# A black and red text Description automatically generated

**PROJECT AND TEAM INFORMATION**

## Project Title

|  |
| --- |
| ***To Develop a real-time Intrusion Detection Firewall System*** |

## Student/Team Information

|  |  |
| --- | --- |
| Team Name: : Package Protectors  Team # T090 |  |
| **Team member 1 (Team Lead)** | Wason, Mansha – 23022973  [mansha2005wason@gmail.com](mailto:mansha2005wason@gmail.com)  Here's why Kit Harington is still rocking his John Snow look |
| ***Team member 2*** | Kala, Samriddhi – 2302223707 [kalasamriddhi@gmail.com](mailto:kalasamriddhi@gmail.com) |
| ***Team member 3*** | Tahzeeb, Humera – 230221338  [humeratahzeeb70@gmail.com](mailto:humeratahzeeb70@gmail.com) |
| ***Team member 4*** | Pant, Muskaan – 230122329 [muskaanpant246@gmail.com](mailto:muskaanpant246@gmail.com)  Here's why Kit Harington is still rocking his John Snow look |

**PROJECT PROGRESS DESCRIPTION (35 pts)**

## Project Abstract

|  |
| --- |
| In a world where digital threats are constantly evolving, protecting networks from cyber attacks has never been more important. Traditional firewalls, while useful, often fall short when dealing with sophisticated attacks like **DoS (Denial-of-Service)** or **port scanning**. They mostly rely on static rules and can’t adapt to new or unusual patterns, leaving systems vulnerable.  Our project aims to build a **real-time intrusion detection firewall** that not only filters network traffic but also intelligently detects suspicious activities as they happen. Imagine a system that keeps an eye on your network 24/7, spots abnormal behavior, and takes action to keep your data safe.  To achieve this, we’re using **Python** with powerful libraries like **Scapy** for capturing and analyzing network packets. We’ll combine **rule-based filtering** with **machine learning models** to detect unusual traffic patterns. This means the system will get better over time at spotting potential threats.  What sets our firewall apart is the **user-friendly web dashboard** built with **Flask**. It will show real-time logs, alert you instantly about potential intrusions, and allow you to update security rules easily. We’re also adding features like **Geo-IP blocking** to manage traffic from specific regions and automated alerts via email or SMS for critical events.  Whether it’s a small office or a home network, our system aims to be a practical, hands-on tool for everyday cybersecurity. It’s designed to be **flexible, scalable, and proactive**, helping users stay one step ahead of cyber threats. In the end, our project will deliver a smarter, more responsive firewall solution that keeps networks safe in a dynamic digital landscape. |

## Updated Project Approach and Architecture

|  |
| --- |
| Our goal is to build a **real-time intrusion detection firewall** that not only blocks suspicious network traffic but also learns to spot unusual patterns. To make this happen, we’re combining traditional filtering methods with smart, adaptive techniques. **How Our System Works:** The firewall has four main parts:   1. **Packet Sniffing and Filtering:**    * We use a Python library called **Scapy** to monitor incoming and outgoing network traffic.    * The system checks each packet against customizable rules (like blocking certain IPs or ports) stored in a simple database (**SQLite**). 2. **Intrusion Detection and Analysis:**    * The firewall doesn’t just follow fixed rules; it actively detects threats like **DoS attacks** or **port scanning** by analyzing traffic patterns.    * We’re adding **machine learning** to make the system smarter over time. Models like **Isolation Forest** and **One-Class SVM** will help spot new or unusual activities by learning what normal traffic looks like. 3. **Logging and Alerts:**    * The firewall logs all suspicious activities and unusual traffic using Python’s **logging module**.    * To keep users informed, it sends alerts via **email** (using SMTP) or **SMS** (using Twilio) whenever it detects something odd. 4. **User-Friendly Web Dashboard:**    * We’re building an interactive dashboard using **Flask**, with a clean and simple design using **Bootstrap**.    * The dashboard will display live traffic stats, alerts, and logs through **real-time charts** made with **Chart.js**.    * Users can easily add, update, or delete firewall rules without restarting the system.  **Keeping Everything Connected:** To make the dashboard responsive, we use **WebSocket** for real-time updates. Data like alerts and logs are shared in **JSON** format, which makes it easy to read and analyze.  By combining traditional filtering with smart detection, our firewall will offer a more **proactive and flexible** way to protect networks from cyber threats. |

## 

## Tasks Completed

|  |  |
| --- | --- |
| Task Completed | Team Member |
| Requirement analysis and research on existing firewall systems. Implemented rule-based detection for DoS attacks and port scanning. Designed the dashboard layout using Flask and Bootstrap | Mansha Wason |
| Configured real-time network traffic capturing and initial filtering. Assisted in setting up the logging and email alert system. | Humera Tahzeeb |
| Implemented core packet sniffing functionality using Scapy. Worked on rule-based detection for DoS attacks and port scanning. Integrated Chart.js for live traffic visualization on the dashboard. | Muskaan Pant |
| Requirement analysis and research on intrusion detection techniques. Developed logging mechanism and integrated the email alert system using SMTP. | Samriddhi Kala |

## Challenges/Roadblocks

|  |
| --- |
| As we develop our real-time intrusion detection firewall, we’ve faced several challenges that have pushed us to improve and adapt. One major issue is **processing network traffic quickly and efficiently**. Capturing and analyzing a high volume of packets in real time without slowing down the system has been difficult. To address this, we are optimizing our use of the Scapy library and exploring better ways to handle data to speed up processing.  Another challenge is **accurately detecting attacks like DoS and port scanning** while avoiding false alarms. It’s tricky to find the right balance between catching real threats and not flagging normal traffic as malicious. To improve accuracy, we’re refining our detection algorithms and adding machine learning models that learn from traffic patterns over time.  Building a **responsive, real-time web dashboard** has also been challenging. Displaying live logs and alerts without delays required experimenting with communication methods like WebSockets and AJAX. We plan to continue optimizing the dashboard to make it fast and user-friendly.  Lastly, coordinating work among team members while working remotely sometimes slowed communication and debugging. To overcome this, we hold regular online meetings, use shared code repositories, and maintain clear documentation.  Although these challenges have been tough, they’ve helped us grow and improve our project. With continued teamwork and problem-solving, we’re confident we’ll overcome these hurdles and create a strong, reliable firewall system. |

## Future Scope

Our real-time intrusion detection firewall system provides strong foundational security by filtering packets at the network layer using raw socket programming. It inspects each packet’s source and destination IP addresses, ports, and protocol type, and enforces a strict rule-based access control mechanism, where only explicitly allowed traffic is permitted—everything else is denied by default. Multi-threaded monitoring across all active interfaces ensures no packet goes unfiltered, and detailed logging is maintained for every decision, enabling traceability and intrusion analysis. Additionally, IP forwarding is disabled at the OS level to ensure strict route enforcement through firewall logic. For future enhancements, the system can be extended to include stateful inspection, machine learning for anomaly detection, a GUI-based rule editor, and integration with threat intelligence feeds. Adding features like IPv6 support, real-time alert notifications, and deep packet inspection (DPI) would further strengthen the firewall, making it more adaptive and suitable for modern, high-performance and cloud-based networks.

## 

## Project Outcome/Deliverables

|  |
| --- |
| The main result of our project is a working real-time intrusion detection firewall system built using Python. This system continuously watches network traffic to spot and block harmful activities like DoS attacks, port scanning, and other unusual behaviors, helping to keep networks safe and secure.  Our key deliverables include:   * A core firewall engine that captures and filters network packets based on customizable rules like IP addresses, ports, and protocols. * An intrusion detection module that uses both rule-based methods and machine learning to catch both known and new types of cyber threats as they happen. * A logging and alert system that records suspicious activities and sends instant notifications via email or SMS to keep administrators informed. * An easy-to-use web dashboard built with Flask, where users can monitor traffic live, receive alerts, and manage firewall rules dynamically. * A Geo-IP blocking feature to filter traffic based on geographic locations for added security. * Complete documentation that guides users through installation, usage, and potential future upgrades.   Overall, this project delivers a powerful, flexible, and user-friendly security tool that small businesses and individuals can use to better protect their networks from ever-changing cyber threats. |

# Progress Overview

|  |
| --- |
| So far, we’ve made solid progress on the project. The core firewall engine and intrusion detection modules are fully developed and tested, which is a big win for us. We’ve also successfully implemented the real-time alert system and basic web dashboard for monitoring traffic and managing rules.  Some parts, like the advanced anomaly detection with machine learning and certain UI enhancements on the dashboard, are still a work in progress. These features are slightly behind schedule but we’re actively working on them and confident they’ll be ready soon.  On the bright side, tasks like packet sniffing, filtering mechanisms, and alert notifications are ahead of schedule, giving us some extra buffer time. This progress allows us to focus more on refining the system’s usability and accuracy in the coming weeks.  Overall, the project is on track to meet its goals, with most critical components completed and remaining features being developed as planned. We’re committed to maintaining this momentum and addressing any challenges promptly to ensure a strong final delivery. |

# Codebase Information

|  |
| --- |
| Our entire project code is securely stored in a GitHub repository to keep everything organized and accessible for the team. The main development work is happening on the main branch, where we regularly merge updates after thorough testing.  We’ve made several important commits that mark key milestones, such as completing the packet sniffing module, integrating the intrusion detection logic, and setting up the web dashboard. Each commit includes clear messages to explain what changes were made, which helps us track progress easily and roll back if needed.  This structured approach ensures smooth collaboration among team members and makes it easy to review and improve the code as we move forward. |

## Testing and Validation Status

|  |  |  |
| --- | --- | --- |
| Test Type | Status(Pass/Fail) | Notes |
| Unit Testing | Pass | All individual modules, including packet filtering and detection, passed successfully. |
| Integration Testing | Pass | Modules work well together, with smooth data flow between components. |
| Performance Testing | Pass | The system handles expected network traffic efficiently without lag |
| Anomaly Detection Testing | In progress | Basic anomaly detection works; machine learning enhancements still being tested. |
| Alert Notification Testing | Pass | Email and SMS alerts are reliably sent upon detecting threats. |
| User Interface Testing | In progress | Dashboard functions well; some UI improvements and bug fixes are ongoing. |

# Deliverables Progress

|  |
| --- |
|  **Firewall Core Engine:** Completed. The core system for capturing and filtering network traffic is fully developed and tested.   **Intrusion Detection Module:** In progress. The basic rule-based detection is working well, and machine learning integration is underway to improve threat detection accuracy.   **Logging and Alert System:** Completed. Suspicious activities are logged, and real-time email/SMS alerts are successfully sent to administrators.   **Flask-Based Web Dashboard:** In progress. The dashboard is functional with live traffic monitoring and alert displays. Some features like dynamic rule management are still being refined.   **Geo-IP Blocking Feature:** Pending. We are planning to implement geographic traffic filtering soon using GeoIP libraries.   **Project Documentation:** In progress. Most of the technical and user guides have been drafted and are being finalized. |