Intrusion Detection System

A thesis submitted in partial fulfillment of the requirement for the award of the degree
Of

Bachelor of Computer Application
in

Faculty of Engineering and Technology

Submitted by

Chinmoy Das
Roll no: ADTU/2022-25/BCASP/014
Abhishek Dutta
Roll no: ADTU/2022-25/BCASP/040

Under the guidance of

Dr. Mala Dutta
Associate Professor
Faculty of Computer Technology

Assam down town University
Guwahati-26, Assam
Session: January-June, 2025

CONTENTS

- *Certificate of Approval* 3
- *Certificate from Guide* 4
- *Certificate from External Examiner* 5
- *Declaration* 6
- *Acknowledgement* 7
- *Abstract* 8, 9
- *List of Figures* 10

1. **Introduction**

1.1. Overview of the Project 10
1.2. Motivation 11
1.3. Scope and Objective 11
1.4. Existing System 11, 12
1.5. Problem Definition 12
1.6. Proposed System 12, 13

2. **Project Analysis**

2.1. Project Requirement Analysis 13
2.2. Gantt Chart 14,15
2.3. Advantage & Disadvantage 14, 15, 16
2.4. Project Life Cycle 16, 17
2.5. Project Feasibility 17, 18

3. **Project Design**

3.1. System Architecture 19
3.2. Data Flow Diagram(DFD) 20, 21
3.3. Use Case Diagram 21, 22
3.4. Sequence Diagram 22
3.5 Interface/ScreenShots 23, 24

4. **Project Implementation**

4.1. Description of the Software Used 25, 26

5. **Testing**

5.1. Types of Testing 26
5.2. Test Cases 27-28

6. **Achievements** 28

7. **Conclusion and Future Scope**

7.1. Conclusion 29
7.2. References 29

Faculty of Computer Technology
Assam down town University
Gandhi Nagar, Panikhaiti, Guwahati- 781026, Assam

CERTIFICATE OF APPROVAL

This is to certify that the project report entitled ***"Intrusion Detection System"*** submitted by Chinmoy Das bearing Roll No. ADTU/2022-25/BCASP/014 and Abhishek Dutta bearing Roll No. ADTU/2022-25/BCASP/040, are hereby accorded our approval as a study carried out and presented in a manner required for acceptance in partial fulfilment for the award of the degree of ***Bachelor of Computer Application*** under Assam Down Town University. This approval does not necessarily endorse or accept every statement made opinion expressed or conclusion drawn as recorded in the report. It only signifies the acceptance of the project report for a purpose which is submitted.

Date: 23/05/25

Place: Guwahati

Faculty of Computer Technology
Assam Down Town University
Gandhi Nagar, Panikhaiti, Guwahati- 781026, Assam

CERTIFICATE FROM GUIDE

This is to certify that the project report entitled ***"Intrusion Detection System"*** submitted by **Chinmoy Das** bearing Roll No. ADTU/2022-25/BCASP/014 and **Abhishek Dutta** bearing Roll No. ADTU/2022-25/BCASP/040 towards the partial fulfilment of the requirements for the award of the degree of ***Bachelor of Computer Application*** under Assam down town University is a bonafide research work carried out by them under my supervision and guidance. This work has not been submitted previously for any other degree of this or any other University.

I recommend that the thesis may be placed before the examiners for consideration of award of the degree of this University.

Date:

Place: Guwahati

Dr. Mala Dutta

Associate Professor, Faculty of Computer Technology

Assam down town University

Faculty of Computer Technology
Assam down town University
Gandhi Nagar, Panikhaiti, Guwahati- 781026, Assam

CERTIFICATE FROM EXTERNAL EXAMINER

This is to certify that the project report entitled ***"Intrusion Detection System"*** submitted by Chinmoy Das bearing Roll No. ADTU/2022-25/BCASP/014 and Abhishek Dutta bearing Roll No. ADTU/2022-25/BCASP/040 towards the partial fulfilment of the requirements for the award of the degree of ***Bachelor of Computer Application*** under Assam down town University is a bonafide research work carried out by him under the supervision and guidance of ***Dr. Mala Dutta***, Associate Dean, Department of Computer Technology, Assam down town University, Guwahati has been examined by me and found to be satisfactory.

I recommend the thesis for consideration for the award of the degree of ***Bachelor of Computer Application*** under Assam down town University.

Date:

Place: (Signature of External Examiner)

DECLARATION

We, Chinmoy Das bearing Roll No. ADTU/2022-25/BCASP/014 and Abhishek Dutta bearing Roll No. ADTU/2022-25/BCASP/040hereby declare that the thesis entitled ***"INTRUSION DETECTION SYSTEM"*** is an original work carried out in the Department of Computer Technology, Assam down town University, Guwahati with exception of guidance and suggestions received from my supervisor, ***Dr. Mala Dutta***, Associate Professor, Department of Computer Technology, Assam down town University, Guwahati. The data and the findings discussed in the thesis are the outcome of our research work. This thesis is being submitted to Assam down town University for the degree of ***Bachelor of Computer Application".***

ACKNOWLEDGMENT

We would like to extend our heartfelt appreciation to everyone who contributed to the successful completion of this project. Our sincere thanks go to our project team members for their dedication and collaboration throughout the project. Each member played a significant role in shaping the outcome.

Special thanks to our supervisor, *Dr. Mala Dutta*, for her consistent guidance and valuable feedback, which greatly enriched the quality of our work.

We would also like to express our gratitude to the respected teaching faculty members of our department for their support, insights, and encouragement during the course of this project. Their contributions, both direct and indirect, were instrumental in helping us navigate through the challenges and achieve our objectives.

ABSTRACT

In today's highly interconnected digital world, safeguarding information systems against unauthorized access, data breaches, and evolving cyber threats is of paramount importance. Traditional defense mechanisms such as firewalls and antivirus software, though essential, often fall short in detecting sophisticated or novel intrusion attempts. As networks continue to expand in complexity and scale, the need for intelligent, real-time monitoring solutions becomes increasingly vital.

This project focuses on the development of a modular and extensible **Intrusion Detection System (IDS)** aimed at monitoring network traffic, identifying malicious activities, and generating timely alerts to mitigate potential threats. The IDS is designed to detect a wide range of intrusions including port scanning, denial of service (DoS) attacks, malware communication, and suspicious behaviors by analyzing both system logs and real-time packet captures.

To enhance detection precision, the proposed system integrates signature-based techniques—using predefined attack patterns—and anomaly-based approaches that recognize deviations from normal network behavior. To reduce false positives and improve adaptability, machine learning models may be introduced to perform pattern recognition and behavioral profiling, particularly for identifying zero-day attacks or advanced persistent threats (APTs).

The system is developed using Snort, a widely recognized open-source IDS, with a user-friendly layer built in Python to simplify configuration and rule management, making it accessible even to users with limited IDS experience. Critical components such as libdaq are used for packet I/O, libdnet for low-level networking utilities, and hwloc for CPU affinity optimization. Additionally, dependencies like OpenSSL, LuaJIT, PCRE2, and flex further extend the system's performance and capability.

Real-time traffic is captured and analyzed using pcap, while the use of zlib supports compression handling. The architecture supports both standalone and distributed deployments, ensuring scalability and flexibility across diverse environments. Whether

deployed in a small network or integrated into an enterprise setup, the system aims to provide reliable protection with low false-positive rates, high throughput, and ease of maintenance.

Overall, this project delivers a lightweight yet powerful IDS solution that bridges the gap between robust network protection and practical usability. By enhancing the core Snort engine with a Python interface and machine learning capabilities, the system aims to offer both technical effectiveness and operational simplicity, aligning with the evolving demands of modern cybersecurity.

List of Figures

Sl no.	Name of the figure/chart	Page No.
1	Gantt Chart	15
2	System Architecture	20
3	Context diagram	21
4	Sequence diagram	23

1. INTRODUCTION

1.1 Overview of the project

This project aims to develop an **Intrusion Detection System (IDS)** that monitors network or system activity to detect and alert against suspicious or unauthorized behavior. As cyber threats continue to grow, an IDS adds an important layer of security by identifying potential attacks in real time.

This project involves the development of an Intrusion Detection System (IDS) that uses Nmap to scan and monitor network activity for known vulnerabilities and threats. By relying on signature-based detection, the system can identify suspicious patterns and behaviors that match a predefined database of attack signatures. The focus is on detecting known intrusions efficiently and alerting users to take necessary action. It is designed for small to medium-scale networks where basic security monitoring is required.

1.2 Motivation

With the rise in cyber threats targeting network vulnerabilities, there is a strong need for simple and effective tools that help monitor and protect systems. Many organizations, especially smaller ones, lack advanced security infrastructure. This project is motivated by the need to provide a basic yet reliable Intrusion Detection System that uses Nmap to identify known threats by scanning for open ports, services, and known vulnerabilities. By focusing on signature-based detection, the system offers a practical solution for early threat detection and strengthens basic network security.

1.3 Scope and Objective

This project focuses on developing a lightweight and effective Intrusion Detection System (IDS) capable of monitoring network or system activity to detect potential security breaches.

The main objective of this project is to develop an Intrusion Detection System (IDS) that can accurately detect and alert on potential security threats in real time. It uses Nmap to scan network activity and detect known types of attacks.It focuses on signature-based detection, making it effective for recognizing predefined threats and alerting administrators for timely response.

1.4 Existing system

Several intrusion detection systems (IDS) are currently available, with Snort, Suricata, and OSSEC being among the most widely used. These systems provide robust features such as real-time traffic analysis, signature and anomaly-based threat detection, and integration with external threat intelligence feeds. Snort, in particular, is a highly reliable open-source IDS known for its flexible rule-based detection engine and wide community support.

However, while Snort offers powerful detection capabilities, its default configuration can be complex for users with limited experience in network security. It typically requires manual rule writing, command-line configuration, and integration with other tools to achieve full functionality. Moreover, Snort in its default form may not be tailored for specific use cases like focused detection of vulnerabilities scanned by tools such as Nmap.

Therefore, this project builds on Snort but customizes and optimizes it for more targeted detection of network scans and known vulnerabilities, offering a lightweight and user-focused IDS environment. By leveraging Snort's engine while simplifying its deployment and tuning it for specific scanning behavior, the project addresses limitations of general-purpose configurations and makes IDS more accessible for smaller-scale or educational use cases.

1.5 Problem Definition

Modern Intrusion Detection Systems (IDS) are often powerful but come with steep learning curves. Most existing tools like Snort or Suricata require prior knowledge of system configurations, rule syntax, and networking fundamentals, which makes them difficult to use for beginners or small teams without deep cybersecurity expertise.

The core problem lies in the lack of a **user-friendly, accessible IDS solution** that still offers real-time detection and strong capabilities. This project aims to solve that by enhancing Snort with a **Python-based interface**, drastically simplifying its usability. Users only need basic Python knowledge to run and operate the system — removing technical barriers and making practical intrusion detection achievable even for entry-level users or educational use cases.

1.6 Proposed System

The proposed Intrusion Detection System (IDS) integrates the powerful detection capabilities of **Snort** with an intuitive, Python-based graphical interface to improve both usability and efficiency. While Snort performs deep packet inspection and rule-based detection of threats, the Python layer simplifies system interaction by automating configuration, monitoring, and alert visualization.

This combination makes the IDS more accessible for beginners and non-expert users who may find traditional command-line IDS tools complex. The system continuously analyzes network traffic in real-time, effectively identifying known vulnerabilities and common scanning behaviors—such as those generated by tools like **Nmap**. It also minimizes false positives and ensures stable performance across networks of varying size, making it suitable for both educational and operational environments.

2. PROJECT ANALYSIS

2.1 Project Requirement Analysis

The Intrusion Detection System (IDS) is designed to enhance overall network security by fulfilling both functional and non-functional requirements. On the functional side, the system must be capable of continuously monitoring network traffic in real time, identifying potential threats through both signature-based detection (which matches known attack patterns) and anomaly-based detection (which flags unusual behaviours). It must also generate immediate alerts upon detection, log all relevant incident data for auditing or forensic analysis, and support flexible rule configuration to adapt to different network environments and threat models.

In terms of non-functional requirements, the IDS must maintain high performance and low latency to ensure it does not disrupt network operations, even under heavy traffic. Scalability is key—it should operate efficiently in both small-scale and enterprise-level networks. The system must also exhibit reliability, remaining active and accurate over long periods with minimal downtime. Usability is another critical aspect; administrators should be able to manage and configure the system without extensive training. Finally, the IDS must be secure against tampering, and its maintenance—including updates to detection signatures and anomaly profiles—should be straightforward to ensure continued effectiveness against evolving cyber threats.

2.2 Gantt Chart

	March	April	May
Information Gathering	■		
Analysis		■	
Design		■	
Coding		■	■
Testing			■
Analysis			■

2.3 Advantage and Disadvantage

2.31 Advantages

1. **Real-Time Threat Detection**
   The IDS provides continuous surveillance of network traffic, enabling immediate identification and response to suspicious or malicious activities. This reduces the window of vulnerability.
2. **Automated Threat Identification**
   Through a combination of signature-based and anomaly-based detection techniques,

the system is capable of recognizing known malware patterns as well as unusual behaviors that may indicate zero-day attacks or insider threats.

3. **Enhanced Security Oversight**
   The system generates detailed logs and alerts, providing actionable insights to system administrators. This enhances the organization's ability to audit, track, and mitigate security events effectively.
4. **Scalability**
   Designed to adapt to varying network sizes, the IDS can scale horizontally to support increased data throughput and traffic volumes, making it suitable for enterprise-grade as well as small-scale deployments.
5. **Customizability and Flexibility**
   Detection rules, thresholds, and behaviors can be custom-configured by administrators, allowing the IDS to align with specific security policies and network architectures.

2.32 Disadvantages

1. **False Positives and Alert Fatigue**
   Anomaly-based detection may flag legitimate traffic as malicious, leading to false positives. This can overwhelm administrators and dilute focus from actual threats.
2. **High Resource Consumption**
   Real-time traffic inspection and deep packet analysis demand significant CPU and memory usage, which could degrade performance on lower-end systems if not properly optimized.
3. **Complex Setup and Configuration**
   Effective deployment requires technical expertise in networking, threat modeling, and system integration. Misconfiguration could either reduce detection efficacy or increase false alerts.
4. **Ongoing Maintenance and Updates**
   To remain effective, the IDS must be routinely updated with the latest threat

signatures and tuned for evolving attack vectors. This introduces long-term maintenance overhead.

2.4 Project Lifecycle

The development of the Intrusion Detection System (IDS) adhered to a structured project lifecycle to ensure a smooth transition from concept to deployment and maintenance. The key phases of this lifecycle are outlined below:

i) Initiation:

The project began with identifying the need for a robust IDS solution to enhance network security. This phase involved setting clear objectives, analyzing the existing security landscape, and gathering initial requirements from stakeholders.

ii) Planning:

In this phase, the project scope was defined in detail. A comprehensive plan was formulated covering task allocation, resource requirements, timelines, and risk management strategies. The system architecture and technology stack were also designed.

iii) Development:

Core functionalities were implemented, including network traffic monitoring, packet analysis, rule-based detection using Snort, and Python-based automation for log handling and alerts. Custom rules were also developed to detect specific intrusion patterns.

iv) Testing & Quality Assurance:

Extensive testing was conducted to ensure the system's stability, detection accuracy, performance, and resilience against known threats. Test cases simulated various attack vectors, including scans with tools like Nmap, to validate real-time detection.

v) Deployment:

The IDS was deployed within the target network environment. Network interfaces were configured appropriately, and integration with existing security systems was ensured. The user interface was finalized for streamlined control (e.g., Start/Stop functions).

vi) Monitoring & Maintenance:

Post-deployment, the system was continuously monitored for performance and reliability. Updates to threat signatures, rule refinement, and routine maintenance tasks were carried out to keep the system effective against emerging threats.

vii) Evaluation & Optimization:

The IDS was periodically evaluated based on feedback and real-world performance. Analytical reviews led to optimizations in detection logic, user interaction, and system efficiency to reduce false positives and improve accuracy.

viii) Closure:

The final phase involved wrapping up project documentation, compiling performance reports, and transitioning the IDS to operational use. Knowledge transfer and long-term support plans were also outlined.

2.5 Project feasibility

The feasibility analysis of the Intrusion Detection System (IDS) project assesses its practicality and potential for successful implementation by examining several critical dimensions:

I. Technical Feasibility:

This dimension evaluates the capability to develop and deploy the IDS using the selected technologies, including machine learning algorithms, real-time monitoring tools, and network analysis frameworks. It further considers the system's compatibility with the existing IT infrastructure and its scalability to accommodate future network expansion.

II. Market Feasibility:

Market feasibility assesses the demand for the IDS within the targeted user base, primarily enterprises and organizations seeking enhanced network security solutions. It includes an analysis of the competitive landscape and identifies possible challenges related to user adoption and market penetration.

III. Financial Feasibility:

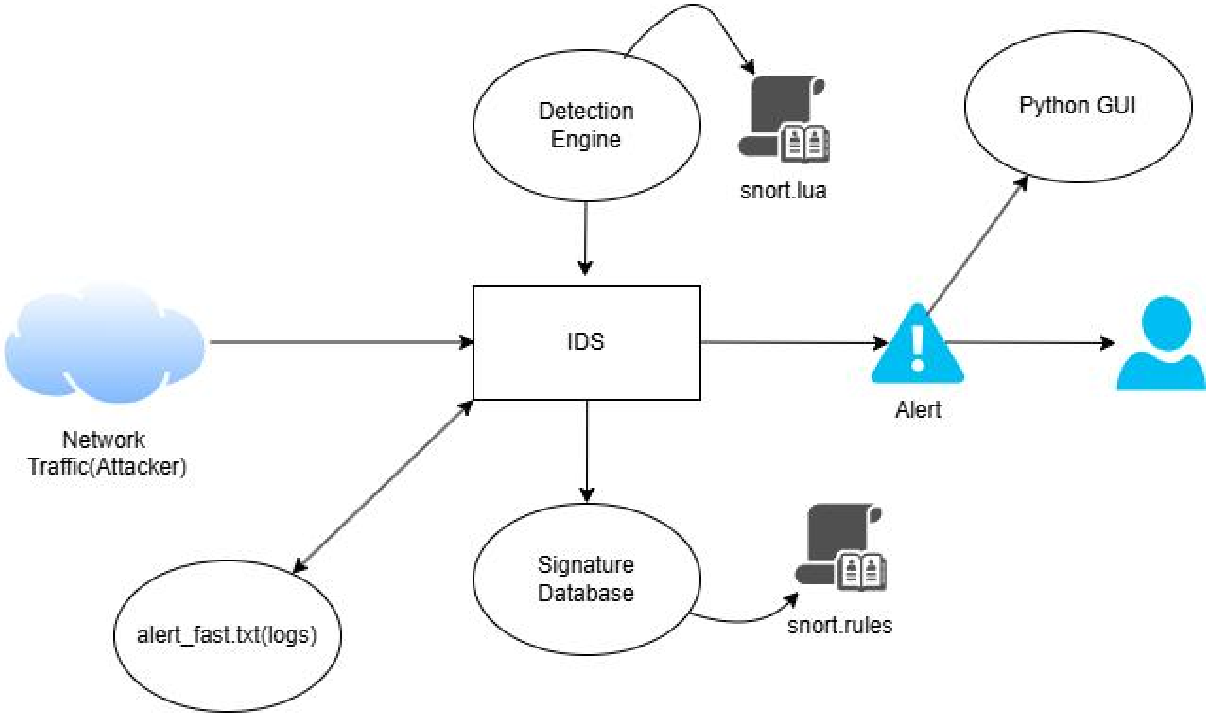
This aspect involves estimating the total costs related to system development, implementation, ongoing maintenance, and updates. These costs are compared against the expected financial benefits, such as reduction in cybersecurity risks and potential savings from averting data breaches, to determine the economic viability of the project.

IV. Operational Feasibility:

Operational feasibility examines the practicality of integrating the IDS into the current network environment without interrupting ongoing business processes. It also evaluates the resource requirements for deployment, monitoring, and maintenance, including necessary personnel and training.

3. PROJECT DESIGN

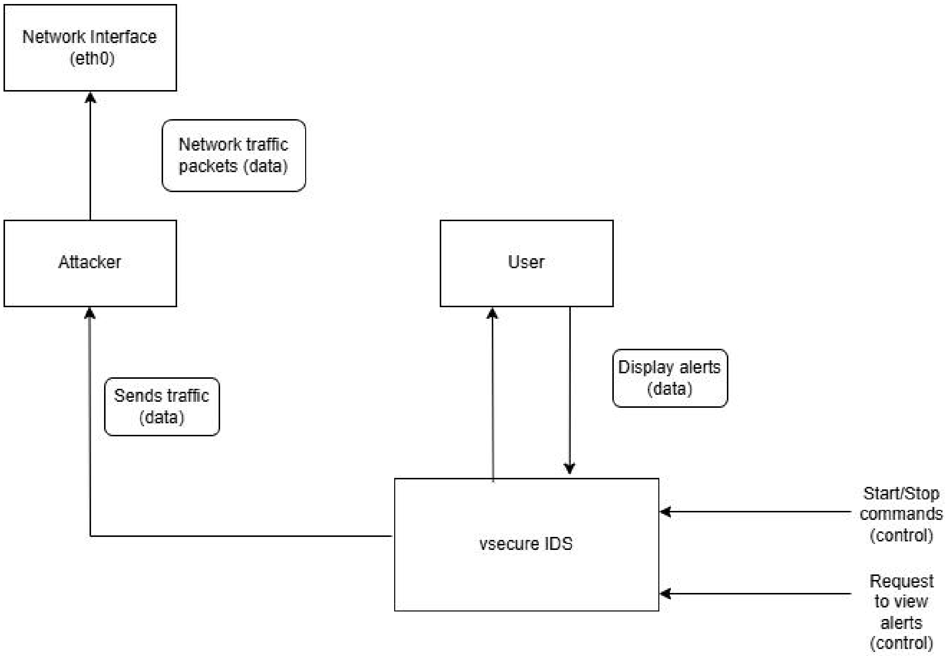
3.1 System Architecture



The proposed IDS architecture combines Snort’s powerful detection capabilities with a user-friendly Python GUI to enable real-time monitoring of network traffic. Incoming traffic is analyzed by the Detection Engine using Snort’s configuration (snort.lua) and rule files (snort.rules) to identify known threats. Alerts are generated for suspicious activity and presented to the user through the Python GUI, while logs are saved in alert_fast.txt for further analysis. This setup reduces the need for advanced technical knowledge, making it lightweight, efficient, and accessible for students or small-scale deployments.

3.2 Data Flow Diagram

3.2.1 Context Diagram



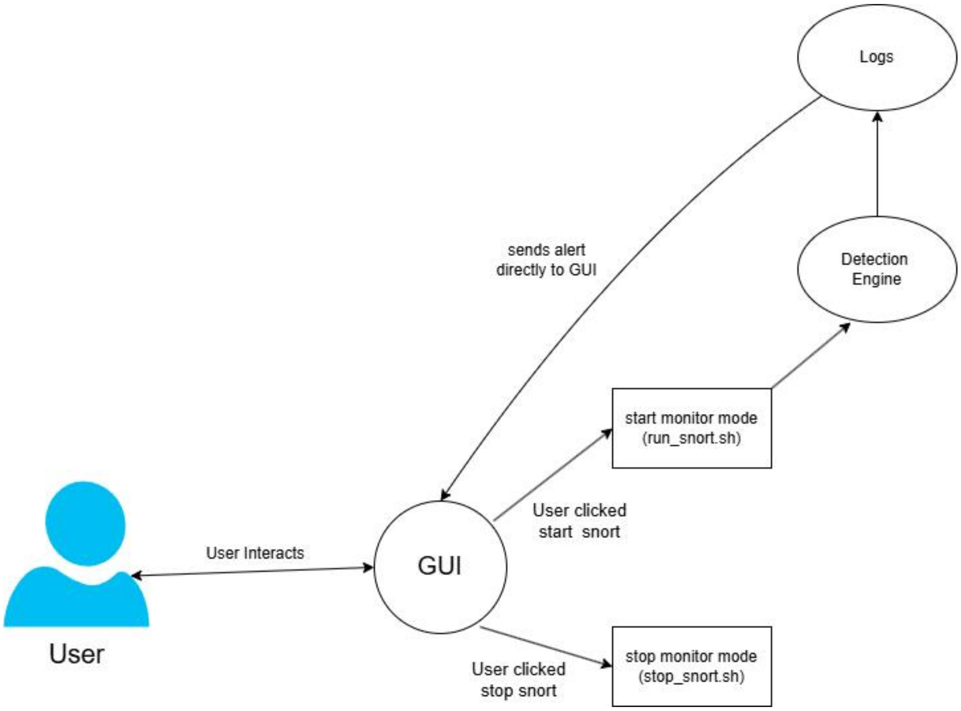
Network Interface (eth0)
Network traffic packets (data)
Attacker
User
Display alerts (data)
Sends traffic (data)
vsecure IDS
Start/Stop commands (control)
Request to view alerts (control)

The context flow diagram you described outlines the operational flow of the vSecure IDS system within a network environment, highlighting its interactions with key entities. Here’s a concise explanation to expand on the diagram while keeping it brief:

The vSecure IDS operates by capturing network traffic through a designated interface, such as eth0, where an attacker may send malicious packets (e.g., port scans or exploit attempts). The IDS, powered by Snort, actively monitors this incoming traffic in real time, analyzing it against its rule-based detection engine (using snort.rules and snort.lua). When suspicious or malicious activity is detected, the system generates alerts, which are logged to alert_fast.txt and simultaneously displayed on the Python-based Tkinter GUI for immediate user visibility.

Users interact with the IDS through the GUI, issuing commands to start or stop traffic monitoring and requesting to view alert logs. This bidirectional interaction ensures that users, even with minimal technical expertise, can control the system and respond to threats efficiently. The streamlined flow—attacker to network interface, IDS processing, alert generation, and user interaction—creates a cohesive and responsive intrusion detection process.

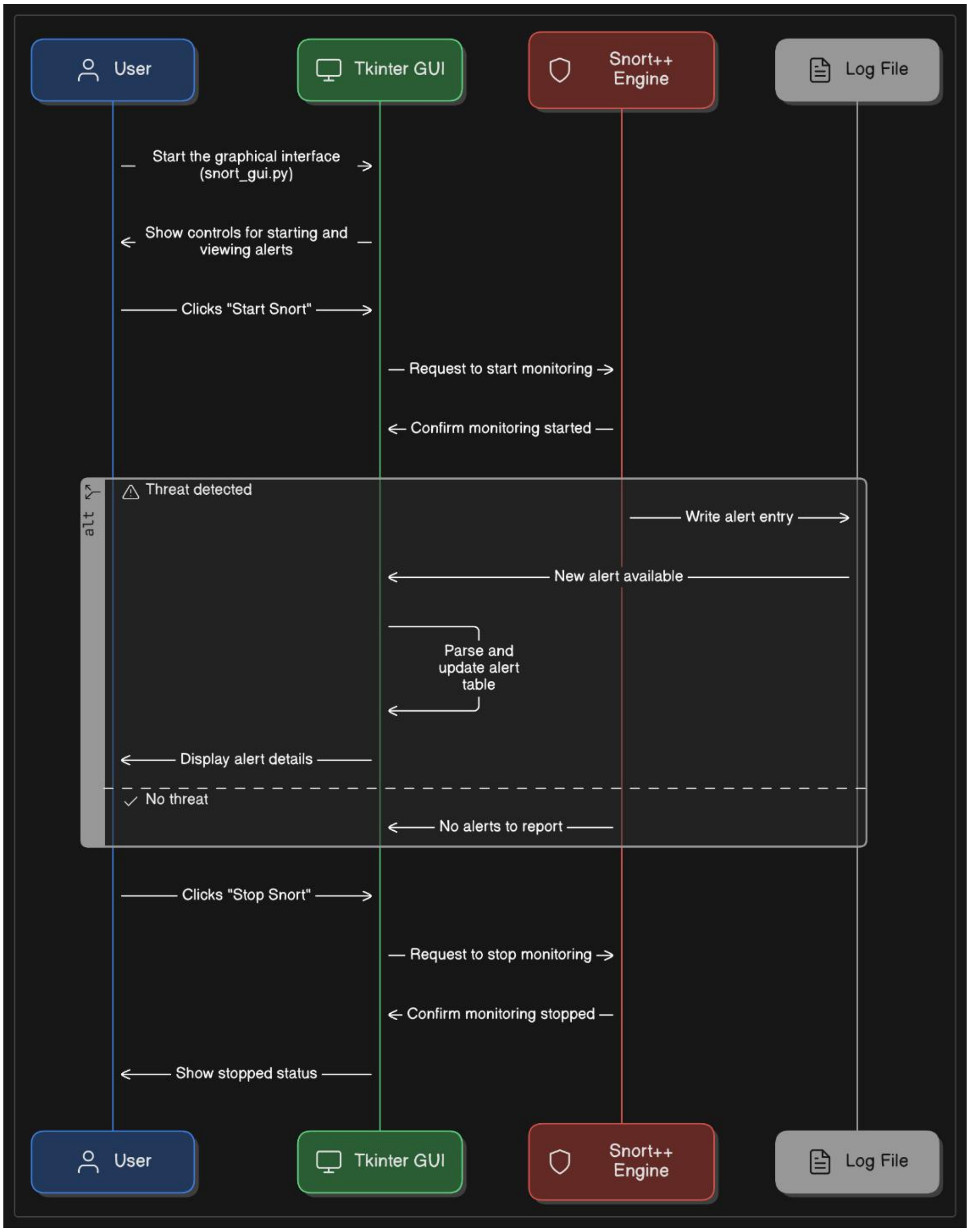
3.3 Use case Diagram



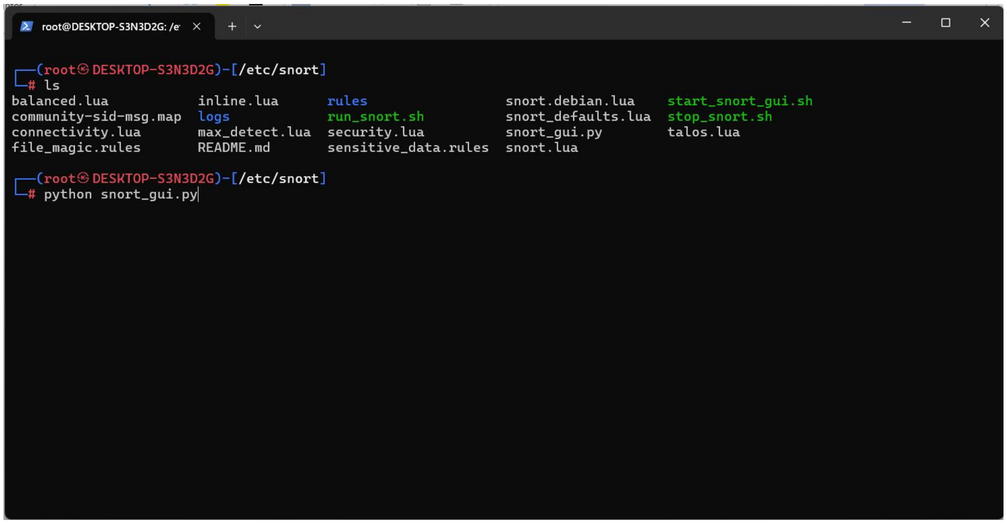
The use case diagram you provided illustrates the interactions between the user, the GUI, and the underlying components of the vSecure IDS system, focusing on how the user engages with the system to monitor and manage network traffic. Here's a concise explanation of the diagram, expanding on its elements while keeping it brief, similar to the style of the context flow diagram explanation:

In the vSecure IDS use case diagram, the "User" interacts with the system through the "GUI," which serves as the central interface. The user can perform two primary actions: initiating the "start_monitor_mode" (executed via the run_snort.sh script) to begin monitoring network traffic, and triggering the "stop_monitor_mode" (via the stop_snort.sh script) to halt the monitoring process. These actions communicate with the "Detection Engine," which is responsible for analyzing network traffic using Snort's rule-based system. When the Detection Engine identifies suspicious activity, it "sends alert directly to GUI," which the user can view in real-time. Additionally, the Detection Engine logs these alerts to the "Logs" component (likely the alert_fast.txt file), allowing for persistent storage and later analysis. This setup ensures the user can efficiently control the IDS and respond to detected threats through a streamlined interaction flow.

3.4 Sequence Diagram



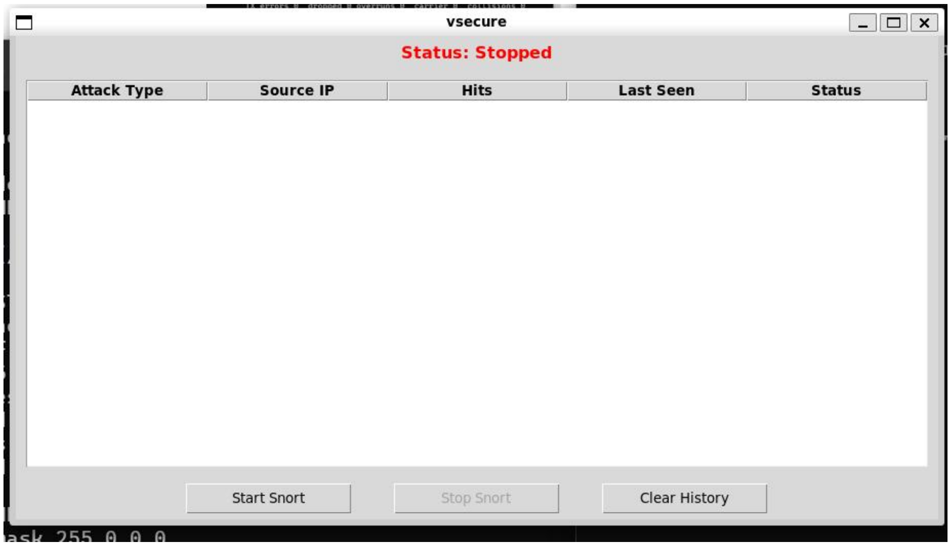
3.5 Interface/ScreenShots



```
┌──(root㉿DESKTOP-S3N3D2G)-[/etc/snort]
└─# ls
balanced.lua          inline.lua       rules                snort.debian.lua    start_snort_gui.sh
community-sid-msg.map logs             run_snort.sh         snort_defaults.lua  stop_snort.sh
connectivity.lua      max_detect.lua   security.lua         snort_gui.py        talos.lua
file_magic.rules      README.md        sensitive_data.rules snort.lua

┌──(root㉿DESKTOP-S3N3D2G)-[/etc/snort]
└─# python snort_gui.py
```

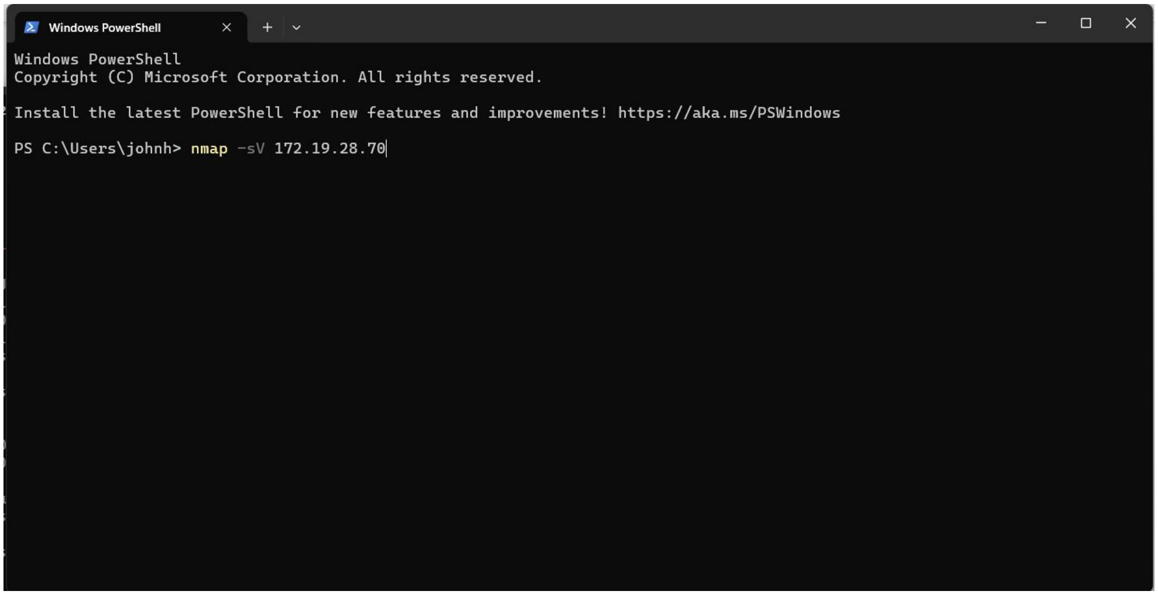
a)The WSL Environment(IDS Machine)



Status: Stopped

Attack Type	Source IP	Hits	Last Seen	Status

b) Tkinter Interface for detection(Main Interface)

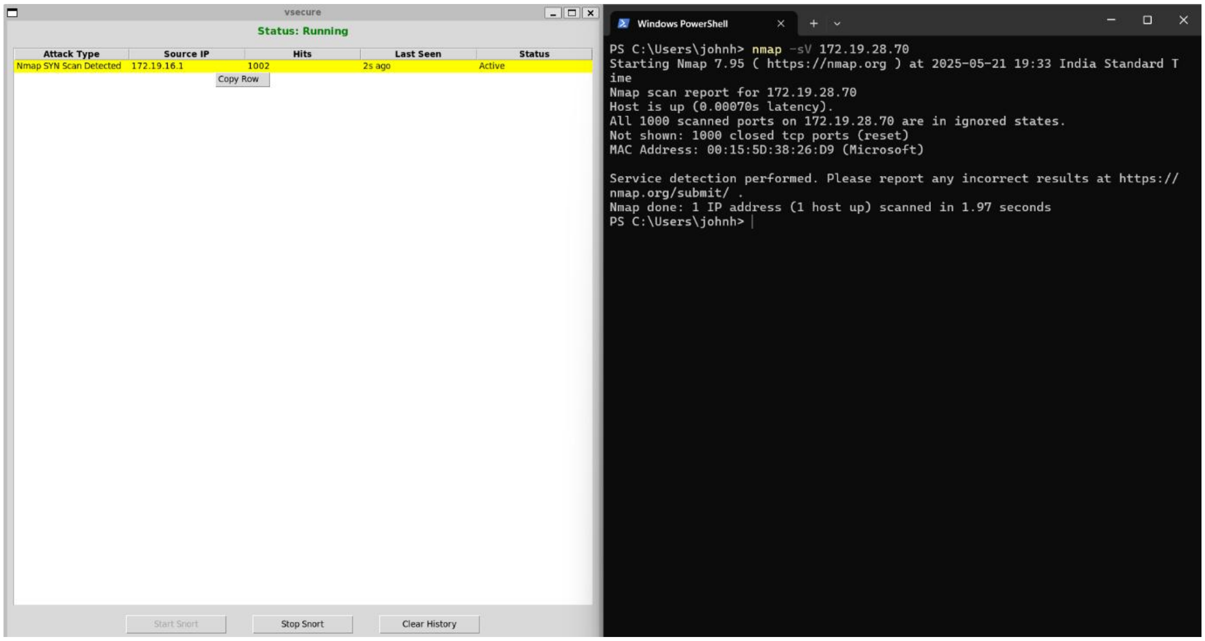


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\johnh> nmap -sV 172.19.28.70
```

c) The Windows Machine (The Attacker Machine)



Status: Running

Attack Type	Source IP	Hits	Last Seen	Status
Nmap SYN Scan Detected	172.19.16.1	1002	2s ago	Active

```
PS C:\Users\johnh> nmap -sV 172.19.28.70
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-21 19:33 India Standard T
ime
Nmap scan report for 172.19.28.70
Host is up (0.00070s latency).
All 1000 scanned ports on 172.19.28.70 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 00:15:5D:38:26:D9 (Microsoft)

Service detection performed. Please report any incorrect results at https://
nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.97 seconds
PS C:\Users\johnh>
```

d) Detection Ongoing(Status: Running)

4. PROJECT IMPLEMENTATION

4.1 Description of the Software Used

For the implementation of the Intrusion Detection System (IDS) focusing on detecting Nmap scans and ping attempts, we used the following software components and tools:

i) Snort IDS:

- An open-source network intrusion detection and prevention system (NIDS/NIPS).
- Capable of real-time traffic analysis and packet logging on IP networks.
- Utilizes a rule-based language to define patterns and signatures of known threats and suspicious activities.
- Our custom IDS rules were developed to detect specific Nmap scanning techniques (NULL, FIN, XMAS, SYN scans) and ICMP ping traffic.

ii) LibDAQ (Data Acquisition Library):

- A critical library used by Snort for packet acquisition and network traffic capturing.
- Provides abstraction for capturing packets from network interfaces.
- Ensures efficient and reliable packet collection from different network types, supporting the IDS's core packet processing.

iii) CMake:

- A cross-platform open-source tool for managing the build process of software using compiler-independent configuration files.
- Used to configure and build Snort and its dependencies on the development system.
- Ensures reproducible and manageable compilation and installation processes.

iv) Nmap:

- A powerful open-source network scanning tool used to test the effectiveness of IDS rules.
- Generates various types of network probes such as SYN, NULL, FIN, XMAS scans, and OS fingerprinting to simulate attacker behavior.

- Provides a practical way to validate and fine-tune IDS detection capabilities against known reconnaissance techniques.

v) Linux Operating System:

- The IDS and testing environment is based on Linux (Ubuntu/Debian) for stability and network tool compatibility.
- Provides native support for networking, packet capturing tools, and open-source IDS software.

5. Testing / Result Analysis

5.1 Types of Testing

To ensure the Snort IDS rules work effectively for detecting various Nmap scan attempts and ping, the following types of testing should be performed:

i)**Signature Testing**:

- Test each IDS rule against the specific Nmap scan or traffic pattern it is designed to detect.
- Ensures that the rule triggers alerts only when the relevant suspicious traffic is present.

ii) **False Positive Testing**:

- Validate that normal, legitimate network traffic does not trigger alerts falsely.
- Helps in tuning the rules to reduce noise.

iii)**Cross-Network Testing**:

- Simulate attacks from different networks or hosts to ensure detection is network independent.
- Confirms IDS rules can detect remote scans and pings accurately.

iv) **Performance Testing**:

- Assess the resource usage of Snort when processing network traffic with these rules enabled.
- Ensures that the IDS does not degrade network performance significantly.

5.2 Test Cases

i)**Signature Test - Nmap OS Fingerprinting:**

- **Test Case:** Run an Nmap OS fingerprint scan (e.g., `nmap -O <target_IP>`) from an attacker machine.
- **Expected Result:** Snort triggers an alert with the message "Nmap OS Fingerprint Attempt."

ii)**Signature Test - Nmap NULL Scan:**

- **Test Case:** Run an Nmap NULL scan (e.g., `nmap -sN <target_IP>`) from a remote host.
- **Expected Result:** Snort triggers an alert with the message "Nmap NULL Scan Detected."

iii)**Signature Test - Nmap FIN Scan:**

- **Test Case:** Run an Nmap FIN scan (e.g., `nmap -sF <target_IP>`).
- **Expected Result:** Snort triggers an alert with the message "Nmap FIN Scan Detected"

iv)**Signature Test - Nmap XMAS Scan:**

- **Test Case:** Run an Nmap XMAS scan (e.g., `nmap -sX <target_IP>`).
- **Expected Result:** Snort triggers an alert with the message "Nmap XMAS Scan Detected."

v)**Signature Test - Nmap SYN Scan:**

- **Test Case:** Run an Nmap SYN scan (e.g., `nmap -sS <target_IP>`).
- **Expected Result:** Snort triggers an alert with the message "Nmap SYN Scan Detected."

vi)**False Positive Test - Normal Traffic:**

- **Test Case:** Generate normal TCP, UDP, and ICMP traffic without scanning or pinging.
- **Expected Result:** No Snort alerts are generated.

vii)**False Positive Test - Normal Traffic:**

- **Test Case:** Generate normal TCP, UDP, and ICMP traffic without scanning or pinging.
- **Expected Result:** No Snort alerts are generated.

viii) **Cross-Network Test - Remote Scan Detection:**

- **Test Case:** Perform each of the above Nmap scans and ping from a host on a different network than the Snort sensor.
- **Expected Result:** Snort detects and alerts on all scan and ping attempts, confirming remote detection capability.

6. Achievements

This project was recognized during **IBM Technovate–2025**, a two-day flagship event held at Assam down town University on April 10–11, 2025, centered around the theme *"Industry Graded Sessions by Tech Experts."* The event featured workshops, expert talks, and a student project showcase judged by IBM professionals including **Mr. Aman Bakshi**, **Mr. Rahul Batra**, and **Mr. Gaurav Singh**.

Out of numerous participating teams, my project—focused on an Intrusion Detection System (IDS) with real-time threat monitoring and Python-based usability—was selected as one of the outstanding student projects. In recognition of its innovation and applicability, I was awarded **IBM-branded goodies and an IBM bag** during the closing ceremony. This acknowledgment highlighted the practical relevance and technical depth of the system in a real-world context.

7.1 Conclusion

This Software Requirements Specification (SRS) document outlines the detailed requirements for the design and development of an Intrusion Detection System (IDS). The primary goal of the system is to monitor network traffic, identify suspicious activities, and alert administrators to potential security threats in real time.

Through this document, we have clearly defined the functional and non-functional requirements, detailed the system features, and identified the stakeholder expectations. The IDS is designed to be scalable, efficient, and capable of adapting to evolving security threats. Emphasis has been placed on performance, usability, and security to ensure that the system meets organizational standards and industry best practices.

7.2 References:

1. **Snort Official Documentation:** **https://www.snort.org/documents#OfficialDocumentation**
2. **Basic Snort Rules Writings:** **https://docs.snort.org/rules/**
3. **WSL Environment:** **https://www.kali.org/docs/wsl/wsl-preparations/**
4. **Python Tkinter**: **https://www.geeksforgeeks.org/python-tkinter-tutorial/**
5. **Nmap Documentation:** **https://nmap.org/book/man.html**