trigger the notification process.

III Static modeling

III.1 Class diagram

In this section we will provide the class diagram of the solution system, this diagram will be helpful to understand the models the methods and attributes and relations of the different entities of our application.

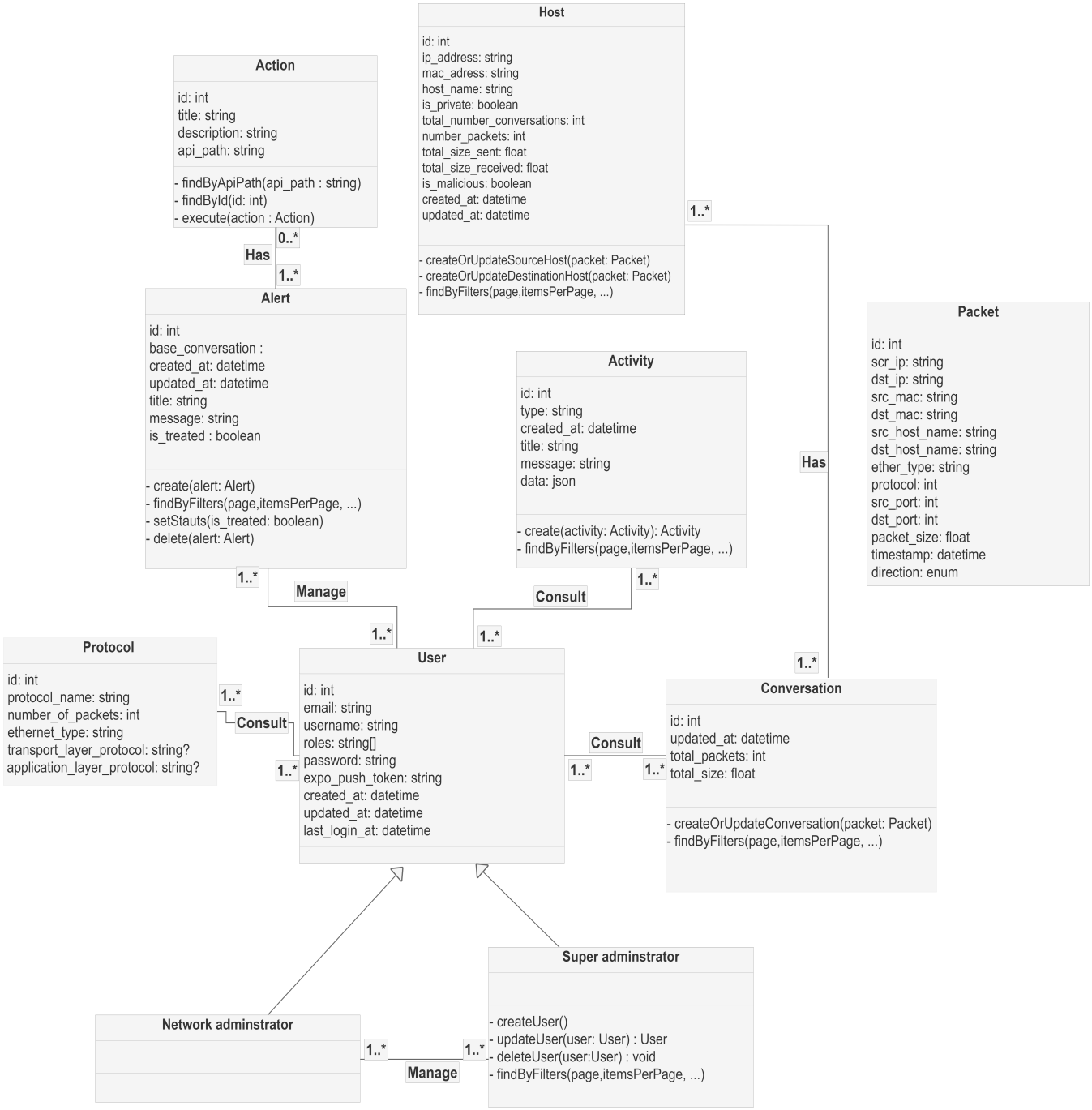


Figure 3.9: System class diagram.

III.1.1 User

The User model represent users in our application, its is a crucial model for making the system secure and create an authentication process, the roles attribute ensures the resources authorization, as we can see there is two main types of user, the super administrator class inherits all the attributes from the user model and contain extra methods, like create or update users. The pushtoken attribute is used for mobile push notifications.

III.1.2 Packet

The packet class is a the model representation of a real network packet, it contain the basic data of a network frame, like the source and destination ip addresses, mac addresses, size, protocol, source and destination ports, Ethernet type, timestamp and direction. the packet is then transformed if possible to a conversation. each packet can create or update a single conversation at a time, because a packet has only one source and one destination.

III.1.3 Conversation & Host

A conversation is very important table or model in our design as it saves and encapsulate the data of network conversation. a network conversation is a connexion between two hosts that can last for a certain amount of time, for example if a host by the IPv4 address of ”192.168.1.x” sends a packet with the destination of the IP address of ”domain.com” , the system handler must first create two hosts in the Host table, using the methods ”createOrUpdateSourceHost” and ”createOrUpdateDestinationHost” , the creates a new conversation in the Conversation table to represent this connexion, later on, if the the same host ”192.168.1.x” sends another packet to the same destination host, the handler must only update relevant attributes in the host and also in conversation, these attributes include primarily the number of packet transmitted, the size of received traffic and the size of sent traffic. concretely, the handler only need to pass the packet to the method ”updateOrCreateConversation” of the Conversation Model and that will execute the flow of creating or updating hosts and conversations. A handler can capture 0 or n packet, a packet can create or update only one conversation and a conversation forcibly must contain exactly two hosts.

III.1.4 Protocol

This class is used to determine the number of packets sent for each protocol by type, this will table is the primary table for creating the protocol distribution pie chart, it contains the protocol name and the number of packets in that protocol, the data is structured in a way that makes it easy to create the chart.

III.1.5 Activity

Activity refers to an event in the system flow where data, title, description and timestamp is saved for later reference, The following are the types of activities Incorporated in the system:

- **Starting & stopping capture** upon starting or stopping the system processes the program saves the data generated by the scanning process (in case of starting) and saves it as an Activity.
- **Creating updating or deleting users** When a user is created,updated or deleted an Activity is saved with the data of the request.
- **Intrusion detected** When a handler decides that a packet or a conversation is malicious the system saves an Activity with relevant data such as the probability of the Machine learning prediction, the MAC address related and the timestamp

III.1.6 Action

In our intrusion detection and prevention application, the Action model represents the mitigation (prevention) mechanism of the system, it has a title, a description and an api_path attributes which is used to determine the API endpoint of the execution of a certain action.

III.1.7 Alert

This class represents the way that a user is notified. an Alert is either created as a consequence of a detection of an abnormal activity in the network, or as a custom alert that a user has created, it has a title, a message and a status, the status is represented by the is_treated attribute. an alert could be associated to an action. For example the alert created automatically when the system detects anomaly behavior in the ARP conversations, the alert is associated to ”De-authenticate MAC address” action.

IV Conclusion

In this chapter, we explored the sequence diagrams and class diagram associated with the key use cases in our system. The sequence diagrams provided a visual representation of the interactions and flow of events, capturing the step-by-step execution of the use cases and the exchange of messages between actors and system components. On the other hand, in the static modeling section, the class diagram offered a global view of the system’s structure, illustrating the relationships, attributes, and methods of the classes. These diagrams provide a comprehensive understanding of the system’s behavior and structure.

Chapter 4

Project realisation

I Introduction

In this chapter, we will dive into the system realization of the solution. We will explore the hardware and software environment used for the development of the system, providing an overview of the tools and technologies employed.

II Hardware development environment

For the hardware development environment of this project, I utilized a laptop as the primary machine for development and testing. The laptop is equipped with 8GB of RAM and an i5 processor, running on the Linux Ubuntu operating system. This machine served as the foundation for implementing and running the IDS/IPDS solution.

In addition to the development laptop, I employed another laptop for simulating the attack scenario. This second laptop, also equipped with an i5 processor, features 16GB of RAM and runs on the Windows operating system. To simulate the ARP poisoning attack, I set up a virtual machine using VirtualBox, running Kali Linux. This setup allowed for the safe and controlled execution of the attack scenario within a controlled environment.

III Software development environment

III.1 Back-end development environment

III.1.1 Docker

I utilized Docker to containerize the application and ensure consistent deployment across different environments.

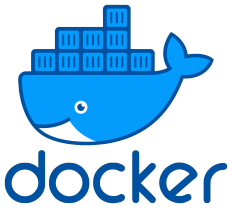


Figure 4.1: Docker logo

> Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production. — docker official docs [3]

Docker simplifies the process of managing dependencies, which is particularly beneficial for our project's requirements. The system rely on various command-line interface (CLI) tools such as tshark, arp-scan, aireplay-ng, and others for network analysis, packet capturing, and ARP poisoning mitigation processes. These tools have specific versions and dependencies that need to be installed and configured correctly. Concretely, i used Docker to encapsulate the following components:

- Linux Ubuntu: Linux Ubuntu served as the operating system for our backend server. The image is configured to included all the necessary dependencies and tools required for the development and execution of our PHP-based application. including php itself.
- Database Management System: I used Docker to encapsulate and manage a MySQL server, which served as the database management system for our backend server. Docker and docker-compose cli tool allowed me to easily configure and deploy the MySQL container, ensuring consistent database operations across different environments.

- Mercure HUB Server: Docker also played a crucial role in encapsulating and managing the Mercure HUB server within our development environment. Mercure is a publish-subscribe protocol that enables real-time communication over HTTP. By containerizing the Mercure HUB server, we could easily deploy and configure it as a separate service, facilitating server-sent events for real-time updates within the solution.

By encapsulating the application and its dependencies within a Docker container, we ensure that all the required CLI tools are available and correctly configured. This eliminates the need for manual installations and configurations on different development machines or servers. Docker's containerization allows us to create a self-contained environment with the necessary tools, ensuring consistent and reliable execution across different environments.

III.1.2 Symfony

Symfony is a powerful PHP framework that played a central role in developing the back-end server of our solution. With its robust architecture and extensive set of tools and libraries, Symfony provided a solid foundation for building scalable and maintainable web applications. The use of Symfony offered several benefits throughout the development process. First and foremost, Symfony follows the principles of model-view-controller (MVC) architecture, providing a clear separation of concerns and enhancing code organization and maintainability. This architectural pattern allowed us to structure our application's logic, views, and data models in a modular and structured manner. Additionally symfony was a great option when it comes to entity creations

Figure 4.2: Symfony logo

III.1.3 Wireshark

Wireshark is a widely-used network protocol analyzer that provides a graphical user interface (GUI) for capturing, analyzing, and interpreting network packets. It supports various protocols and offers robust filtering and analysis capabilities, making it an indispensable tool for network troubleshooting and security analysis

Figure 4.3: Wireshark logo

In order to create an accurate and efficient machine learning model, it is necessary to comprehend the underlying network protocols and behaviors, in particular ARP communications and uses. Wireshark plays a vital role in this process by capturing, collecting, and filtering data in an intuitive manner. By analyzing the captured network packets, i gained insights into the communication patterns, data payloads, and network metadata, allowing to understand and model the network behavior effectively. in the figure 4.4 below we can see the traffic capture interface, for demonstration i filtered for address resolution protocol traffic (ARP)

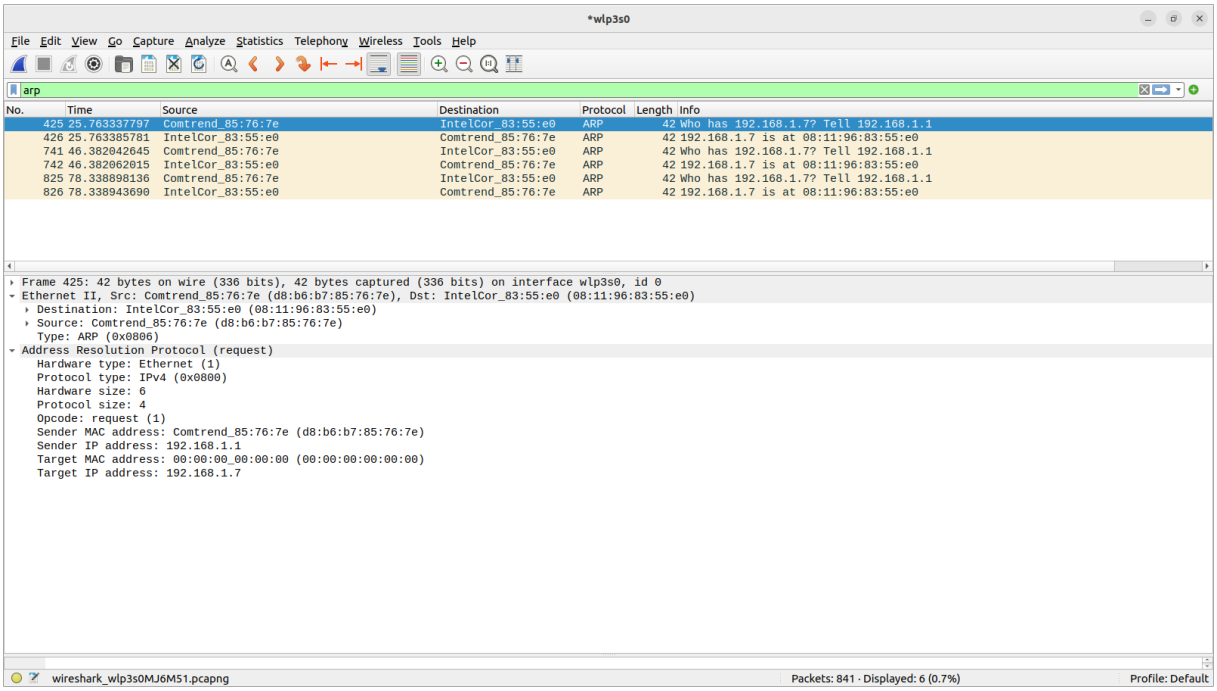


Figure 4.4: Wireshark capture interface

Filtered traffic data can be stored and transformed to formats that can be accessible with scripting languages such as Python, and then processed and cleaned. (see Data & Machine learning section)

The main goal was to analyse the attack and normal behaviour or address resolution protocol (ARP) communications and automate a way to capture - extract - and analyse the captured traffic.

Tshark

Tshark is the command-line equivalent of Wireshark, providing similar functionality in a text-based environment. As a command-line tool, Tshark offers the flexibility of automation and scripting, enabling us to perform custom analysis tasks on captured network traffic. Tshark complements Wireshark by providing a CLI interface for capturing and analyzing packets, applying display filters, and extracting information from captured data. [2] Tshark is integrated in our PHP Symfony backend using the Symfony Process Bundle, that is a bundle that helps execute system commands, tshark command in our case, each Handler in the system (see system design chapter) will be responsible of managing a separate tshark process, the bundler make easier to get the PID (process id) then if we were to use built-in exec function in PHP, this will prove handy later on in the application flow.

III.1.4 Python

Python is a programming language widely used in data processing, machine learning, and scientific computing. In our project, Python plays a crucial role in various aspects.

Figure 4.5: Python logo

Firstly, Python is used for data processing, cleaning, and transformation tasks. With its rich ecosystem of libraries such as Pandas, NumPy, and SciPy, Python provides powerful tools for manipulating and analyzing data. These libraries offer functionalities for data wrangling, handling missing values, data normalization, feature engineering, and more. Python's ease of use and expressive syntax make it an ideal choice for these data preprocessing tasks. Furthermore, Python serves as the primary language for machine learning in our project. It provides extensive libraries and frameworks, including scikit-learn, TensorFlow, and PyTorch, which enable us to build and train machine learning models. Scikit-learn, in particular, is a widely-used library for machine learning algorithms and tools.

In the project the main use of python is for searching creating testing and saving the

model that will be later used in the process of real time network traffic analysis.

III.1.5 Scikit-learn

Figure 4.6: Scikit-learn logo

Scikit-learn is a powerful and popular Python library for machine learning. It provides a wide range of algorithms and tools for tasks such as classification, regression, clustering, and dimensionality reduction. With its intuitive API and extensive documentation, scikit-learn is widely used by data scientists and machine learning practitioners. One of the key strengths of scikit-learn is its comprehensive collection of machine learning algorithms. It includes popular algorithms such as decision trees, random forests, support vector machines, logistic regression, and many more. These algorithms are implemented in an efficient and optimized manner, allowing users to train models on large datasets with ease.

Scikit-learn also offers various utilities for data preprocessing, feature selection, and model evaluation. It provides tools for handling missing data, scaling and normalizing features, and transforming categorical variables. Additionally, scikit-learn incorporates methods for model evaluation and selection, including cross-validation, hyperparameter tuning, and performance metrics.

III.2 Front-end development environment

III.2.1 React

React is a JavaScript library for building user interfaces. It allows developers to create interactive and dynamic UI components that can be easily reused and composed together. React follows a component-based architecture, where each component manages its own state and renders a view based on that state. It provides efficient updates to the UI by using a virtual DOM (Document Object Model) and performing minimal updates to the actual DOM.

Figure 4.7: React logo

React is widely used in web development for several reasons. First, it offers a declarative syntax, which makes it easier to understand and maintain the UI code. Developers can focus on describing what the UI should look like in different states, and React takes care of updating the actual UI efficiently. Additionally, React has a large and active community, which means there are plenty of resources, libraries, and tools available for developers to enhance their React applications.

TypeScript

TypeScript is a typed superset of JavaScript that adds static typing to the language. It provides compile-time type checking, which helps identify errors and bugs early in the development process. TypeScript enhances JavaScript with features like interfaces, type annotations, and strict null checks, which improve code quality, maintainability, and developer productivity. It also enables better tooling support, such as autocompletion, refactoring, and type inference.

ReactQuery & Axios

To establish communication between the Symfony backend and the React frontend, we can use RESTful APIs. Symfony provides excellent support for building RESTful APIs using its AbstractController class. We can define routes and controllers within Symfony to handle

various API endpoints and their corresponding actions.
In our React frontend, we can consume these RESTful APIs using libraries like React Query or axios. React Query simplifies data fetching and state management by providing hooks and utilities to interact with APIs. We can use React Query's useQuery, useMutation, and useQueryClient hooks to fetch data from the Symfony backend, handle mutations, and manage the cache for optimized performance.

Formik & Yup

In the frontend forms, i used Formik and Yup to simplify and enhance input validation. Formik is a popular form library that provides a simple and intuitive way to manage form state, handle form submission, and perform validation. It offers features like field tracking, form validation, and error handling, making it easier to build complex forms with minimal effort. Yup, on the other hand, is a powerful schema validation library that works seamlessly with Formik. It allows us to define validation rules and schemas for our form fields, ensuring that user input adheres to specific requirements and constraints. Yup provides a declarative and intuitive syntax for defining validation rules, making it easier to handle complex validation scenarios. in our case we specifically used Yup for validating email addresses, IP addresses and MAC addresses.

Echarts

In our React frontend, we leverage the power of Echarts to visualize and display data in an interactive and appealing manner. Echarts is a robust and flexible JavaScript charting library that offers a wide range of chart types, including bar charts, line charts, pie charts, and more. It provides a rich set of features and customization options, allowing us to create visually stunning and informative charts.

With Echarts, we can easily transform raw data into meaningful visual representations. We can configure chart options such as axes, labels, legends, and tooltips to present data in a clear and understandable way. Echarts also supports interactive features like data zooming, tooltip interactions, and chart animations, providing users with an engaging and dynamic data exploration experience.

Integrating Echarts with React is seamless, thanks to its React wrapper library called echarts-for-react. This library provides a convenient way to use Echarts in React components,

allowing us to easily import and render charts within our application. It provides React-specific features like props-based configuration and lifecycle hooks, making it easier to incorporate Echarts into our React-based project.

III.2.2 React Native

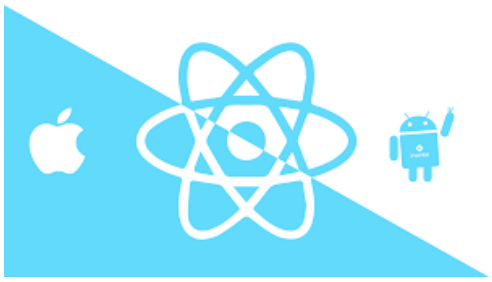


Figure 4.8: React Native logo

React Native is a framework for building native mobile applications using React. It allows developers to write mobile apps using JavaScript and leverage the same component-based architecture as React for building UI components. React Native uses native components and APIs, providing a native-like experience to users on both iOS and Android platforms. React Native allowd me to create cross-platform mobile applications with a single codebase. This eliminates the need for separate development teams or codebases for iOS and Android, saving time and effort. Additionally, the React Native community is actively growing, and there is a vast ecosystem of libraries and resources available to enhance the development process. Another benefit of using React Native is to leverage existing knowledge of React to build powerful and feature-rich mobile applications. The ability to write once and deploy to multiple platforms, along with the extensive community support, makes React Native and Expo a popular choice and my personal choice for mobile app development.

III.2.3 Expo

Figure 4.9: Expo logo

Expo is a set of tools and services that complements React Native and simplifies the development process. It provides a development environment and a range of pre-built compo-

nents, making it easier to build and deploy React Native applications. Expo also offers features like easy app sharing, over-the-air updates, push notifications, and access to device hardware through a unified API.

IV Machine learning case study

Machine learning is well-suited for ARP spoofing detection due to its ability to analyze and identify complex patterns in network traffic data.

IV.1 Data collection

In order to generate non-malicious Address Resolution Protocol (ARP) data, i used Wireshark, by capturing network traffic containing ARP packets between different hosts. This data was then labeled as non-malicious, as it represented normal and legitimate network communication. Then the task was to simulate an attack scenario, we intentionally generated malicious ARP traffic. By utilizing various tools and techniques, such as ARP spoofing using kali linux ettercap, we manipulated the ARP packets to exhibit malicious behavior. This simulated attack traffic was then captured and saved along with the relevant data. To distinguish between the non-malicious and malicious ARP data, we applied appropriate labels. The non-malicious data was labeled as ”non-malicious” to indicate its benign nature, representing legitimate network communication. Conversely, the manipulated ARP traffic generated during the attack simulation was labeled as ”malicious”.

IV.2 Data Exploration

The dataset used for the analysis contains the following columns:

- `src_ip`: Source IP address.
- `src_mac`: Source MAC address.
- `timestamp`: Timestamp of the network activity.
- `ab_number_packets_transmitted`: Number of packets transmitted from source A to destination B.
- `ba_number_packets_transmitted`: Number of packets transmitted from source B to destination A.

- ab_total_packets_length: Total length of packets transmitted from source A to destination B.
- ba_total_packets_length: Total length of packets transmitted from source B to destination A.
- total_arp_duplicate_detected: Total number of ARP duplicates detected.
- total_arp_duplicate_frame: Total number of ARP duplicate frames.
- seconds_since_last_duplicate_arp: Time elapsed in seconds since the last duplicate ARP.
- duration_seconds: Duration of the network activity in seconds.
- ab_number_arp_packets: Number of ARP packets transmitted from source A to destination B.
- ba_number_arp_packets: Number of ARP packets transmitted from source B to destination A.
- host_a_number_conversations: Number of network conversations involving source A.
- host_a_is_private: Boolean indicating if source A is a private host.
- host_b_number_conversations: Number of network conversations involving source B.
- host_b_is_private: Boolean indicating if source B is a private host.
- last_packet_arp_opcode: Last ARP opcode observed in the network activity.
- seconds_since_last_arp_from_current_src: Time elapsed in seconds since the last ARP packet from the current source.
- is_malicious: Binary label indicating if the network activity is malicious or not.

During the data exploration phase, we conducted a simple analysis to understand the distribution of the target variable, which indicates whether the network activity is malicious or not. To visualize this, we created a histogram using the matplotlib library.

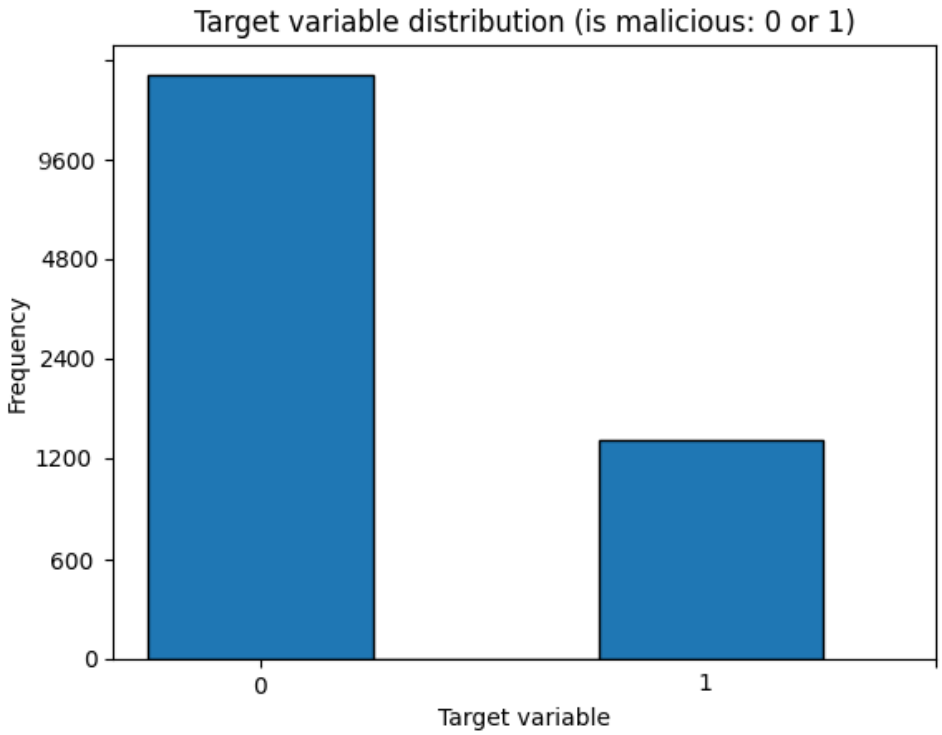


Figure 4.10: Target attribute distribution

IV.3 Data Processing

The preprocessed data was further processed as follows:

- Splitting the data into a training set and a testing set.
- Converting the `timestamp` column to the datetime data type.
- Extracting relevant features from the `timestamp` column, such as the hour, minute, and second.
- Dropping the original `timestamp` column from the dataset.
- Encoding the categorical columns using one-hot encoding to represent them as numerical binary features.
- Converting the boolean columns (`host_a_is_private` and `host_b_is_private`) to integer binary values.
- One-hot encoding the character column (`last_packet_arp_opcode`).
- Normalizing the remaining (numerical) columns.

The resulting training set (`X_train`) contains n samples and m features, while the testing set (`X_test`) contains p samples and m features, where m represents the total number of features.

IV.4 Model selection

Before choosing the model that will be integrated in the system, i conducted an evaluation of machine learning libraries to select the most suitable model for our task. Specifically, i explored two popular libraries: TensorFlow and Scikit-learn. the comparison must be based on factors such as model architecture, ease of implementation and performance. and because our system has a high velocity data source (internet traffic) the performance factor is the dominant one.

IV.4.1 TensorFlow Neural Network models

For the TensorFlow implementation, we utilized a Long Short-Term Memory (LSTM) model architecture. The defined model consists of two LSTM layers with 50 and 100 units, respectively. To mitigate overfitting, a dropout layer with a rate of 0.2 was added after each LSTM layer. The final layer is a Dense layer with a sigmoid activation function, producing a binary output.

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.LSTM(50, input_shape=(X_train.shape[1], 1))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.LSTM(100, return_sequences=False))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(1, activation='sigmoid'))
```

During the evaluation of the LSTM model we fitted the model for 100 epochs and employed accuracy and loss metrics. The accuracy chart presented the model's prediction accuracy across different epochs, providing valuable insights into its learning progress over time.

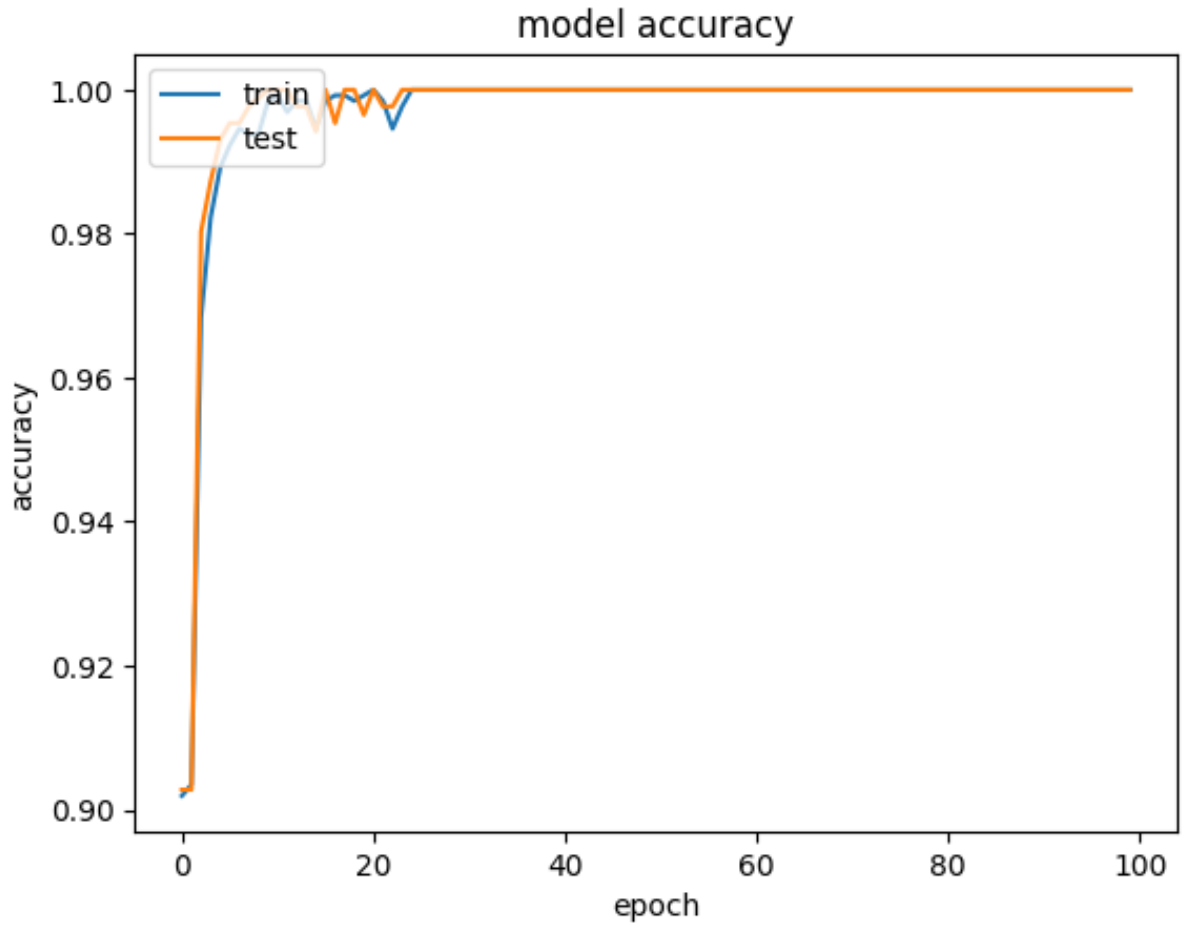


Figure 4.11: LSTM Model accuracy graph

In this case, the chart indicates that the model's accuracy started at around 90% in the first epoch and steadily improved over subsequent epochs. By epoch 10, the accuracy reached a high level of approximately 99-100%. This demonstrates that the model was able to successfully learn the patterns in the data and make highly accurate predictions. The fluctuations observed in the accuracy chart, ranging from 0.1% to 0.2% for a few epochs, indicate some minor variations in the model's performance during training. However, after these fluctuations, the accuracy remained constant at 100%, suggesting that the model achieved a perfect accuracy in predicting the target variable.

Furthermore, we examined the loss chart, which illustrated the training and validation loss of the model over epochs. This chart allowed us to evaluate the convergence of the model and identify any signs of overfitting or underfitting.

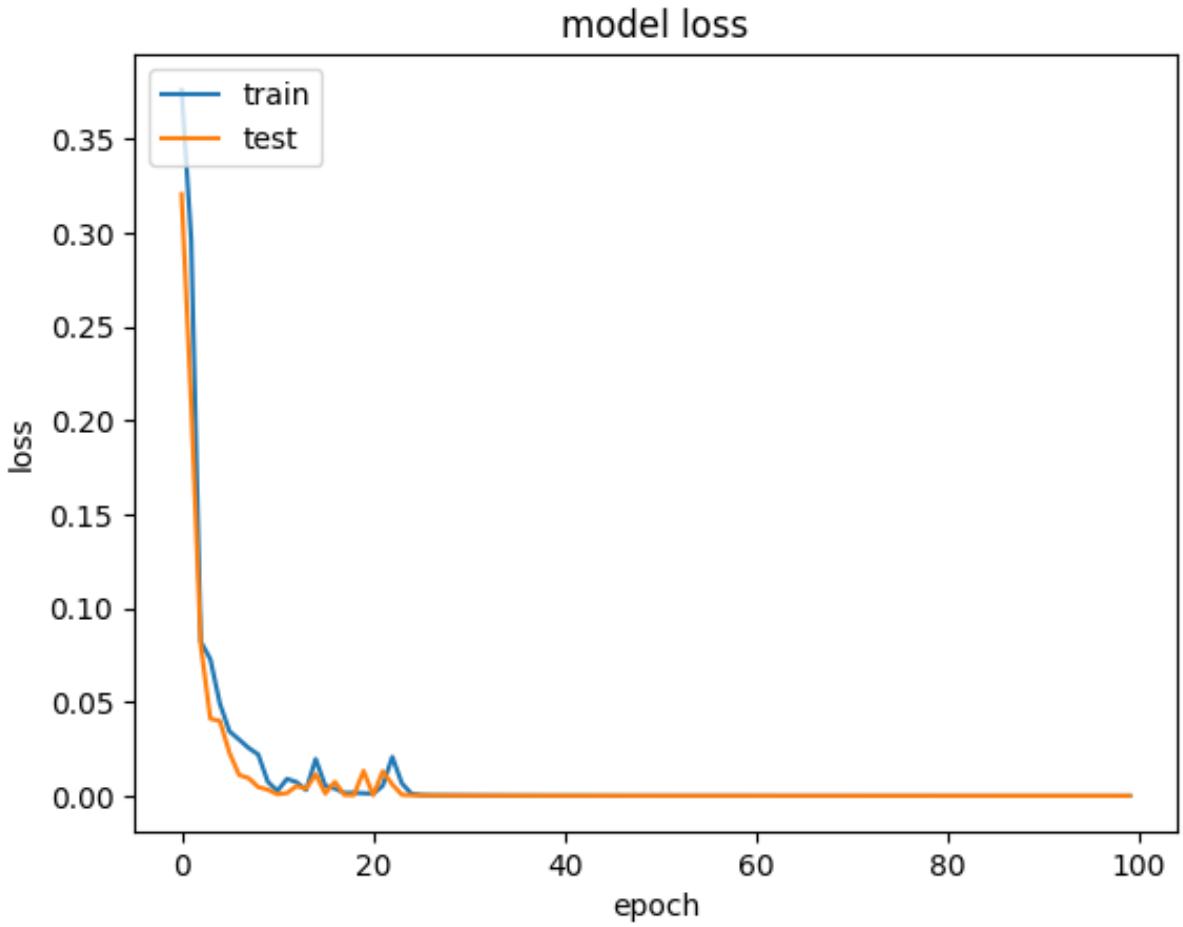


Figure 4.12: LSTM Model loss graph

The loss started near 0.35 and gradually decreased as the model learned from the data. The fluctuations observed between epochs 1 and 20 indicate minor variations in the loss during training. As the loss approached 0, it remained constant at 0, indicating that the model achieved a very low loss and effectively converged. This implies that the model was able to capture and represent the underlying patterns in the data accurately, resulting in minimal errors in its predictions.

Based on the evaluation, the LSTM model implemented using TensorFlow demonstrated promising results in terms of accuracy and score. The loss chart indicated a decreasing trend, indicating that the model successfully learned the underlying patterns in the data.

Based on the observation that the model had already converged by epoch 25-30, it suggests that further training beyond this point may not significantly improve the model's performance. Lowering the number of epochs can be considered to better visualize and analyze the model's behavior during the early stages of training. By reducing the number of epochs, we can focus on the initial learning process and assess how the model rapidly improves its accuracy and reduces the loss. This can provide valuable insights into the early convergence behavior and allow for more efficient training, especially if the convergence is already achieved within a

relatively small number of epochs. the following is the updated chart with only 25 epochs:

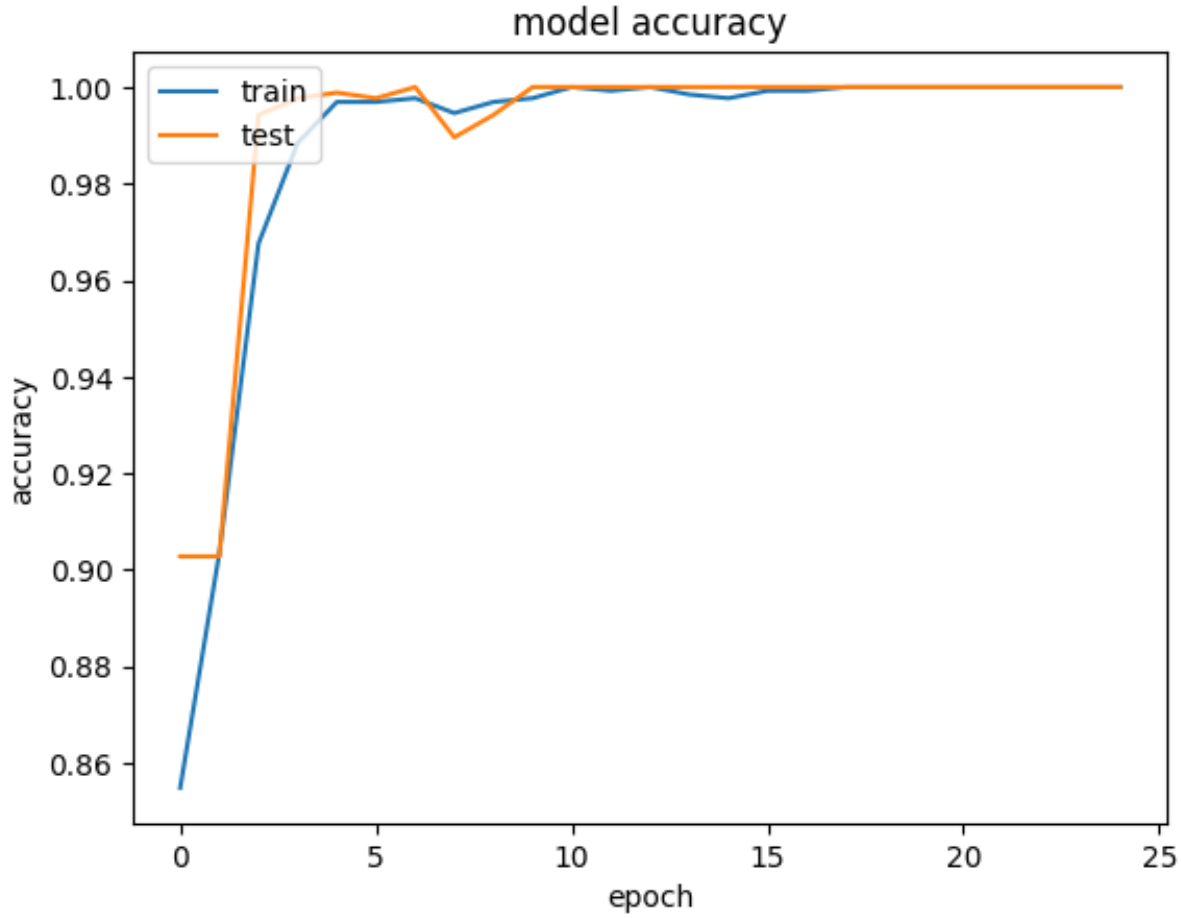


Figure 4.13: LSTM Model accuracy graph (2)

- **Performance considerations** One drawback of using TensorFlow is its performance when running on systems without a GPU. Compared to other simpler algorithms, TensorFlow can be relatively slow, especially when dealing with large datasets or complex models. The reliance on GPU acceleration for efficient computation is a limitation that is more aggravated when dealing with a high velocity data.

  Furthermore, if alternative models demonstrate similar classification performance to TensorFlow, it may be more advantageous to choose those models over neural networks. The computational overhead and infrastructure requirements associated with deep learning frameworks like TensorFlow might not be justified if simpler algorithms can achieve comparable results.

IV.4.2 Scikit-learn models

For the Scikit-learn models, we focused on model selection and pipeline creation. One of the key advantages of Scikit-learn is its extensive collection of machine learning algorithms

and tools. To ensure a streamlined and efficient workflow, we leveraged the concept of pipelines.

A pipeline in Scikit-learn allows us to assemble multiple steps, including data preprocessing and model training, into a single unified workflow. It ensures that the data flows seamlessly through each step, enabling consistent and reliable processing. Within the pipeline, we performed column selection based on data type and applied appropriate transformations, such as categorical encoding for string columns and standard scaling for numeric columns.

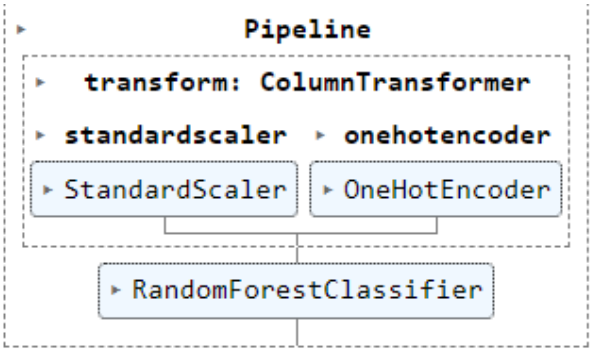


Figure 4.14: Example of a pipeline

The use of pipelines simplifies the code by encapsulating the entire workflow into a single object. This not only enhances code modularity and reusability but also ensures the integrity and consistency of the data processing pipeline during cross-validation and model selection.

In addition to pipelines, we utilized the concept of grid search to fine-tune the hyperparameters of the models. Grid search allows us to systematically explore various combinations of hyperparameters and identify the best parameter configuration for optimal model performance.

To evaluate the performance of different models, we conducted experiments using various Scikit-learn classifiers, such as K-Nearest Neighbors (KNN), Multi-Layer Perceptron (MLP), Random Forest, Support Vector Machine (SVM), and Logistic Regression. Each model was evaluated using appropriate performance metrics, such as accuracy, and the results were recorded for comparison.

To present the findings of the model evaluation, we created a table that summarizes the performance metrics and corresponding parameter configurations for each model. The table provides a comprehensive overview of the performance achieved by each model, allowing for an informed decision-making process.

Model	Score	Parameters
0	0.9841	weights=uniform, n_neighbors=1, leaf_size=211, algorithm=ball_tree, model=KNeighborsClassifier(...)
1	0.9864	solver=lbfgs, learning_rate=constant, hidden_layer_sizes=(560, 50), activation=relu, model=MLPClassifier(...)
2	1.0000	n_estimators=121, max_features=sqrt, criterion=entropy, bootstrap=True, model=RandomForestClassifier(...)
3	0.9856	shrinking=True, probability=True, kernel=rbf, gamma=scale, coef0=2, C=41, model=SVC(...)
4	0.9803	solver=saga, multi_class=ovr, C=1, model=LogisticRegression(...)
5	1.0000	n_estimators=1, model=AdaBoostClassifier(...)

Based on the evaluation results, it was observed that the Random Forest classifier achieved a perfect score of 1.0, indicating excellent classification performance. However, the AdaBoost classifier also achieved a perfect score. In this scenario, considering the computational resources required, it was decided to choose the Random Forest classifier over AdaBoost due to its potentially lower resource consumption.

IV.4.3 Decision

After evaluating the TensorFlow LSTM model and comparing it with various Scikit-learn models, several factors were considered to make an informed decision.

Firstly, the performance of the TensorFlow LSTM model was assessed in terms of accuracy and score. The accuracy chart showed a steady increase from 90% at the first epoch to nearly 100% by epoch 10. The model achieved consistent accuracy of 100% thereafter. Similarly, the loss chart demonstrated a decreasing trend, indicating successful learning and convergence of the model.

On the other hand, the Scikit-learn models were evaluated using a grid-search. Various classifiers, including K-Nearest Neighbors, Multi-Layer Perceptron, Random Forest, Support Vector Machine, and Logistic Regression, were compared based on their performance metrics.

Finally, based on the performance metric we choose to implement the Random forest classifier and test it in other network enviroments.

V Principal graphical user interfaces

In this section, we present the principal graphical user interfaces (GUIs) developed for the application. We will highlight the most important interfaces and provide brief descriptions of their content and functionality. we will proceed to discuss the individual graphical user interfaces and their respective descriptions.

V.1 Authentication

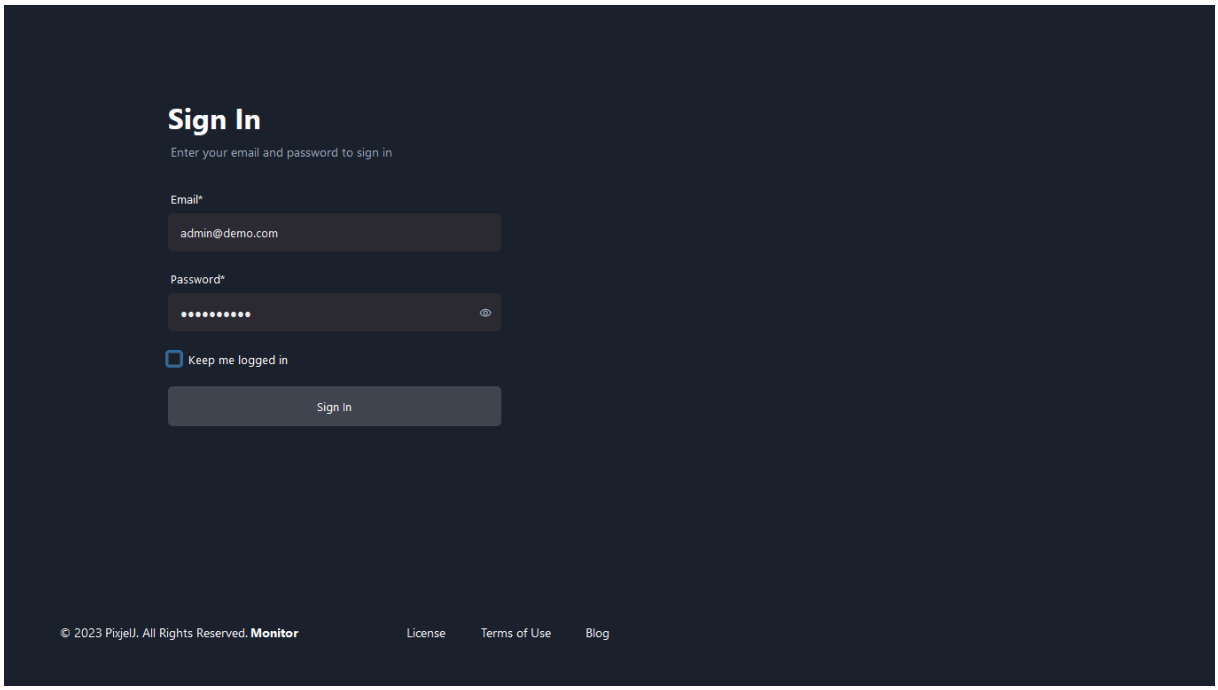


Figure 4.15: Authentication screen

Like any application, the application screens and functionalities are protected with a authentication page, the user must provide correct credentials to obtain a token that will use later for the RESTful API consumption and ressouce authorization.

V.2 Home Dashboard

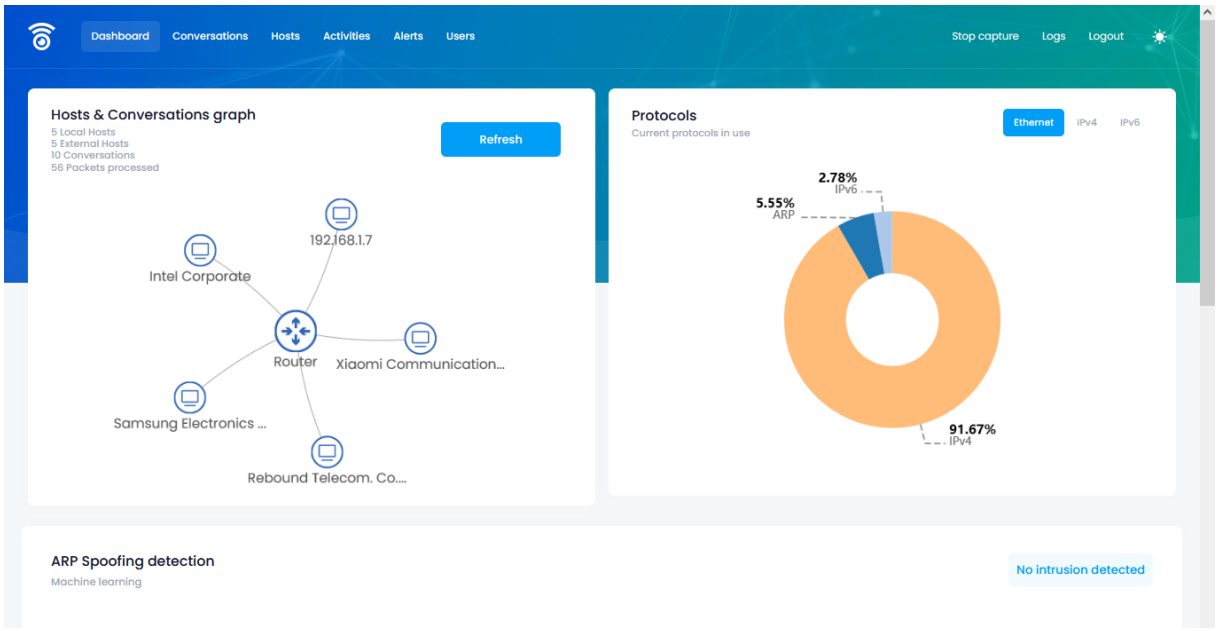


Figure 4.16: Home dashboard screen (1)

The application layout consist of a top bar containing on the left side menu links to the pages available in the system such as conversations, hosts, activities,alerts and users.
while on the right side of the top bar contain actions buttons such as the start capture button which starts the process of network traffic capturing. and other utilities such as to logout and to open logs side panel. The main part of the dashboard page contain several charts and graph that give an overview of the state of the network, these graphs includes:

- **Local connected hosts graph** this graph is basic version of the conversations graph shown in the conversations page, it shows only local connected devices (hosts) and display traffic informations such as the ip address and total trasmitted and recieved size of network traffic

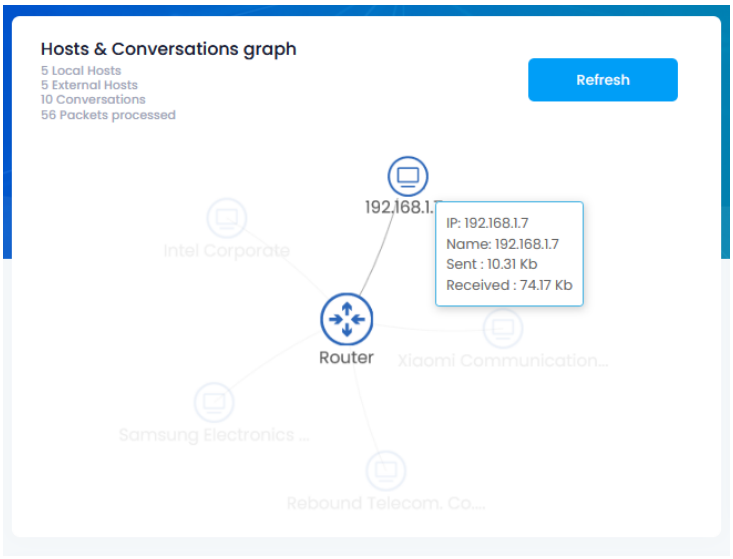


Figure 4.17: Basic conversations graph chart

- **Protocol distribution charts** User can switch between these three chart using the top right buttons of the chart
  - **Ethernet protocol distribution pie chart** this charts give a visual insight on the Ethernet level protocols used in the network, typically these protocols are IPv4 , IPv6 and ARP.

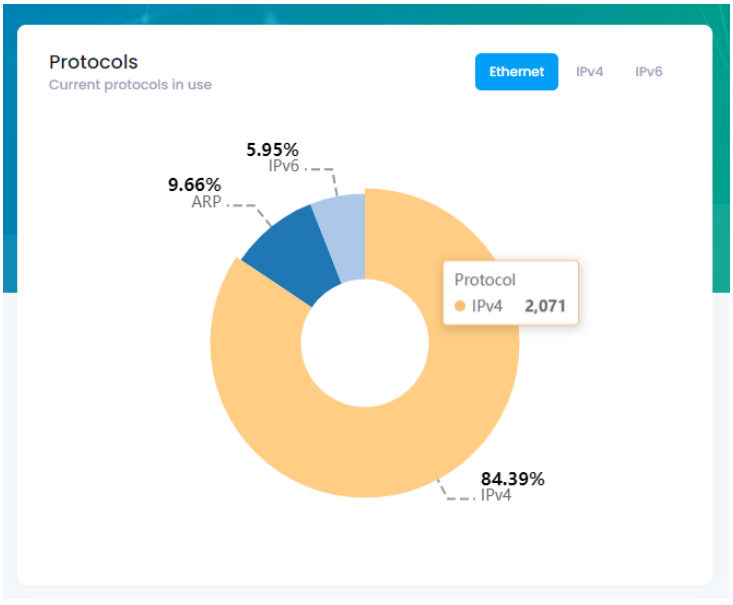


Figure 4.18: Ethernet protocol distribution Pie chart

  - **IPv4 protocol distribution pie chart** this charts give a visual insight on the IP level protocols used in the network,especially IPv4 traffic, this chart is divided into transport layer protocol names or ports and application layer protocols or ports. the

protocols taken into consideration are the protocols cited in the functional requirements of this project, any other protocol will display as a port number. for example in the graph below the port ”5222” is not registred as a protocol so it shows as the protocol number

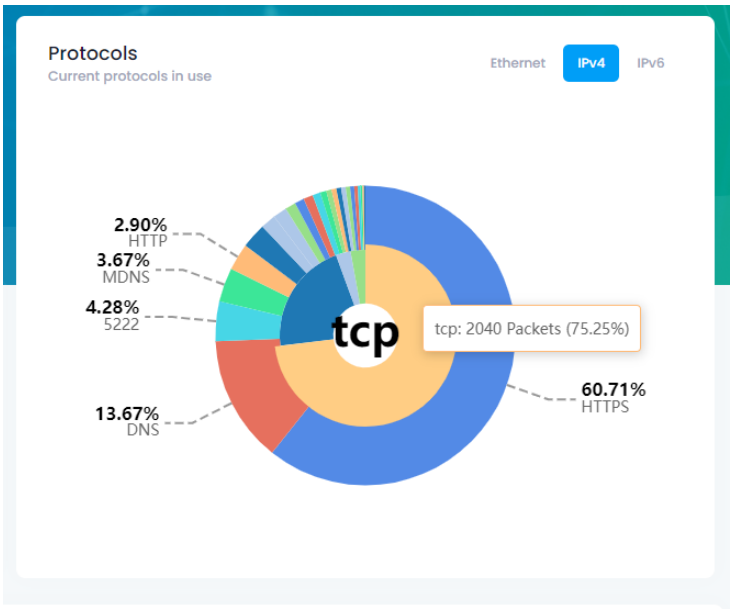


Figure 4.19: IPv4 protocol distribution Pie chart

– **IPv6 protocol distribution pie chart** similarly to IPv4 chart this chart also divided into transport and application layer protocol

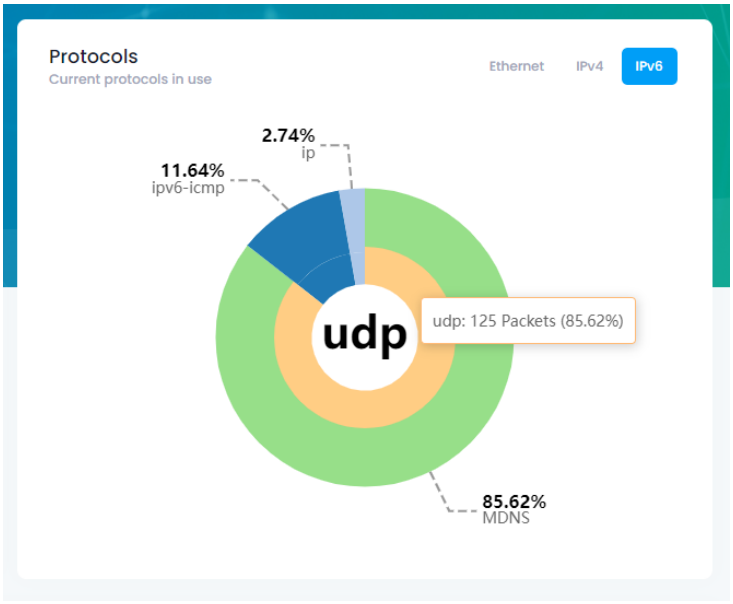


Figure 4.20: IPv6 protocol distribution Pie chart

- **Machine learning predictions multi-line chart**

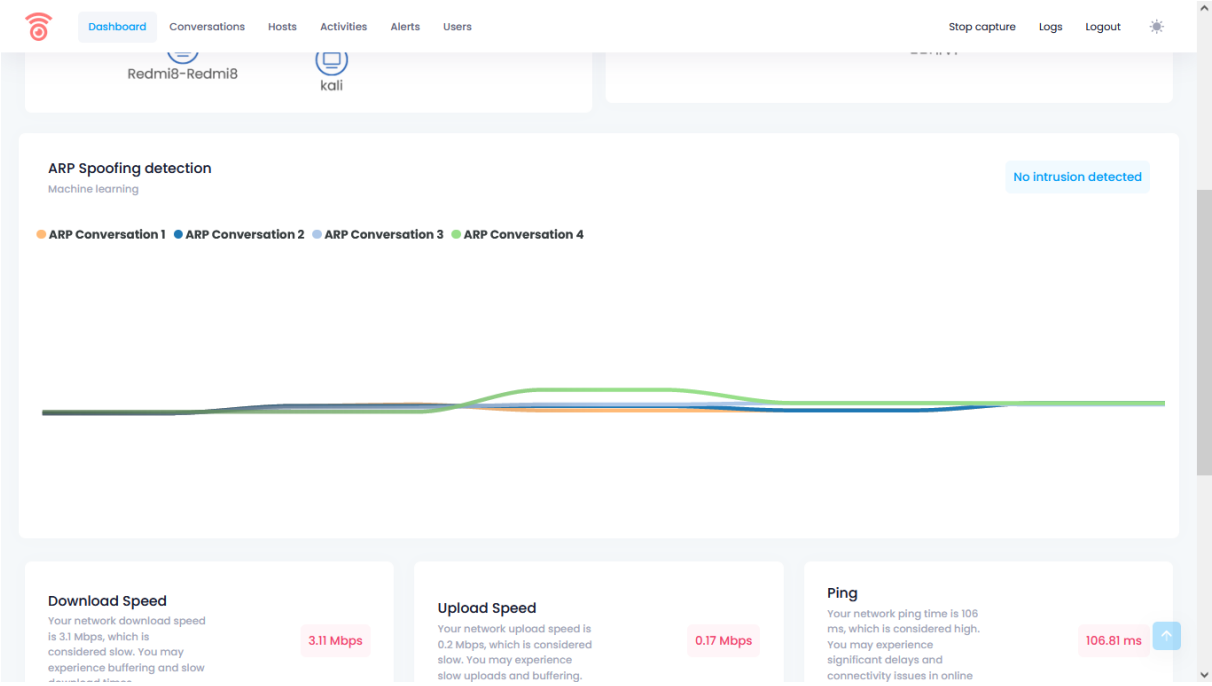


Figure 4.21: Home dashboard screen (2)

The chart shown in this part of the dashboard is a critical chart in the system as it shows the machine learning prediction over time and by ARP conversation identifier, this is an intuitive way to directly see if there any ARP attacks in the network, this chart updates in real time with each ARP packet captured by the network interface of the system.

Figure 4.22: ARP poisoning machine learning mutli-line chart

- **Network speed line charts**

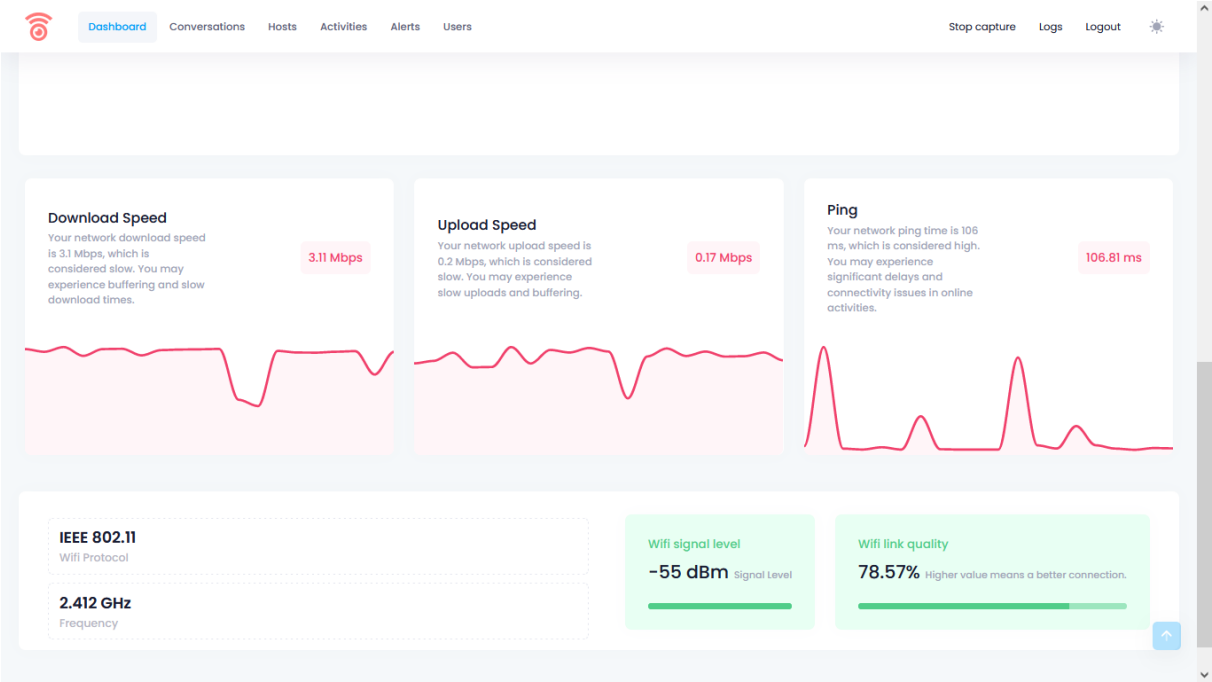


Figure 4.23: Home dashboard screen (3)

Finally for the dashboard page we have also a third part in the bottom of the page containing graphs and information about the network speed and latency, and also some basic information such as WiFi frequency and link quality as shown above. Each of the download speed, upload speed and ping latency has a custom message and color to set depending on whether the internet connection speed is slow, normal or fast.

V.3 Conversations

The ”Conversations” page provides users with a comprehensive view of their interactions and conversations. By clicking on the ”Conversations” link in the menu, users can access this page and explore additional features and functionalities. It serves as an extension of the basic graph displayed on the main dashboard, offering users a more detailed and interactive experience. Here, users have the ability to filter the conversations based on various criteria, allowing for a more focused analysis. The tabular format presents the conversation data in a structured manner, making it easier for users to review and analyze the information. This page acts as a valuable resource for users to gain insights, track trends, and make informed decisions based on the data available.

V.3.1 Nodes & edges graph

The conversations graph is considered as a very important aspect of the project as it provide a simple yet effective and beautiful way of visualizing the network topology of the network.

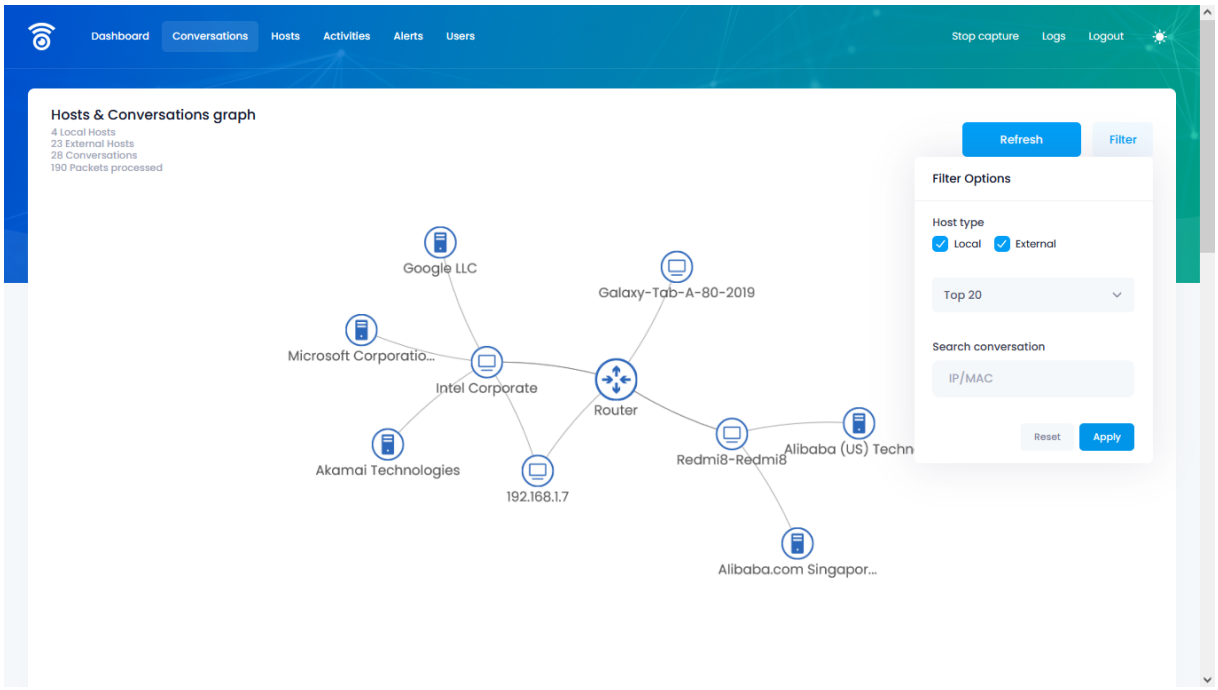


Figure 4.24: Conversations screen (1)

The filter popup is a useful tool that the network administrator can utilize to filter for IP addresses, MAC addresses , or even Host names. use can also choose whether he want to see only local hosts (devices that are internally connected to the router or Access point.) or external hosts or both. on top of that user can also select the number of conversations to show in the graph, the API with filter and group by the total size of the conversation, effectively selecting top 5 or 10 or 20 conversations in the network. beside that user can also click on a node to dynamically display the connections of that node both external and local. this help enhancing UX (user experience) by reducing the amount of clicks, instead of the user writing the ip address of the node that he wants to filter for, he just clicks on it.

V.3.2 Table of conversations

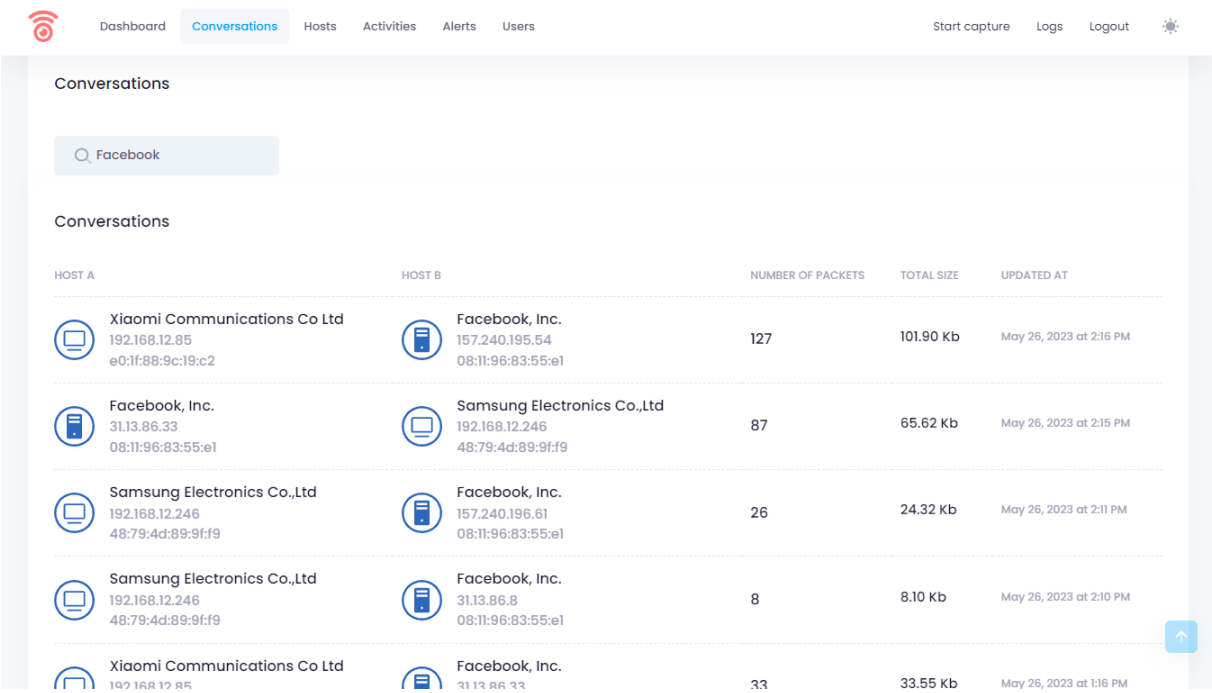


Figure 4.25: Conversations screen (2)

alongside with the graph the user also is provided with a tabular component that holds the information on the conversations, where user can search a conversation by IP, MAC or host name. Although one might argue its is the same data displayed twice, it is in fact not, as the graph chart will group the nodes by host name , displaying only the connections having a unique domain name, the table displays all the conversations as they are. one obvious example to demonstrate and explain the diffrence is a client that is connected to mutiple IP addresses of the same domaine, say ”facebook.com”, the chart will show only one Facebook node, but in the table there will be all the ”facebook.com” servers (diffrence being the ip address)

V.4 Activities

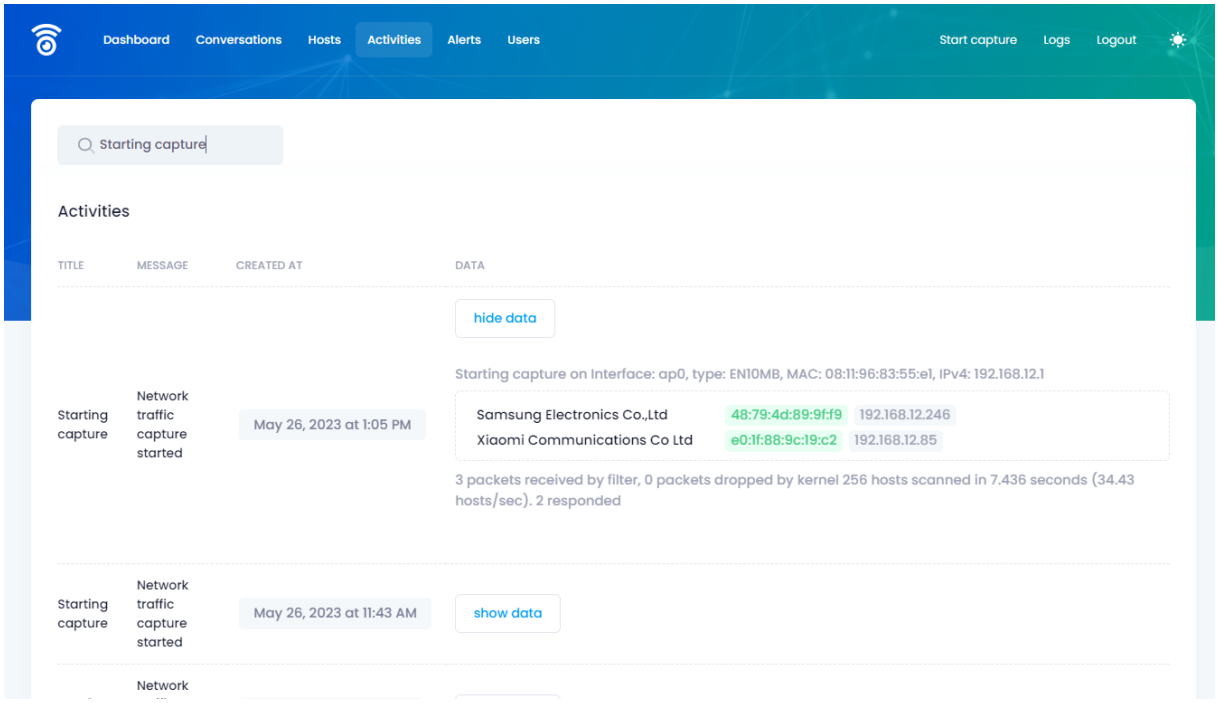


Figure 4.26: Activities screen

In this page we have the list of activities, by design user cannot edit nor delete an activity , only view and search. An activity could have a data attribute, in that case a user can click show data to show the data of that activity.

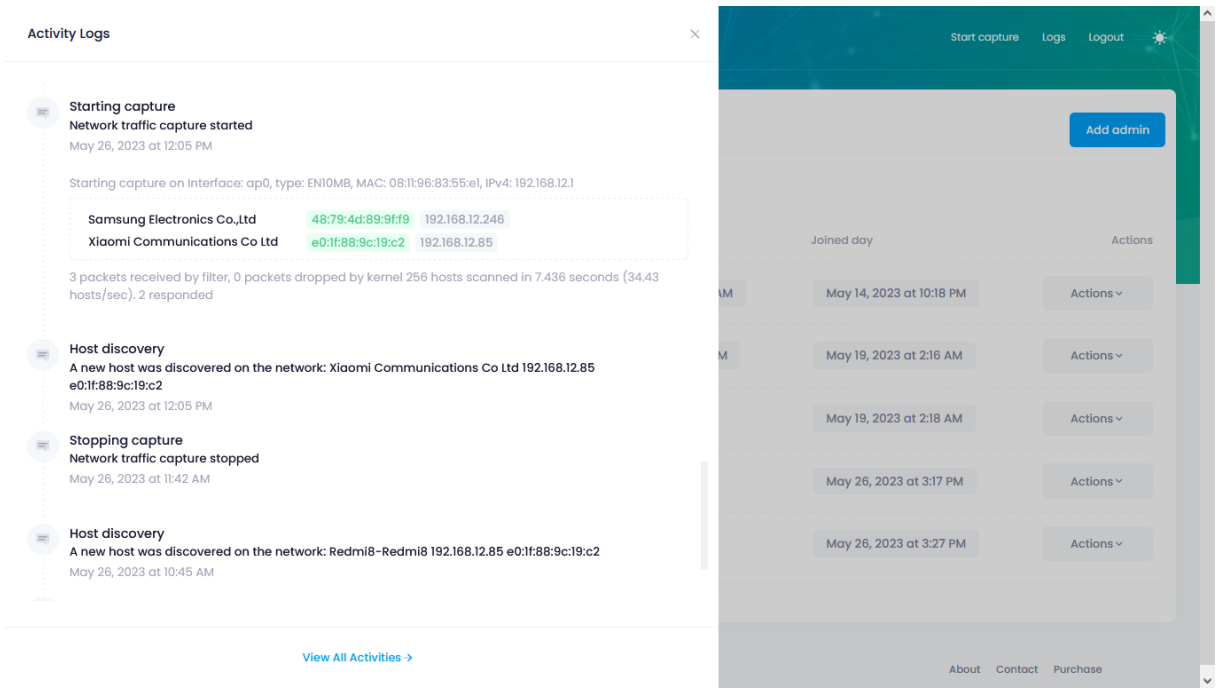


Figure 4.27: Logs slider

the activities are also visible from within the log slider, to see the logs side-panel, user must click the Logs button on the top right of the web application. it's a convenient way to access activity logs from any part of the application

V.5 Alerts

The alerts section of the dashboard is a way for the network or super administrator to manage alerts, including inspecting, updating the status executing action and creating new custom alerts. there is mainly two types of alerts:

- **Intrusion detection alerts** alerts that are created automatically when a handler has detected an attack.
- **custom alerts** alerts that are created by users

A user can either ignore the alert: setting it to treated status, or execute the action of the alert. if an alert doesn't admit an action, the button doesn't appear.

When a user wants to create an alert a form appears with the fields to type, such as the title of the alert and the message, also the user must choose if the alert is associated to an action, if so a mac address field shows. additionally the user has the option of notifying other admins about this alert, if this is checked, the admin that are connected through the mobile application receive a push notification having the title and message that the creator of the alert had specified.

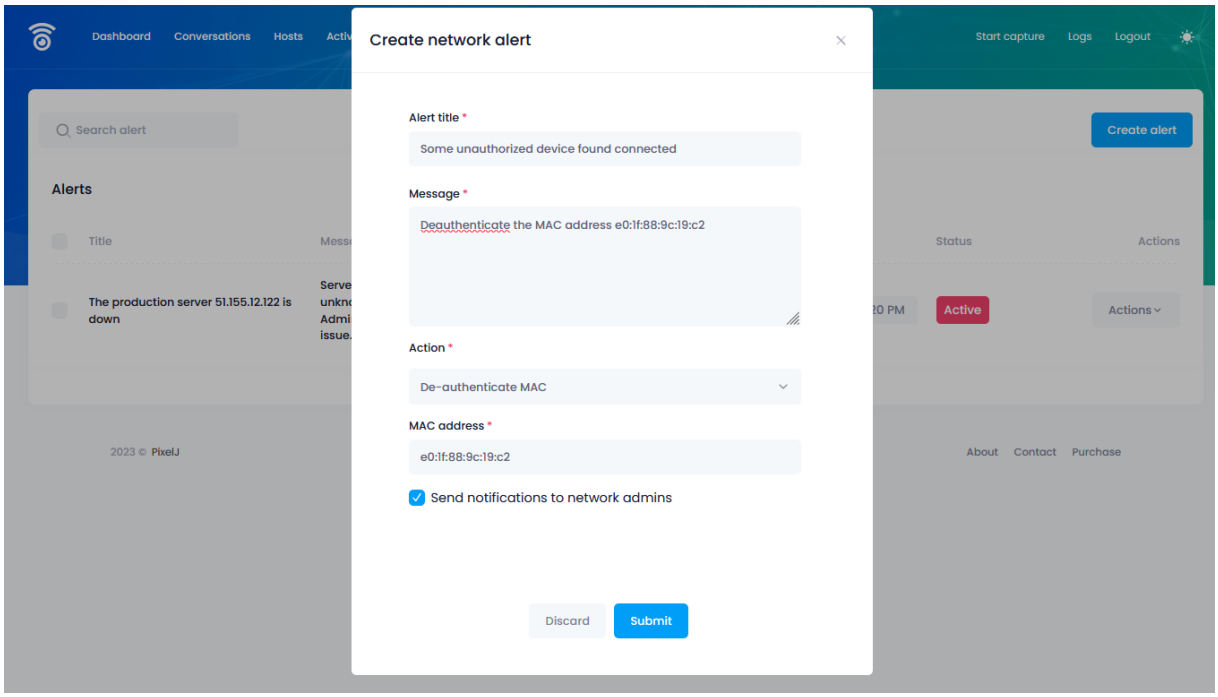


Figure 4.28: Alerts page

V.6 Users

This section is only available for super administrator, its the page where user can manage users in the system, including view,create, update and delete users.

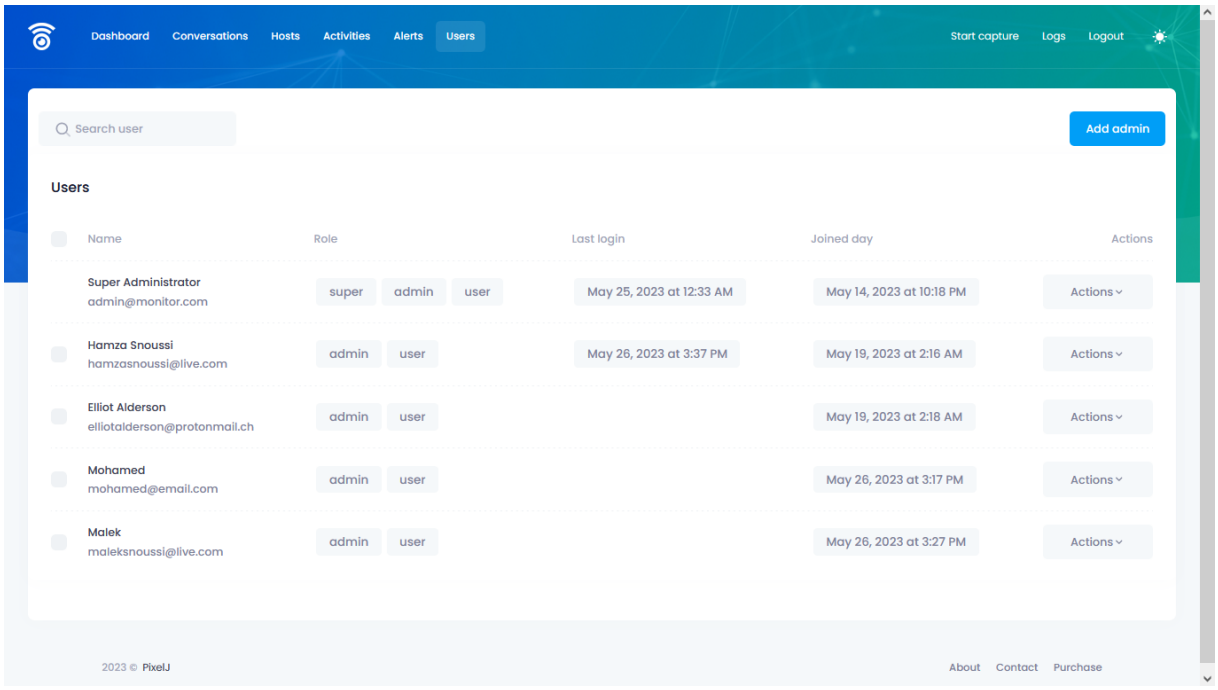


Figure 4.29: List Users

The table of the users contain

- **User info** Basic user info such as email address and username
- **Roles** the roles of the user
- **last login** The last time the user logged in into his account

The super admin is the one responsible and capable of updating user passwords.

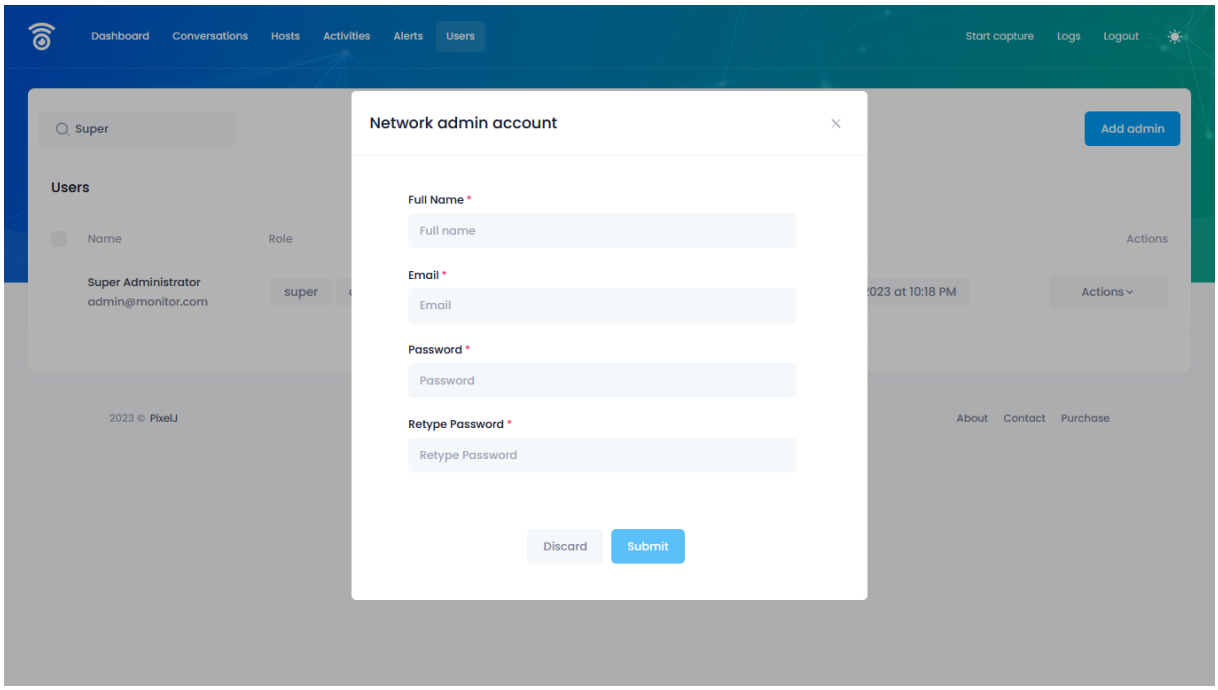


Figure 4.30: Create User (1)

The forms are controller and validated.

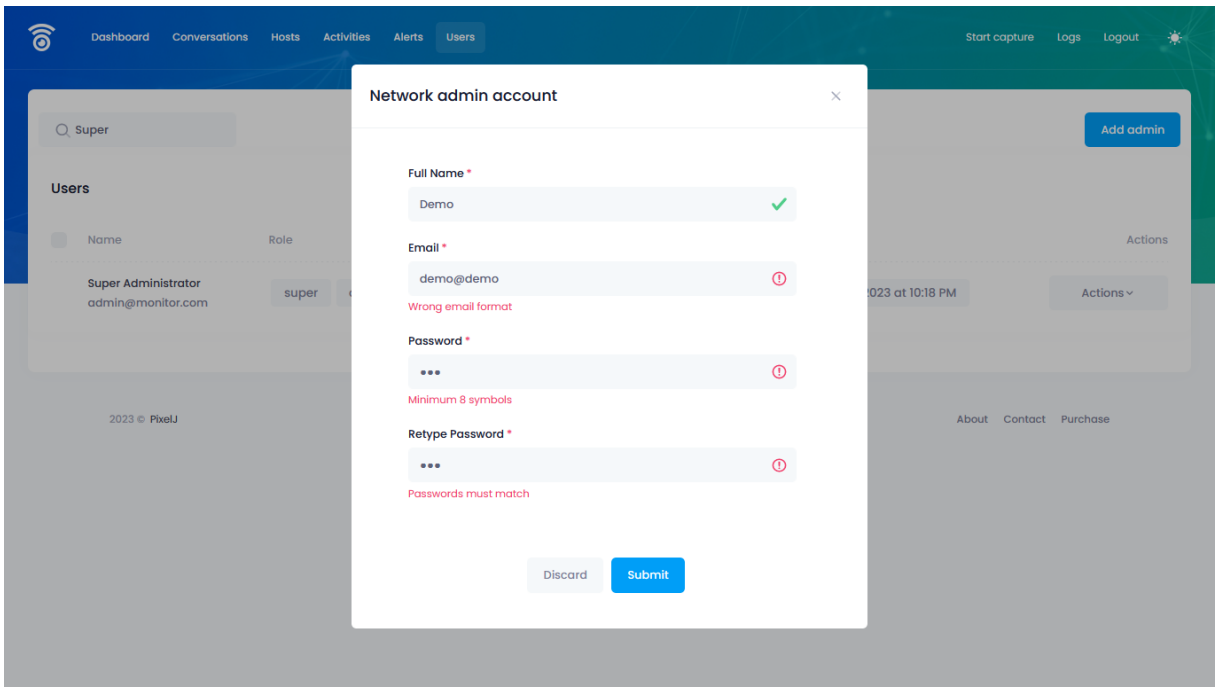


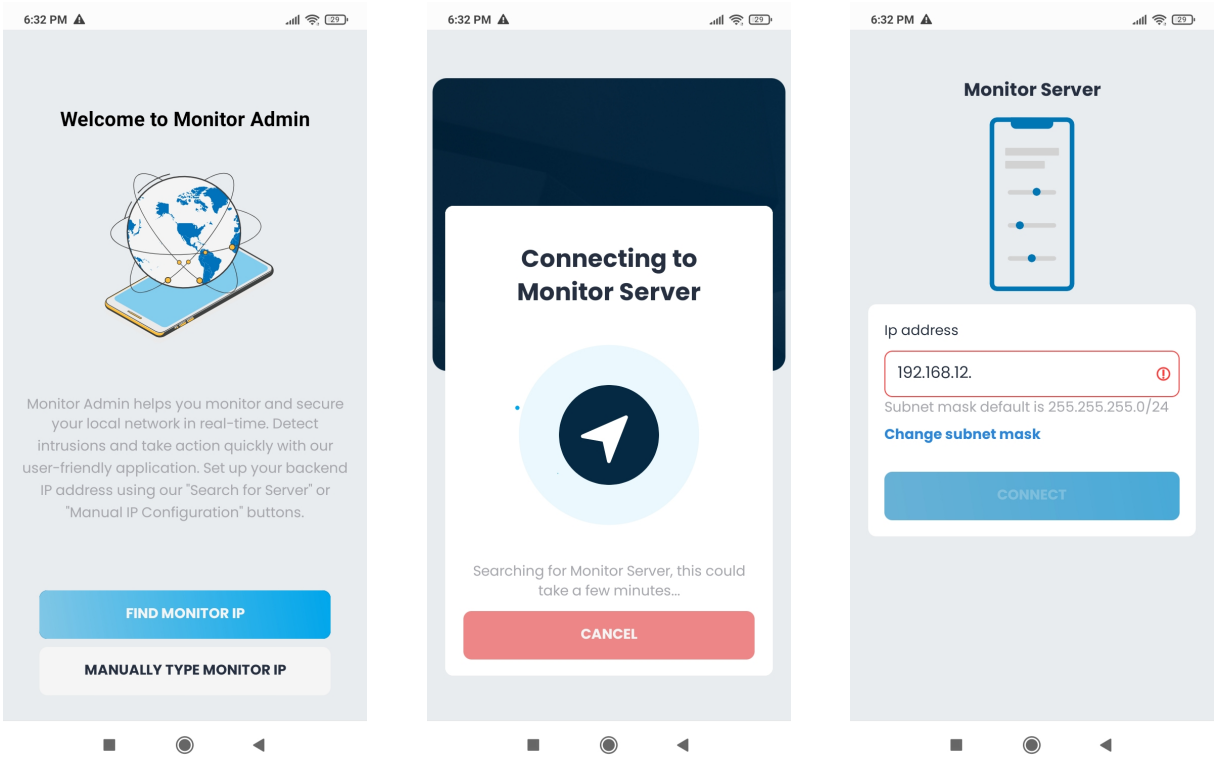
Figure 4.31: Create User (2)

V.7 Mobile application

The mobile application is a minimal version of the web application, it has minimal features in comparing the web application, in the same time it serves as a better way to receive

notifications and confirm action as we all know that mobile applications are more solicited than web application.

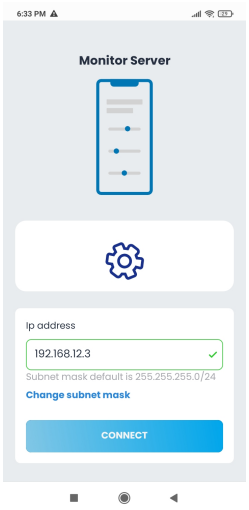
V.7.1 On-boarding & configuration



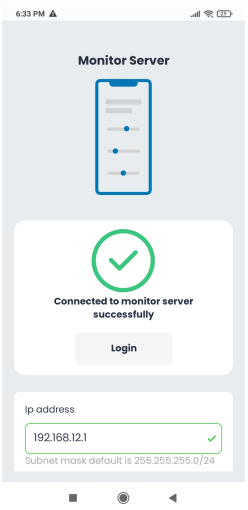
(a) Screen Shot 1 (b) Screen Shot 2 (c) Screen Shot 3

Figure 4.32: Mobile App On-boarding Screens (1)

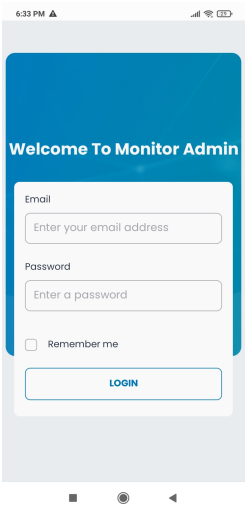
These on-boarding screens contain a necessary configuration that user must configure upon installation which is to find the monitor back-end server, the user have the choice of manually typing the ip address of the back-end server, or let the an automated program search for the server by trying each host in local network until the application finds the expected response. When succeeded the app navigates to the login screen.



(a) Screen Shot 1



(b) Screen Shot 2

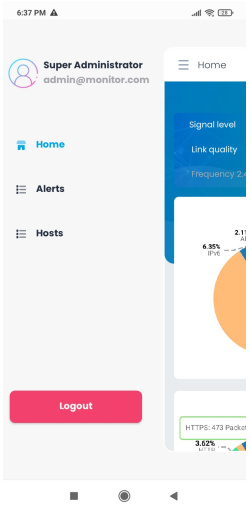


(c) Screen Shot 3

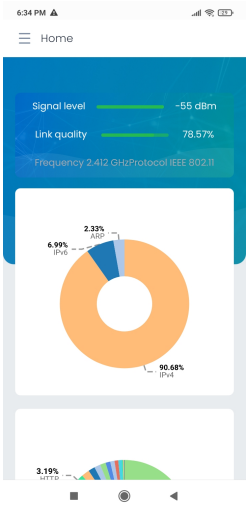
Figure 4.33: Mobile App On-boarding Screens (2)

V.7.2 Home screen

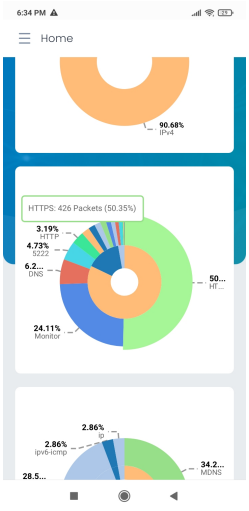
The home component has a lot of similarities with the home of the web application, it contain the graphs related to the protocol distributions and network speed metrics.



(a) Screen Shot 1

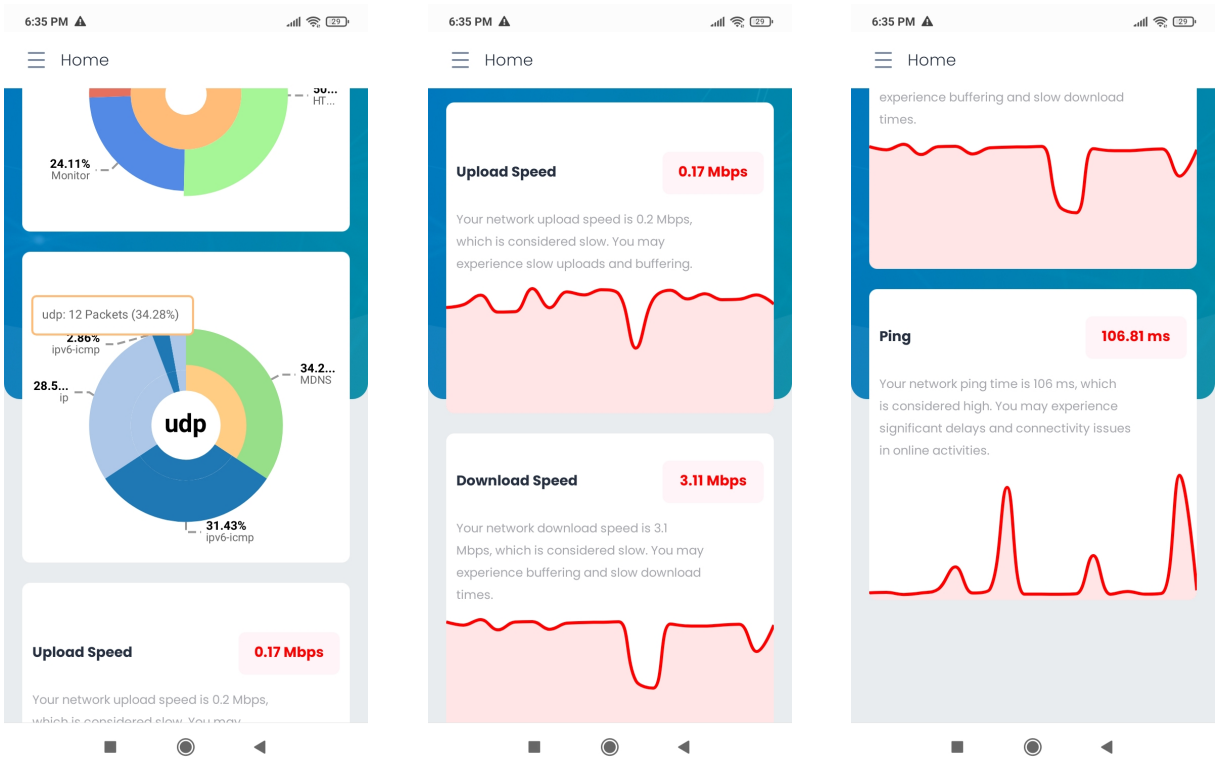


(b) Screen Shot 2



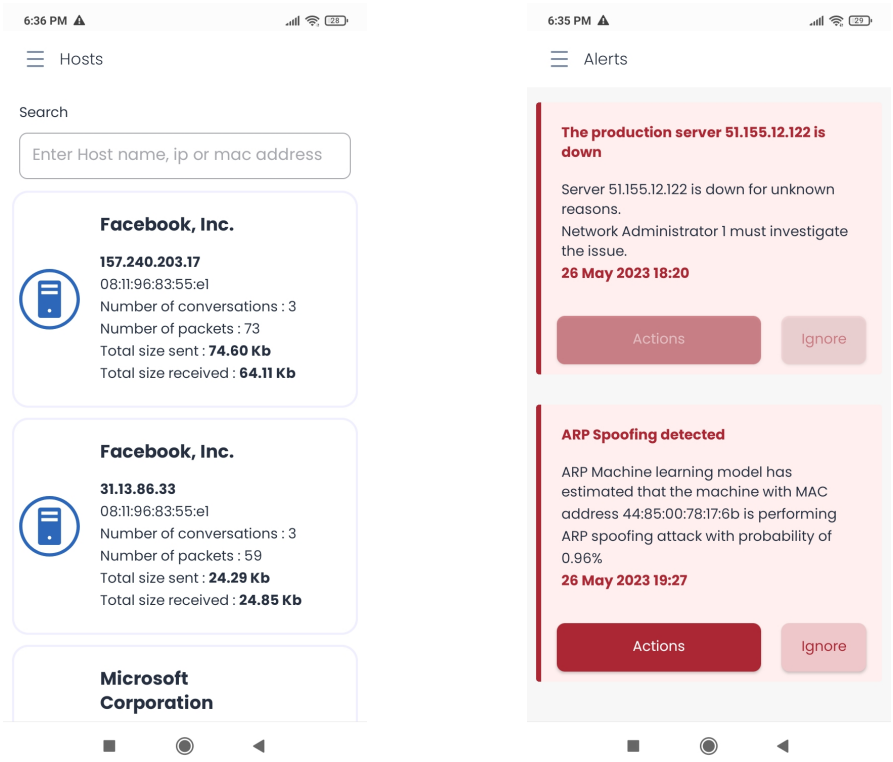
(c) Screen Shot 3

Figure 4.34: Mobile App Home Screen (2)



(a) Screen Shot 1 (b) Screen Shot 2 (c) Screen Shot 3

Figure 4.35: Mobile App Home Screen (2)



(a) Screen Shot 1 (b) Screen Shot 1

V.7.3 Hosts screen

The mobile user as also access to a hosts page where he can view and seach hosts and retrive information such as IP and MAC address, number of conversations, and volume of traffic generated in the network.

V.7.4 Alerts screen

The main feature of the mobile application is to be notified upon abnormal traffic detection, and allow user to execute action with a simple button click and confirmation. here we present the alerts page containing any available alerts, including the information of the alert the title the message and the date of the alert. The user can click on actions if available to see the actions related to the alert. the user can then confirm the execution of the action.

V.8 Solution test: Attack simulation

In this section we will dive into simulating an ARP poisoning attack on our local network and test if the system will detect and alert the users.

The attacker will first scan the local network for devices connected using ettercap, or any arp-scan tool, we will use ettercap in this demonstration because it can simulate all the attack in one place.

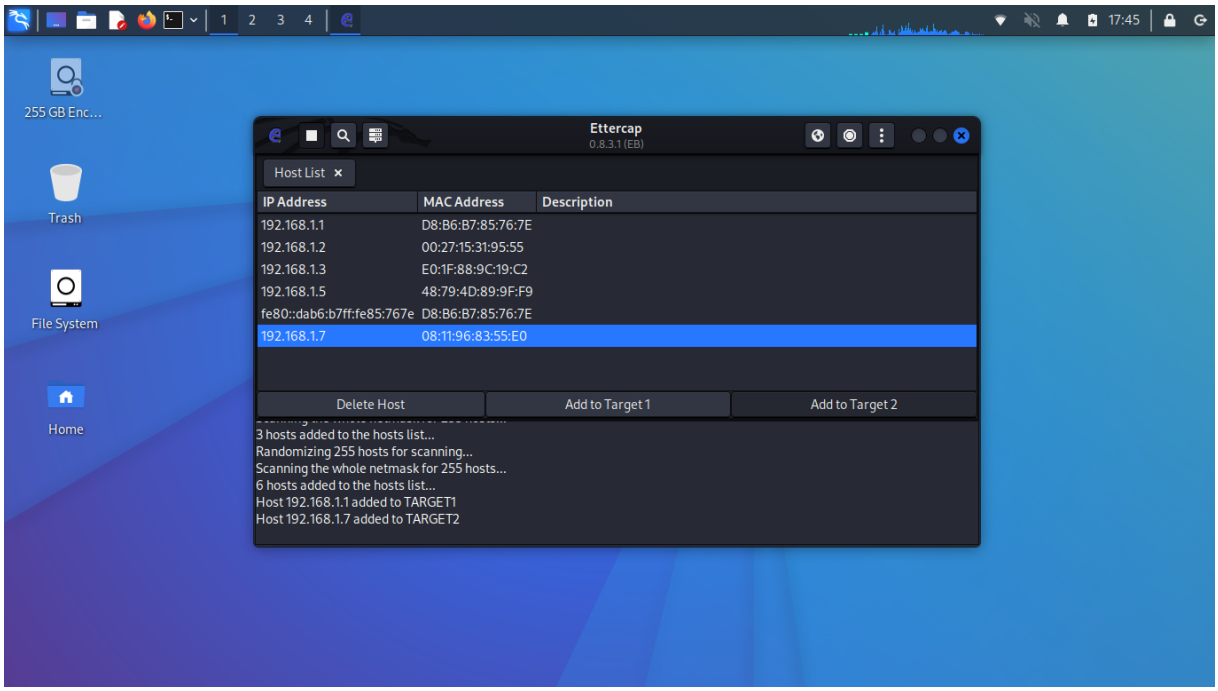


Figure 4.36: Ettercap scan for hosts

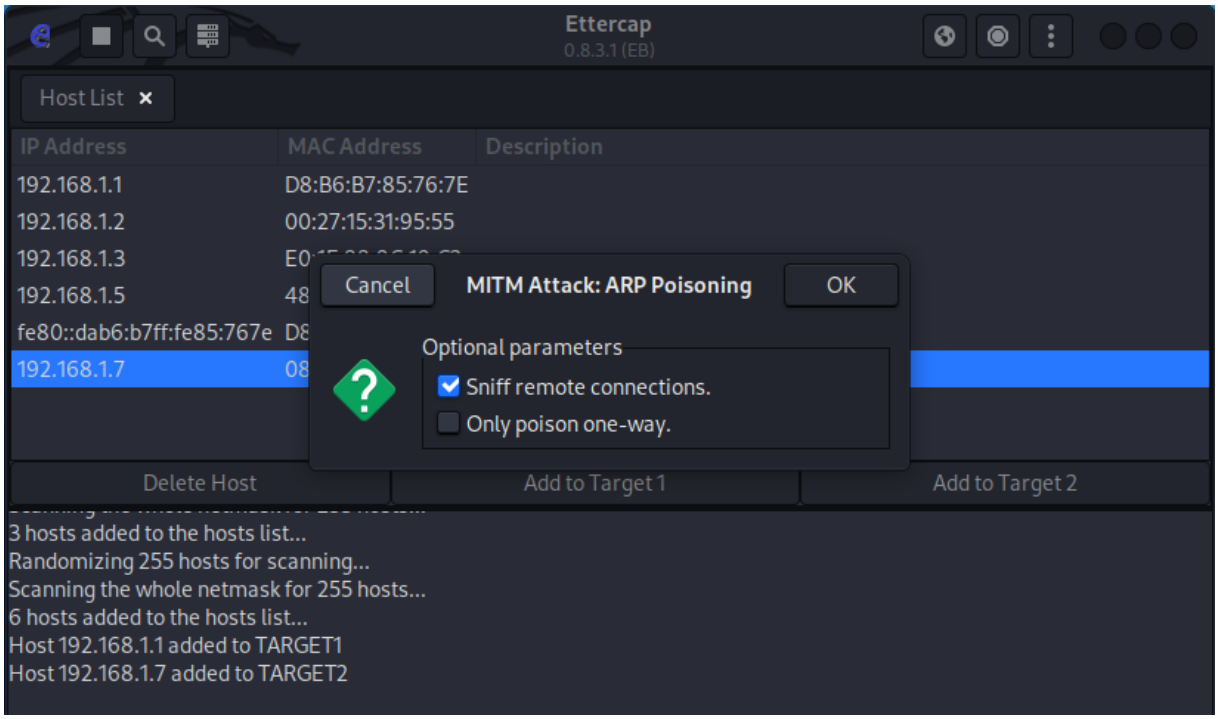
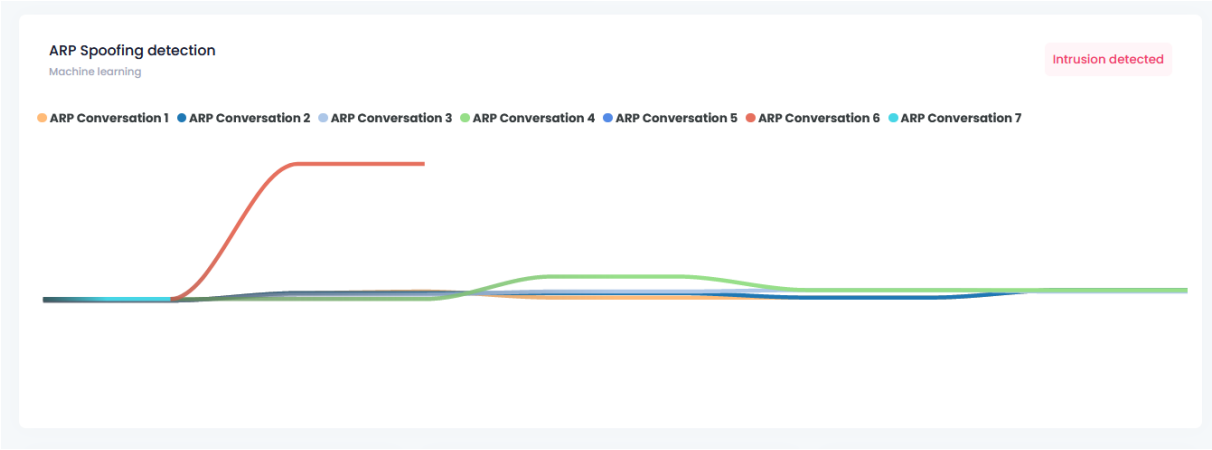


Figure 4.37: Ettercap start ARP poisoning

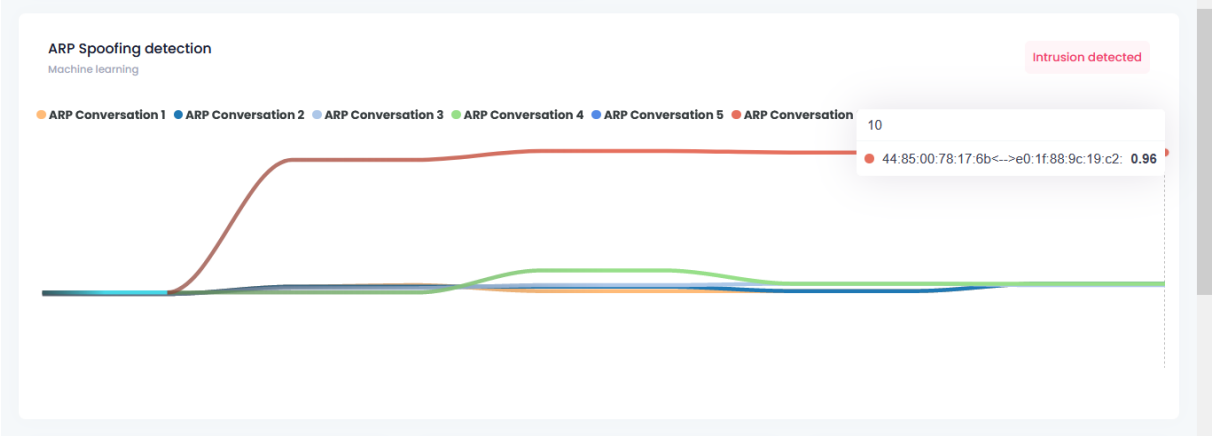
Those two simple steps had effectively enabled the attacker to trick the router and the victim into sending their data to the malicious host, attacker could listen to the victim traffic and steal sensitive information and potentially go beyond.

Our system actively listening to ARP traffic has already detected that the ARP conversation involving the address of the attacker is abnormal.

As you can see the multi line chart is indicating an anomaly and on the top right it indicates that an intrusion has been detected. the system had also created an alert and notified the users.



(a) Screen shot 1



(b) Screen shot 2

Figure 4.38: Stacked Vertical Graphs

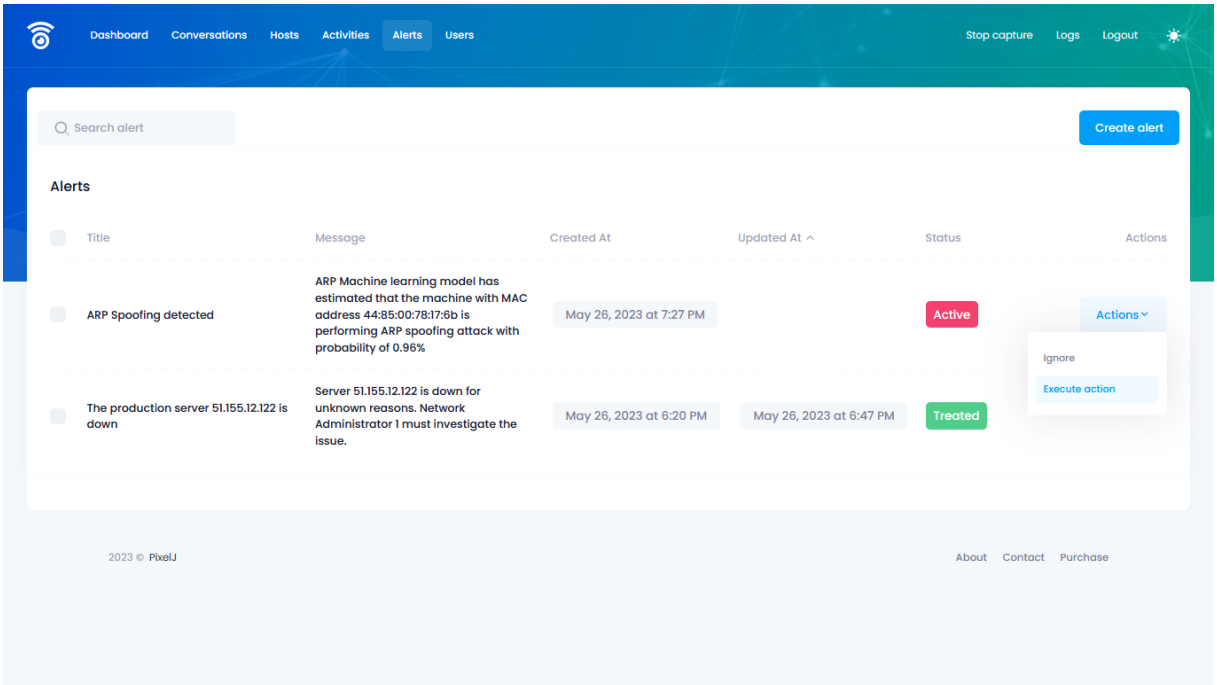


Figure 4.39: Alerts page

The user can either execute the action of de-authenticating the malicious mac address to mitigate the attack, from the web interface or from the mobile application.

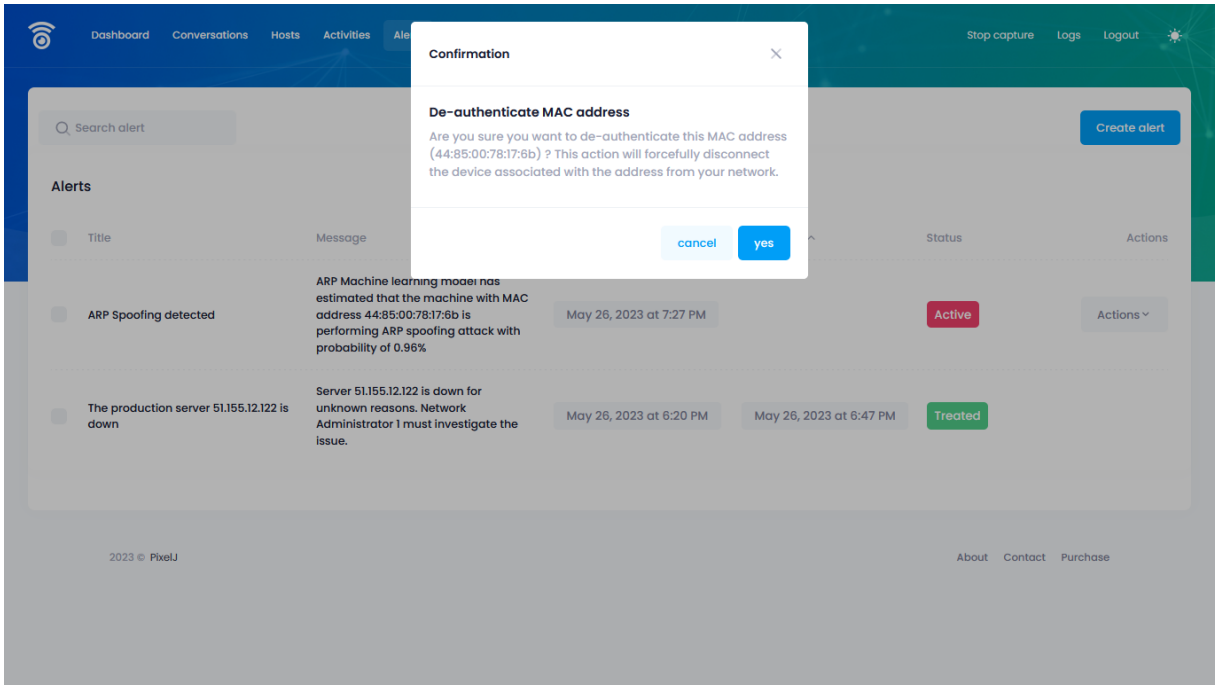
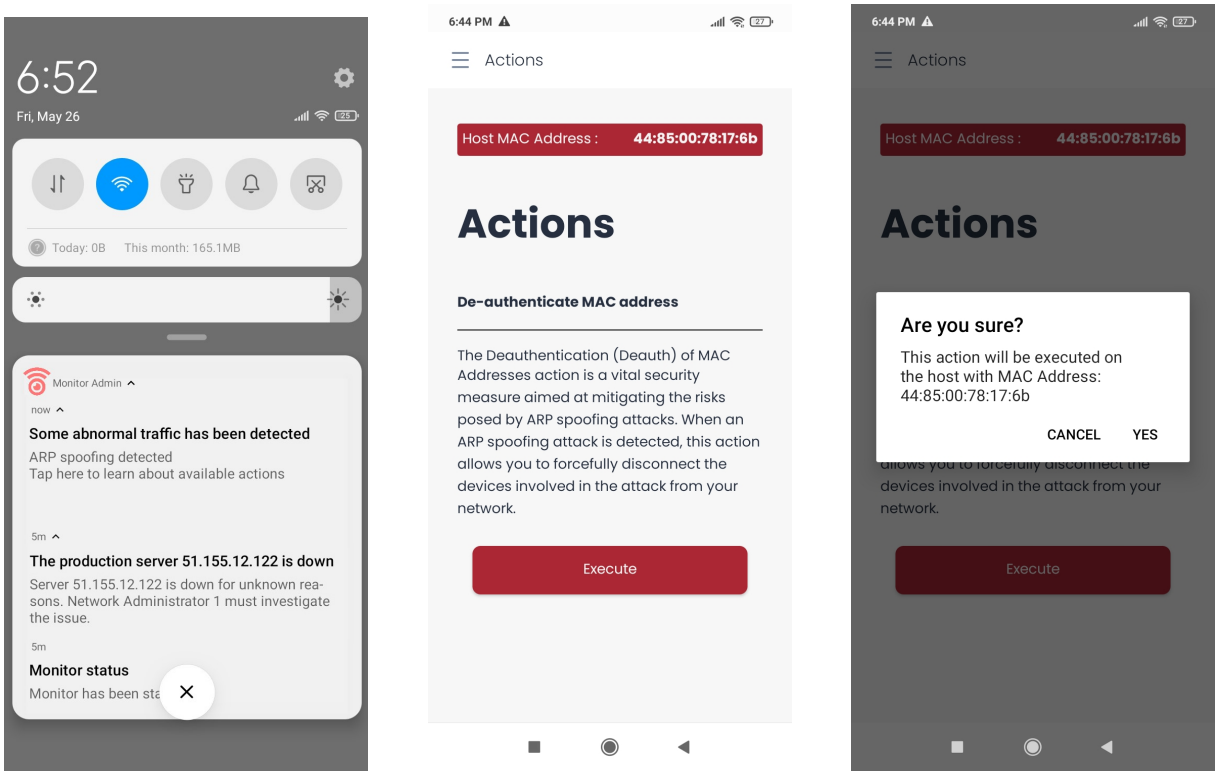


Figure 4.40: confirm action execution



(a) Screen Shot 1 (b) Screen Shot 2 (c) Screen Shot 3

Figure 4.41: Mobile App Home Screen (2)

Triggering the action will execute a script in the system server that continuously sends de-authentication frames with the mac address of the malicious host to the access point making it impossible for the attacker to reconnect again.

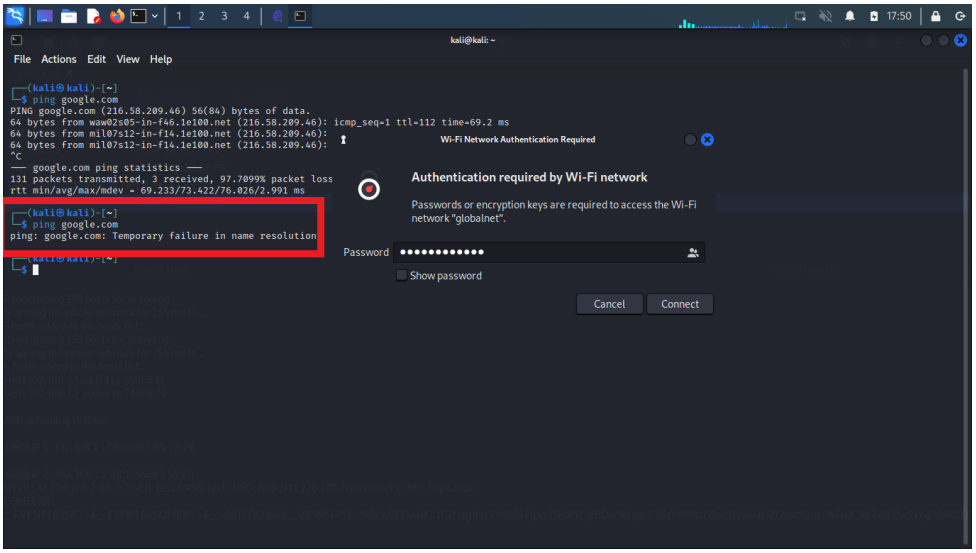


Figure 4.42: Action executed

VI Conclusion

In this chapter we walked through the different hardware and software environment used in order to implement our project, then we discussed the machine learning model selection through different model testing, including tensorflow and scikit-learn models, then we presented the principal user interfaces of the web and mobile applications. finally we conducted the solution test by simulating an ARP attack in our network environment and saw how our application reacts.

General conclusion

In conclusion, the objective of this project was mainly to enhance my knowledge in network security while applying what i studied in machine learning techniques. The project focused on the development of an Intrusion Detection System (IDS)/Intrusion Prevention System (IPS) using machine learning algorithms.

Throughout the project, various challenges were encountered, The initial task involved creating the core system, which involved capturing packets using tshark and extracting relevant information. Understanding and automating this process proved to be a significant learning experience. Additionally, implementing the machine learning model presented its own set of obstacles.

The comprehensive study and implementation of the system has significantly expanded my knowledge in network traffic analysis and network security, particularly in the area of the ARP protocol.

The developed system effectively detects and timely responds to one specific type of Man-in-the-Middle (MitM) attack. It meets the requirements set within the project timeline, serving as a solid starting point for a more comprehensive system. However, further work is required to expand the system's capabilities, such as studying and implementing detection and prevention mechanisms for additional attack types.

The project demonstrates great potential for future improvements and enhancements. The system's modular and extendable design allows for the addition of new attack detection and prevention flows. Further network monitoring charts and information can be incorporated, while the mobile application can be enhanced to offer more functionalities and actions for users.

Netography

[1] Address resolution protocol Wikipedia. https://en.wikipedia.org/wiki/Address_Resolution_Protocol, 01/02/2023.

[2] Tshark. https://www.wireshark.org/docs/man-pages/tshark.html, 04/03/2023.

[3] Docker Documentation. https://docs.docker.com/get-started/overview/, 11/04/2023.

[4] Symfony Documentation. https://symfony.com/doc/current/index.html, 15/03/2023.

[5] What is symfony. https://symfony.com/what-is-symfony, 15/03/2023.

[6] TensorFlow. https://www.tensorflow.org/learn, 19/04/2023.

[7] Scikit-learn. https://scikit-learn.org/, 23/04/2023.

[8] Wireshark. https://www.wireshark.org/about.html, 29/03/2023.