APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 6 No. 2 (2025) 605-624
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

AI-based Intrusion Detection System (IDS) using Behaviour Analysis

Goh Teck An[1], Isredza Rahmi A Hamid[1]*

[1] *Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA*

*Corresponding Author: rahmi@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2025.06.02.033

Article Info

Received: 20 July 2025
Accepted: 19 November 2025
Available online: 30 November 2025

Keywords

Intrusion Detection System (IDS), Deep Learning, Long Short-Term Memory (LSTM), Behavior Analysis

Abstract

A rise in cyber threats, such as Denial of Service (DoS) attacks, has highlighted the limitations of traditional Intrusion Detection Systems (IDS) in Malaysia's cyberspace. Existing IDS face difficulties in detecting new threats due to reliance on signature databases. To address these issues, an AI-based IDS utilizing Long Short-Term Memory (LSTM) networks and behavior analysis techniques is proposed. LSTM improves anomaly detection by learning traffic patterns over time, while behavior analysis enhances accuracy by focusing on deviations from normal network behavior. Agile methodology is used in the tool development to ensure continuous improvements. The tool consists of three modules: monitoring module, anomaly detection module and user interface module. The AI-based IDS tool aims to provide real-time threat detection for individuals and organizations. Testing was conducted using two scenarios – benign-only traffic and simulated DDoS traffic, to validate detection accuracy and system responsiveness. The results show the tool is having an accuracy of 90.32% in real-time DDoS detection. The outcome is an AI-based IDS capable of real-time threat detection, enhancing network security for individuals and organizations.

1. Introduction

In the field of cybersecurity, the demand for innovative approaches to safeguarding networks and systems has increased due to the ongoing evolution of cyber threats. Based on the Cyber Incident Quarterly Summary Report Q2 year 2024 [1], the amount of intrusion incidents had been increased from 97 cases at Q1 year 2024 to 104 cases at Q2 year 2024, while the amount of intrusion attempts incidents had been increased from 65 cases to 106 cases at the same period. This data shows the escalating danger of intrusion threats in Malaysia cyberspace. In order to prevent such threats from causing problems to various organizations, an intrusion detection system (IDS) supported by advanced technologies such as artificial intelligence (AI) and behavior analysis should be developed.

An IDS is a network security technology originally built for detecting vulnerability exploits against a target application or computer [2]. In general, the detection methods of IDS can be divided into two categories: misuse (signature-based) detection, which identifies known attack patterns, and anomaly detection, which uses AI algorithms to identify novel and previously unseen threats [3]. The effectiveness of traditional IDS [4], [5], [6] can be limited when encountered with rapidly growing cyber threats. They rely on static rules, which may struggle to detect new threats, and generate high false positive rates, leading to potential security gaps and alert fatigue among security teams [2].

To enhance the effectiveness of common IDS, advanced technologies such as artificial intelligence (AI) and behaviour analysis will be integrated into the security framework. AI enables IDS to analyze large datasets, identify complex patterns, and adapt to evolving threats in real-time, which behavior analysis helps distinguish normal behavior from anomalies. A user-friendly interface with intuitive navigation and visualizations will be provided to enhance usability. The objectives of this project are to design an Intrusion Detection System (IDS) using dynamic approach, to develop an AI-based IDS by incorporating behavior analysis and to evaluate the tool based on system functionality and user requirements.

The AI-based IDS is designed to be accessible to users across various backgrounds, from IT departments to non-technical individuals. It focuses on detecting a key cyber threat: Denial of Service (DoS) attacks by using Long-Short Term Memory (LSTM) detection model and behavior analysis. The tool enhances its ability to detect anomalies and respond to threats in real time. The development of the tool using Python and TensorFlow. There are four key modules included in the AI-based IDS which are monitoring and data collection, anomaly detection, deep learning model management, and user-friendly interface for easy management. The tool will continuously monitor network traffic, comparing it to established normal baselines to generate alerts for any suspicious activity. To improve its compatibility, the tool is designed to run efficiently in the background without affecting system performance. This project goal is to contribute to the advancement of intrusion detection technologies by integrating AI and behavior analysis, addressing current gaps in cybersecurity, and providing a scalable and effective solution for various environments.

The remaining of this paper is organized as follows: Section 2 provides a comprehensive review of related works, discussing IDS and integration of AI algorithms and behaviour analysis. Section 3 outlines the methodology employed in developing the AI-based IDS. Section 4 discusses about the system analysis and design. Finally, Section 5 concludes the current work and highlight future contribution.

2. Related Work

This section discusses the related work that has been conducted for the development of AI-based IDS, including types of attacks, purpose and classification of IDS, exploration for LSTM networks and behavior analysis, and reviews of existing IDS solutions.

2.1 Denial of Service (DoS) Attack

DoS attack is described as an attack happening when authorized users are unable to access information systems, devices, or other network resources due to the actions of malicious actors. Typically, DoS attack can result in the unavailability of a particular network service or the temporary loss of all network connectivity and services. The most common method of DoS attack on the Internet occurs when the attacker floods a network server with a large amount of traffic. Smurf Attack and SYN flood are the examples of this type of attack. Various symptoms or anomalies could indicate the DoS attack, including unusually slow network performance, unavailability of certain websites, or inability to access to any website.

DoS attack is especially dangerous because it can easily disrupt critical online services such as banking, healthcare systems, government platforms, or business websites. This can causing serious financial and operational damage to the organizations. If not detected early, DoS attacks may lead to prolonged service downtime and damage organization's reputation. In some cases, attackers may use DoS attacks as a distraction to carry out more serious intrusions in the background. Thus, early detection of DoS attacks is important to ensure service availability, maintain user trust, and reduce potential losses. The best way to detect and identify DoS attacks and particularly DDoS attacks is through the network traffic monitoring and analysis.

Work by [7] using machine learning (ML) techniques to detect port scanning and Distributed Denial of Service (DDoS) attacks by analyzing network traffic in the IDS. They utilized CICIDS2017 dataset which contains realistic and recent attack vectors in the model training and testing for the system. As a result, most algorithms including the variants of tree, discriminant analysis, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and ensemble classifier achieved over 90% accuracy in detecting DDoS attacks while 9 out of 22 algorithms are having a classification accuracy of 85% or higher. This study highlights the potential of machine learning in effectively distinguishing between normal traffic and attack traffic.

Work by [8] proposed an IDS using Deep Learning (DL) techniques to identify DDoS attacks in network environments. Their approach included preprocessing steps such as feature elimination, duplication removal, and normalization before feeding the data into various deep learning models, including Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), and Long Short-Term Memory (LSTM). The CIC-DDoS2019 dataset was used for training and evaluation. As a result, the CNN-based inception-like model achieved the highest accuracy

with 99.99% for binary classification and 99.30% for multiclass classification. This work demonstrated the effectiveness of combining data preprocessing with deep learning to improve accuracy for DDoS detection.

Work in [9] analyzes cybersecurity incidents occurred from 2013 to 2023, focusing on major threats such as DDoS and malware attacks. This study highlights a significant rise in the frequency and sophistication of DDoS attacks, particularly in 2022, with predictions using the AutoRegressive Integrated Moving Average (ARIMA) model indicating a continued increase over the next five years. This finding emphasizes the need for adaptive and data-driven cybersecurity strategies to address evolving challenges effectively.

A work from [10] proposed a DDoS detection tool utilizing Artificial Neural Networks (ANN) tailored for homelab environments. The system was designed to offer an effective yet lightweight IDS solution for users with limited resources. Their approach focused on accurately identifying DDoS attacks while minimizing false positives, ensuring smooth server performance. The tool finally achieved a recall and F1-score of 0.97, showing the feasibility of implementing ANN-based DDoS detection in small-scale, real-life deployments without compromising detection capability.

2.2 Intrusion Detection System (IDS)

In general, IDS is a security tool that continuously monitors the host and network traffic to detect any potential attempts to penetrate the system. If malicious behavior is detected, the IDS will generate warnings or notifications to the host and network administrators. IDS is an early warning system which helps organizations to monitor network activities in real time. When any potential attack is detected, it can provide sufficient time for the organizations to investigate and respond to those security incidents. Besides, IDS will log and record all the data about the incidents which can be utilized by the security team for post-incident analysis, enabling them to understand the nature of the attack and strengthening the overall security of the system.

2.2.1 Classification of IDS

Typically, IDS are classified based on two aspects, its deployment method, and its detection method. A classification taxonomy is shown in Fig. 1.

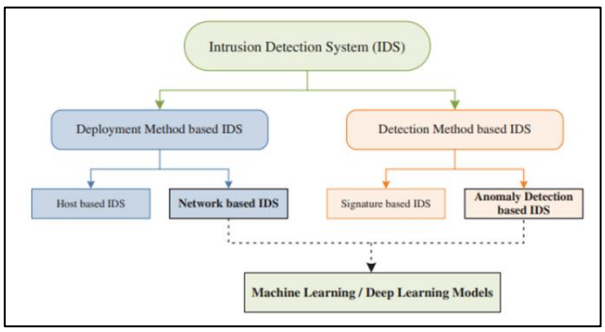

Fig. 1 *Intrusion Detection System Classification Taxonomy [11]*

In the deployment perspective, IDS can be subclassified as host-based IDS (HIDS) or network-based IDS (NIDS). HIDS is deployed on a single host machine, and it will monitor and examine the audit data from this machine and derive their conclusions based solely on that information. Thus, the organization may need to deploy this type of IDS to every host that requires intrusion protection, which can overwhelm the network performance for each node and affect the IDS efficiency. In contrast, NIDS is deployed within the organization's network to passively monitor the network for suspicious activity. It can protect all devices inside the network from intrusions at the same time. This kind of IDS depends solely on the TCP/IP protocol suite. Thus, it is architecture-independent and able to monitor heterogeneous networks in nature.

Signature-based IDS (SIDS) detect known attacks by matching patterns or signatures stored in a database. While SIDS are highly efficient at detecting known attacks, they require constant updates to the signature database. Without regular updates, novel or new attacks may go undetected. Additionally, maintaining a large signature database can be resource intensive. Anomaly detection-based IDS (AIDS), on the other hand, identifies abnormal behavior relative to a predefined normal profile of a user or system. It can detect unknown and new attacks by comparing real-time activities with this baseline. However, AIDS may have a high False Acceptance Rate (FAR) due to the difficulty in defining clear boundaries between normal and abnormal behavior.

2.3 Deep Learning Algorithm

Deep Learning (DL) is a branch of Artificial Intelligence (AI) that focuses on enabling computers to learn from examples, similar to how humans learn. Instead of relying on manually crafted rules, DL models use multiple layers of artificial "neurons" in a large neural network to automatically discover patterns hidden in raw data. By leveraging this layered learning capability, DL models perform extremely well in tasks that needs abstract analysis and understanding of complex behaviours. Long Short-Term Memory (LSTM) and Transformer are well-known examples of DL models.

2.3.1 Long Short-Term Memory (LSTM)

LSTM is one of Recurrent Neural Network (RNN) architecture that was introduced by Hochreiter and Schmidhuber in 1997 [12]. This model has specialized hidden units that inherently tend to preserve inputs over an extended period. The LSTM architecture involves the memory cell that is handled by three gates: the input gate, the forget gate, and the output gate [13]. Networks in LSTM architecture can be stacked to generate deep architecture, enable it to learn even more complex patterns in sequential data.

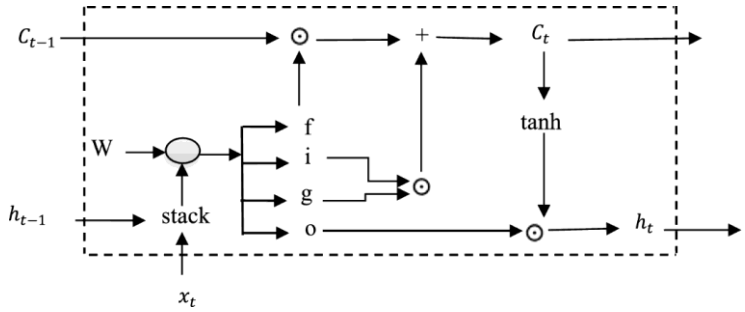

Fig. 2 *LSTM architecture [13]*

Fig. 2 shows the basic LSTM architecture. LSTM has a chain structure that consists of four neural networks and memory blocks called cells. The gates in an LSTM network manage the flow of information from the previous cell by controlling how much data is added or removed. At each step, the model maintains two states: the hidden state (***h***) and the cell state (***c***). The cell state is represented by the line running at the top of the structure and it will carry long-term memory. This state can preserve past data over long sequences. The hidden state is shown as the bottom line, and it will carry short-term memory. This state will directly affect the output at each step since its value will be passed into the weight (***W***) together with the next input to determine the next cell state. The forget gate (***f***) will determine how much information the previous cell state will forget. The input gate (***i***) will be combined with candidate cell state (***g***) to decide what new information to be stored in the cell state. The output gate, ***o*** will be used to check how much information from the cell state will be passed to the hidden state and used as output. This architecture allows LSTM to effectively manage long-term dependencies, making it suitable to perform tasks that involve sequential data.

2.3.2 Transformer

Transformer is neural network architecture designed to process sequential data efficiently. Instead of processing data step by step like LSTM, Transformer process entire sequences in parallel, which makes them become faster and suitable to handle long-range dependencies. Generally, Transformer will consist of these primary components: tokenizers, embedding layer, transformer layers and un-embedding layer. Tokenizers will convert every input into tokens or embeddings before sending them into the block. Next, the embedding layer will convert these tokens and their positions into vector representations. Transformer layers can be further divided into encoder layers and decoder layers, both layers will work together to perform transformations on the vector representations and extract more information. At the end, the final vector representations will be converted back to a probability distribution at un-embedding layer. Due to its ability to capture relationships between information, the Transformer has become one of the popular DL models that are used in language translation, anomaly prediction and large-scale models.

2.3.3 Related work in Deep Learning Algorithm

Deep learning models have gained significant attention due to their ability to learning complex patterns from large volumes of data. Among these models, LSTM and Transformer have shown promising results when handle sequential data. LSTM is learning long-term dependencies through its unique memory cell structure, which allowing it to remember important features over time while Transformer used a self-attention mechanism that enables it to focus on the most relevant parts of the sequence without relying on sequential processing. Various work used LSTM and Transformer models individually or in combination to enhance detection performance. Work by [14] shows that LSTM-based model outperforms the Transformer-based model and self-attention model in the context of multi-agent navigation and obstacle avoidance, provide a better balance through several sequence lengths. In addition, work by [15] shows that Bi-directional LSTM has an excellent result of 99.70% accuracy for KDDCUP-99 dataset and performs better than other algorithms such as Ada Boost, Decision Tree, and K-Nearest Neighbour. In conclusion, LSTM models are good in tasks with clear succession and being uniquely superior at handling long-term dependencies. Beside that, another work from [16] presents a comparative study of Advanced Persistent Threat (APT) detection using Multilayer Perceptron (MLP) and Naïve Bayes algorithms. APT attacks are particularly dangerous due to their stealth and long-term presence in targeted systems. The results of the study indicate that MLP achieved a higher true positive rate (TPR). This study showing the superior accuracy of deep learning approaches in detecting APTs. Based on these works, it's clear that LSTM can effectively model sequential data and capture long-term dependencies. The specialized memory cell architecture helps LSTM networks to have high accuracy in detection of various types of attacks such as DoS and APT that unfold over time. Besides, LSTM can adaptively store and forget information, which gives it a distinct advantage in tasks where patterns emerge gradually or span across long sequences. However, there are some disadvantages can be observed on LSTM models. This model commonly has high computational intensity where it may require more processing power and longer training times compared to simpler models.

2.4 Behavior Analysis

Behavior analysis approach is utilized for searching and modelling the behavioral patterns of the network traffic to generate a baseline which is called normal behavior. This baseline will then be used to detect any unusual activities, whether those are caused by malicious actors, insiders, or malicious software. This approach plays an important role in minimizing the potential damage by allowing security teams to quickly detect and respond to cyber-attacks, and it does not have the ability or functionality to prevent or block them from happening. After the initial discovery, the security teams may apply human judgement, comprehensive understanding of the environment and organization to give additional analysis to determine whether the anomaly was suspicious.

User Behavior Analytics (UBA) and User and Entity Behavior Analytics (UEBA). Are behavior analysis techniques used for threat detection [17]. UBA focuses on monitoring user activities within a network to identify anomalies, such as unusual data downloads, accessing confidential data, or login from unfamiliar locations. UBA is effective in detecting insider threats, compromised credentials and fraud, primarily involving human actors. However, it has limitations like lower visibility into network activities and a narrower scope compared to UEBA.

UEBA is an extension of UBA, analyses not only user behavior but also the behavior of other entities in the network, including devices, hosts, applications, and data repositories. By incorporating AI and machine learning, UEBA can detect a broader range of threats, including hidden attacks that masked by legitimate activities. UEBA often integrates with Security Information and Event Management (SIEM) systems to provide enhanced analysis. Table 1 shows the summary of the pros and cons of UBA and UEBA. Both techniques can be utilized to enhance cybersecurity by analyzing behavior patterns, but there are several key differences that make UEBA a more advanced and comprehensive solution. UBA focuses only on monitoring human user behavior based on event logs, which limits its detection capabilities to threats involving people or malware. In contrast, UEBA extends this by also analyzing entities such as devices, applications, and network. Furthermore, UEBA can provide more visibility and deeper insights for threat detection and investigation while seamlessly integrate into existing security tools and systems. This is a major improvement over UBA, which typically relies on limited event log data and offer less visibility into network activities while facing compatibility challenges. In general, UEBA will be the better approach if compared to UBA

Table 1 *Comparison of UBA and UEBA [17]*

UBA	UEBA
Focuses on user behavior only	Covers both user and entity behavior
Relies on event logs only	Integrates data from multiple sources
Can detect threats involve human actors or malware	Can detect threats involve human actors, malware, or machine actors
Provides limited visibility into network	Provides more visibility into context for threat investigation
May not integrate well with existing security products and systems	Can integrate seamlessly with existing security products and systems

2.5 Study on Related Systems

This section will explore the key features and limitations of various open-source IDS such as Snort, Suricata, and Zeek. A comprehensive understanding of existing IDS will be useful in developing better IDS systems. By analyzing their strengths and weaknesses, some areas for improvement can be identified to design a good IDS solution.

2.5.1 Snort

Snort was a Command Line Interface (CLI) system created by Martin Roesch in 1998 [4]. It consists of eight different types of modules which are: basic modules, codec modules, inspector modules, IPS action modules, IPS option modules, search engines, and Shared Object (SO) Rules. It uses a rule-based language that combines anomaly, protocol, and signature inspection methods to detect potentially malicious activity. Snort can perform network inspection in various ways, including reading traffic directly from a packet capture, running passively on a network interface to sniff traffic, and act as IDS and IPS for real-time threat detection and prevention. In most situations, Snort will provide real-time protection to the users, except it is configured into packet logger mode which will only logs packets without immediate analysis. Snort primarily operates using signature-based detection and it only relies on the predefined rules to identify threats.

2.5.2 Suricata

Suricata was a signature-based network intrusion detection engine that was released by Open Information Security Foundation (OISF) in 2009 [5]. Suricata is primarily a command-line tool that enables real-time monitoring and analysis of network traffic to detect and respond to various types of cyber threats and attacks. Suricata can be used in many ways including network traffic analysis, intrusion detection and prevention, protocol analysis, file extraction and analysis, traffic pattern detection, network flow monitoring, and log generation and reporting. Suricata relies on signature-based detection to detect threats, but it also consists of limited anomaly detection techniques to identify suspicious activity based on protocols' behavior.

2.5.3 Zeek

Zeek is an open-source NIDS tool that only supports IDS mode [6]. Zeek utilize both signature-based and anomaly-based detection for network traffic monitoring. Zeek can log into the network activity in detail including application-layer transcripts such as Hypertext Transfer Protocol (HTTP) sessions with URIs, key headers and Domain Name Server (DNS) requests with replies. In addition, Zeek consists of built-in functionality for real-time analysis and detection tasks, such as HTTP sessions file extraction, SSH brute-forcing detection and SSL certificate chains validation. Zeek cluster features make it suitable for implementation in major corporations where it supports scalable load-balancing.

2.6 Comparison between Existing IDS Systems and AI-based IDS

Table 2 shows the comparison between existing IDS with the AI-based IDS. For the detection technique, Snort relies on signature-based detection, while Suricata and Zeek utilize hybrid detection. The AI-based IDS will be using anomaly detection and integrates it with LSTM networks to increase its accuracy. Due to their reliance on signature-based detection, Snort and Suricata will be facing issues to adapt to novel attacks, while Zeek and the AI-based IDS will have higher adaptability to these threats. Snort and Suricata can consume large amounts of resources for storing and matching the signatures. On the other hand, Zeek and the AI-based IDS will consume less resources since both utilized anomaly detection which usually have lower resource consumption. For the

False Positive Rate (FPR), Snort has the highest FPR following Suricata, they have high FPR since they are relying on signature-based detection which can generate large amount of false positive with improper rule configuration. Apart from that, Zeek is having lower FPR since it utilizes both signature-based and anomaly detection, which enhances its accuracy and reduces its false positive rate. The AI-based IDS is having a potentially lower FPR from the existing system by associated with AI algorithms and behavior analysis.

Table 2 *Comparison table between reviewed system with the AI-based IDS*

Features/System	Snort	Suricata	Zeek	AI-based IDS
Detection Technique	Signature-based	Hybrid	Hybrid	Anomaly-based
Adaptability to New Attacks	Low	Medium	High	High
Resource Consumption	High	High	Moderate	Moderate
False Positive Rate	95.5% [18]	74.3% [19]	20% [20]	12.8%
Integration of AI Algorithms	No	No	No	Yes
Integration of Behavior Analysis	No	Yes	No	Yes
User Interface	CLI	CLI	CLI	GUI
Real-Time Alerting	Yes	Yes	No	Yes
Logging and Output	Yes	Yes	Yes	Yes
Ease of Use	No	No	No	Yes

3. Methodology

This section outlines the methodology for the tool, including the chosen method and requirements of software and hardware.

3.1 Agile Model

The Agile model [21] is a software development approach that emphasizes flexibility, collaboration, and iterative progress. Unlike traditional waterfall models which have linear and sequential approaches, the Agile model is having iterative approach which promotes the development of software in multiple short cycles called sprints. Fig. 3 shows the Agile SDLC. Agile model can be divided into six phases such as Requirements, Design, Development, Testing, Deployment, and Review. Each phase focuses on specific tasks to ensure the project moves forward efficiently. In Requirements phase, requests from stakeholders are gathered by the developers to identify the core features and functionalities of the system, the requirements collected are often flexible and may evolve over time based on feedback. Design phase is focusing on planning how the system will be built, this includes architecture design, tools or framework selection, and outlining component interaction. After that, Development phase is the actual coding part, where system modules, interfaces, and necessary models has to be developed. Testing phase is used to test and verify the functionality of the system after the development to ensure it meets the expected requirements. Next, the system will be deployed in Deployment phase for user testing in order to gather the user feedback. Lastly, at Review phase, the reviews from the previous phase will be used to further improve the system in the next sprint.

The development process had been breaking into manageable iterations, enabled developers to respond to changing requirements and incorporate feedback quickly. The concept of sprints enables the teams to continuously improve their software to achieve better customer satisfaction.

Fig 3 *Agile SDLC [22]*

3.2 Software and Hardware Requirements

During the system development, specific software and hardware has been used to ensure optimal performance and efficiency. The software used in the development is Visual Studio Code and TensorFlow. Visual Studio Code serves as the integrated development environment (IDE), while TensorFlow provides the deep learning framework for building and training the LSTM model for threats detection. The hardware requirements include a computer with an Intel Core i5-1035G1 CPU, 20GB of RAM, and 1TB of SSD storage.

4. System Analysis and Design

In this section, the analysis and design of the AI-based IDS will be explained in detail, covering the system development workflow, system requirements, Unified Modelling Language (UML) diagrams, system architecture, and user interface of the project.

4.1 System Development Workflow

The application development workflow for each phase of the system development process will be shown in the Table 3. The flow of activities and their respective outputs will be listed in detail to provide a comprehensive overview of the system development process.

Table 3 *Application development workflow*

Phase	Description	Task	Output
Requirements	Defines the project's direction by gathering relevant information, analysing feasibility, and outlining key objectives.	1. Conduct research on existing IDS 2. Analyze the feasibility and scope of the tool 3. Prepare the proposal for the tool 4. Construct a Gantt chart based on the Agile methodology	1. A proposal with well-defined problem statements, objectives, scopes, and expected outcomes 2. A Gantt chart to outlining task timelines
Design	Focus on translating gathered requirements into system architecture, UI designs, and technical specifications.	1. Create system architecture 2. Design User Interface 3. Define system requirements (functional, non-functional) 4. Define software and hardware requirements 5. Develop UML diagrams	1. System architecture diagrams 2. List of system requirements 3. User interface design 4. UML diagrams
Development	Implements the tool by coding the AI-based IDS, developing models, and integrating core functionalities.	1. Write source code for tool modules 2. Pre-training LSTM model with related datasets 3. Determine thresholds for potential malicious activities by considering various factors	1. An AI-based IDS with low false positive rate and high accuracy
Testing	Verifies the tool's effectiveness by exposing it to real-world threats, fixing bugs, and refining performance.	1. Prepare some datasets that contain real-world cyber threats 2. Evaluate the tool's performance and record its false positive rate 3. Fix errors through iteration process	1. Result of the tool's false positive rate 2. Error log and their respective solutions

Table 3 *(cont)*

Phase	Description	Task	Output
Deployment	Introduces the IDS to users, ensuring installation, configuration, and real-life performance adjustments.	1. Release the IDS to potential users 2. Prepare some general documents like installation and configuration instructions 3. Adjust the tool based on user review	1. Installation and configuration instructions for the AI-based IDS 2. Refined AI-based IDS with better performance
Review	Gathers user feedback to enhance the tool, ensuring it meets expectations and functions smoothly.	1. Obtain review and evaluation from users 2. Analyze their review and suggestions for the IDS 3. Improve the tool according to the user review	1. User acceptance testing form 2. Finalized AI-based IDS which ready for production environments

4.2 System Requirements

The functional and non-functional requirements of the AI-based IDS will be discussed in this section.

4.2.1 Functional Requirement Analysis

Functional Requirements define all the features and functionalities that the AI-based IDS must have to perform its intended tasks. These requirements will be acted as guidance for the development of the tool's core functionalities. Table 4 displays the essential functional requirements for the AI-based IDS. The AI-based IDS will need to be able to track network activity in real-time and able to detect anomalies in high accuracy. When any threat is discovered, the threat classification will be done to detect the nature of threat and alerts based on the type of threats will be sent to the users. For future analysis, the AI-based IDS will be able to record all network traffic into a log which can be observed by the users.

Table 4 *Functional requirements for AI-based IDS*

Functional Requirement	Description
Network Traffic Monitoring	The tool must continuously monitor network traffic in real-time, capturing data from various network sources.
Anomaly Detection	The tool should be able to detect anomalies in network traffic by comparing its behavior against the normal baseline.
Threat Identification	The tool must be able to identify DDoS attacks once an anomaly is detected.
Alert Generation	The tool must generate alerts for detected anomalies or threats, providing details such as the source and severity of the threat.
Traffic Data Logging	The tool will log onto network traffic and alert information for future analysis, auditing of the system.

4.2.2 Non-Functional Requirement Analysis

Non-functional requirements define overall quality and performance expectations for the AI-based IDS. These requirements focus on tool characteristics such as performance, reliability, and usability. It is important to ensure that this tool performs optimally under various environments and provides seamless user experience. Table 5 describes the non-functional requirements for the AI-based IDS. This tool should be able to maintain its performance with minimal delays, and able to handle growing network traffic without degradation. Besides, it should be able to operate continuously with minimal downtime and able to detect threats in high accuracy. Next, this tool must be straightforward to use and able to work across various operating systems and network environments to ensure better user experience.

Table 5 *Non-functional requirements for AI-based IDS*

Non-Functional Requirement	Description
Performance	The tool must be capable of processing and analysing network traffic in real-time, with minimal latency to ensure rapid threat detection.
Scalability	The tool should be scalable to handle increased network traffic.
Reliability	The tool must maintain a high level of availability, ensuring it operates continuously without failures or interruptions.
Accuracy	The tool must accurately identify true anomalies and threats, with a low false positive rate.
Usability	The user interface must be intuitive and easy to navigate, allowing users to quickly understand their functions
Compatibility	The tool should be compatible with a wide range of operating systems and network environments to ensure seamless deployment in various settings.

4.3 System Analysis

This section explains the system analysis for the project, including the Unified Modeling Language (UML) diagrams such as use case diagram, sequence diagram, and activity diagram. These diagrams are essential for understanding how the tool works and ensuring its design is clear and efficient.

4.3.1 Use Case Diagram

The Use Case Diagram provides a visual representation of how the user interacts with the AI-based IDS. It highlights the tool's primary functionalities and the actions available to the user. In this tool, the user can perform four key actions as shown in Fig. 4. Firstly, the user can monitor network traffic to observe real-time data and gain insights into ongoing network activities. Secondly, the user can receive alerts from the tool when suspicious behavior is detected on the network. Thirdly, the user is able to turn the system on or off, allowing for control over the operational status of the IDS based on specific needs or conditions. Lastly, the user can check logs and historical records for reviewing past network events or conducting further analysis. These core functionalities aim to support users in maintaining network security and responding effectively to potential threats.

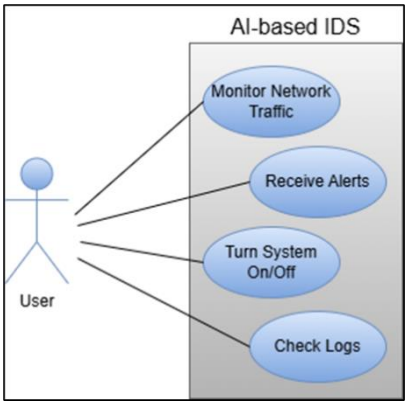

Fig. 4 *Use Case Diagram of AI-based IDS*

4.3.2 Sequence Diagram

The Sequence Diagram in Fig. 5 illustrates the flow of actions and data exchanges that occur when the user interacts with the tool. The components inside the sequence diagram will be explained in Table 6.

Table 6 *Components in the sequence diagram*

Components	Description
User	The person who interacts with the tool
User Interface	The module that displays various information to the user, such as network traffic and alerts.

Table 6 *(cont)*

Components	Description
Monitoring and Data Collection	The module that is responsible for capturing and monitoring the network traffic. It will send every captured data to an anomaly detection and notification module for analyzing process.
Anomaly Detection and Notification	The module that responsible for analyzing incoming network data using the trained LSTM model to detect potential DDoS attacks and generate alerts for the user.
Deep Learning Model Management	The module that adjusts the normal baseline and trains LSTM model.

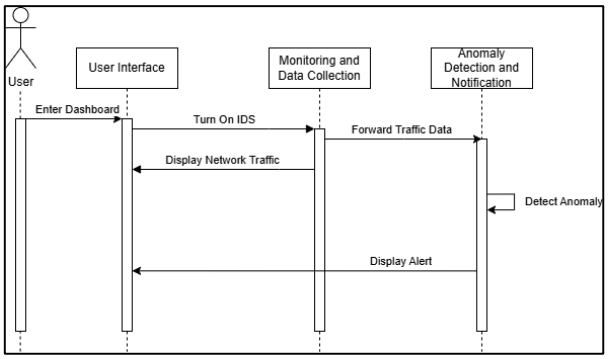

Fig. 5 *Sequence Diagram of AI-based IDS*

4.3.3 Activity Diagram

The Activity Diagram in Fig. 6 illustrates the dynamic flow of actions within the AI-based IDS. The tool will start monitoring the network traffic when the tool turns on. After network data is collected, the tool will display them to the user at real-time while sending these data for the anomaly detection. If the data is suspicious, then an alert will be sent to the users, both benign and malicious traffic data will then be stored into a log that can be used for further investigation. This tool will automatically run continuously until the user turns off the tool.

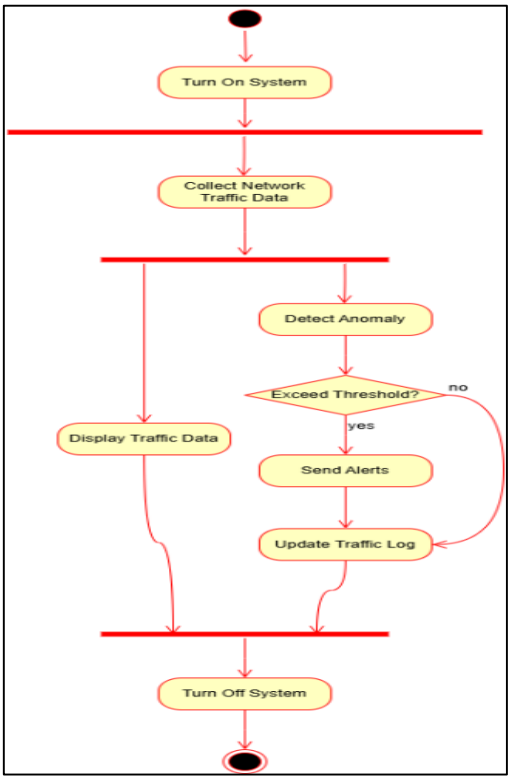

Fig. 6 *Activity Diagram of AI-based IDS*

4.4 General System Architecture

General System Architecture outlines the high-level structure of AI-based IDS. It describes how the various components interact to achieve the tool's objectives, focusing on the flow of data and the relationships between key modules. This section will provide an overview of how the tool operates, allowing for a better understanding of its functionality and design.

4.4.1 System Design Diagram

System design diagram serves as a high-level blueprint that highlights the key modules and their relationships, providing a clear overview of how the tool operates. Fig. 7 shows the system design of the AI-based IDS. The tool starts by collecting real-time network traffic through the Monitoring and Data Collection Module to generate network data and logs. This data will be passed to Anomaly Detection and Notification Module for analysis using a trained LSTM model to check whether the behavior exceeds the threshold. For DDoS detection, the model evaluates 14 features, which are 'Destination Port', 'Total Length of Fwd Packets', 'Total Length of Bwd Packets', 'Fwd Packet Length Max', 'Fwd Packet Length Mean', 'Bwd Packet Length Max', 'Bwd Packet Length Mean', 'Fwd Header Length', 'Average Packet Size', 'Avg Fwd Segment Size', 'Avg Bwd Segment Size', 'Subflow Fwd Bytes', 'Subflow Bwd Bytes', 'Init_Win_bytes_forward', as used in the dataset 2018CICIDS. Traffic will be detected as malicious if its behavior exceeds a threshold of 0.7 [23]. An alert will be sent. Then, this incident will be recorded if any anomaly is detected. If not, the data will be simply logged for future reference.

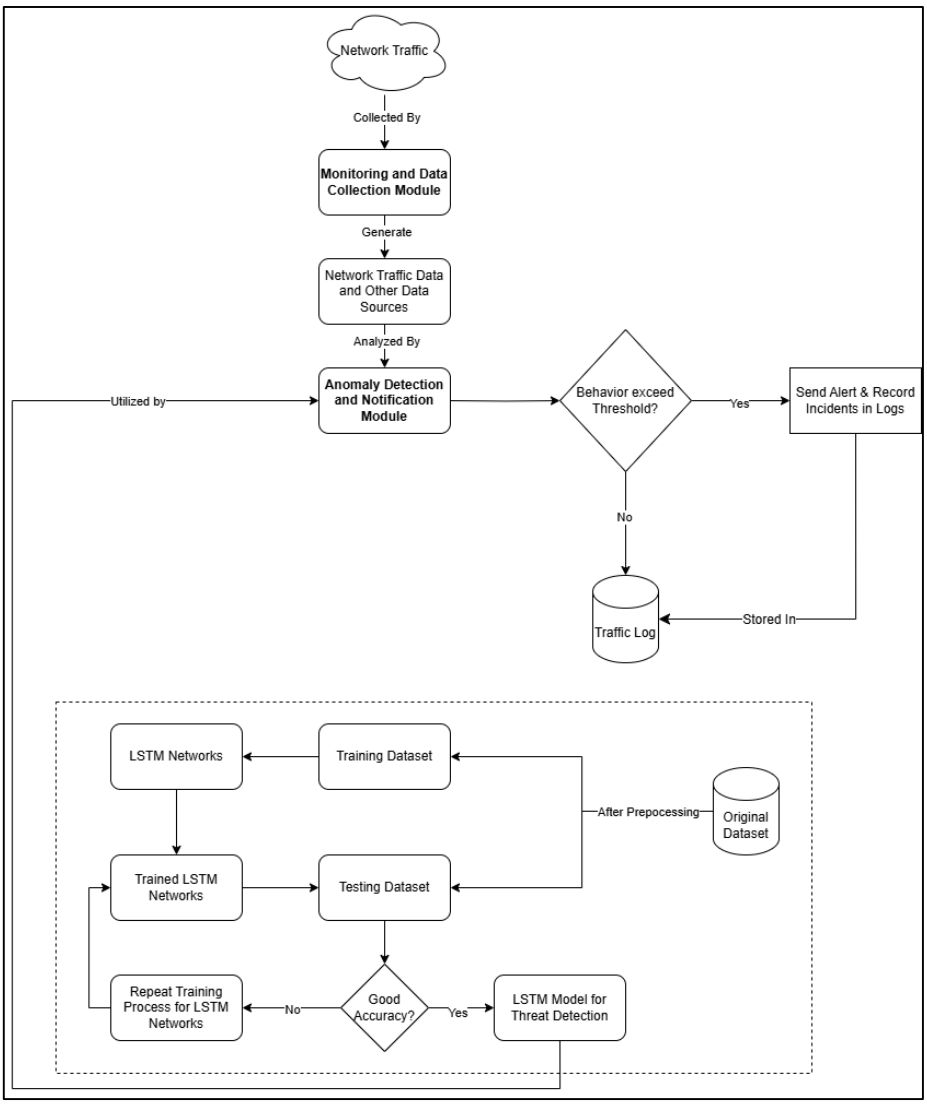

Fig. 7 *System Architecture Design of AI-based IDS*

4.5 User Interface Design

User Interface Design focuses on the design and layout of the interface through which users interact with the AI-based IDS. A well-designed interface is essential for ensuring users to have better experience with the tool. There are three key pages that have been created which are the Dashboard, Alert, and Network Monitor.

4.5.1 Dashboard Page

The dashboard serves as the main page for users, displaying an overview of the tool's status. It includes an easy-to-use "On/Off" button to start or stop the tool. Key details from other pages, such as alerts and network activity, are summarized here for quick access. A visual graph of inbound and outbound traffic is also available for a better view. Users can navigate between sections using buttons on the dashboard or the navigation bar on the left. Fig. 8 displays the dashboard page design for the AI-based IDS.

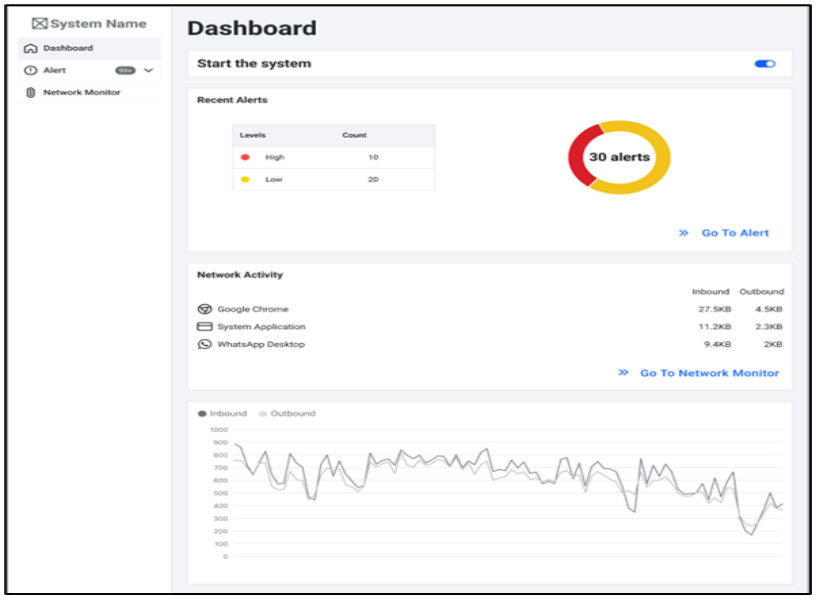

Fig. 8 *Dashboard Page Design of AI-based IDS*

4.5.2 Alert Page

The alert page in Fig. 9 focuses on providing detailed information about anomalies detected or potential threats. Each alert is listed with key details such as the sources, severity level, and number of counts. This page ensures users stay informed about suspicious activities and act against them if needed.

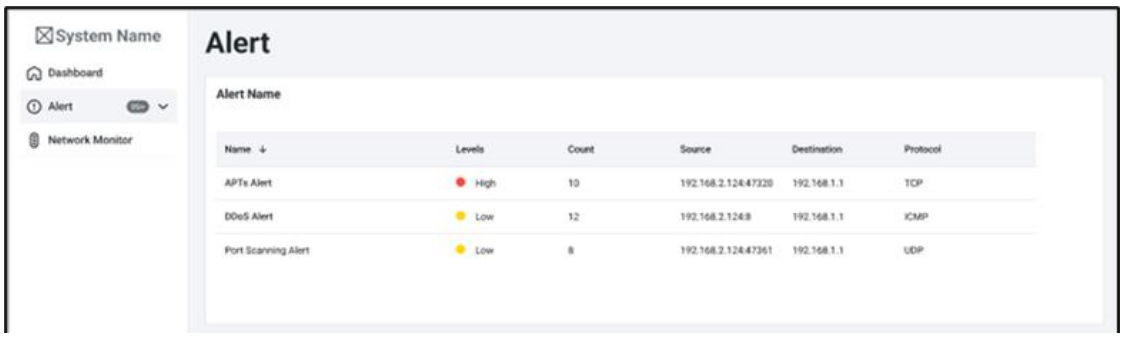

Fig. 9 *Alert Page Design of AI-based IDS*

4.5.3 Network Monitor Page

Fig. 10 represents the network monitor page that provides real-time insights into network activity. It displays detailed data, such as traffic volume, applications using the network, and traffic history from the past few days. This page enables users to analyze and monitor network activities effectively.

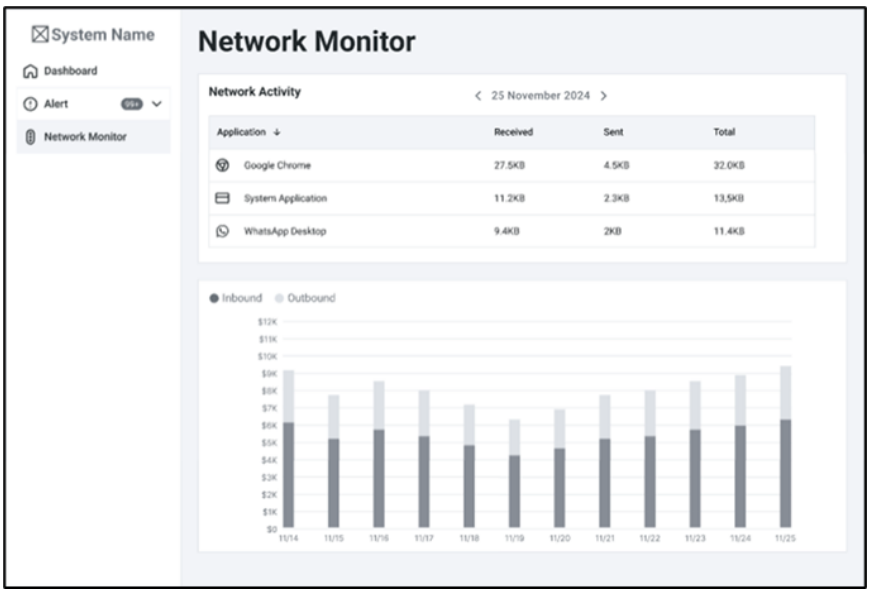

Fig. 10 *Network Monitor Page Design of AI-based IDS*

5. Results and Discussion

In this section, the results obtained from the development and evaluation of the AI-based IDS will be presented and discussed in detail. This includes an overview of the system interface, the performance of the detection model, and the insights gained through tool testing and evaluation.

5.1 System Interface Overview

The User Interface (UI) of the AI-based IDS has been designed to provide a clear, intuitive, and accessible experience for users. The tool allows users to monitor real-time network activities, receive alerts regarding detected anomalies, and review historical data logs. The following subsections present the main components of the system interface.

5.1.1 Main Dashboard

The main dashboard shown in Fig. 11 serves as the central hub of the tool, displaying real-time visualizations of network traffic and summary statistics. It provides users with an at-a-glance view of tool status, current traffic flow, and recent detections.

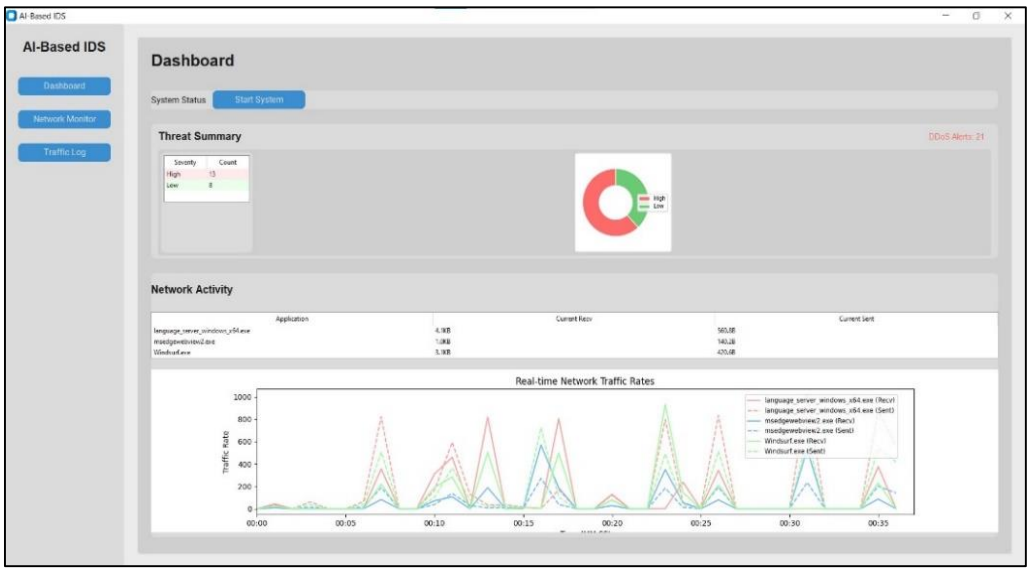

Fig. 11 *Dashboard Page of AI-based IDS*

5.1.2 Network Monitor Page

The network monitor page presented in Fig. 12 provides detailed, real-time information on ongoing network connections and traffic patterns. Users can observe the live packet flow, identify unusual activities, and cross-check against the detection results.

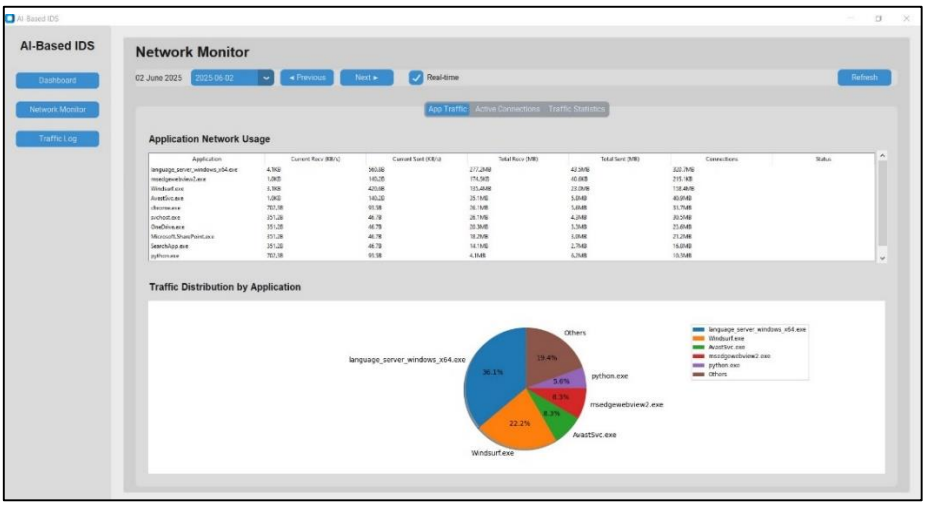

Fig. 12 *Network Monitor Page of AI-based IDS*

5.1.3 Traffic Log Page

The traffic log page displayed in Fig. 13 gives users a clear view of all traffic activities captured by the tool. The top section features an attack timeline graph, showing exactly when DDoS attacks are happening, making it easy to spot attack periods in single sight. Below that, the real-time traffic table lists key details like source IP, source port, destination IP, destination port, and the detection result. The page also includes handy options like filters (All, Benign, DDoS), a historical view to check past traffic logs, and an export button for saving the data. This setup helps users monitor and analyze network activities efficiently, without getting overwhelmed by too much technical detail.

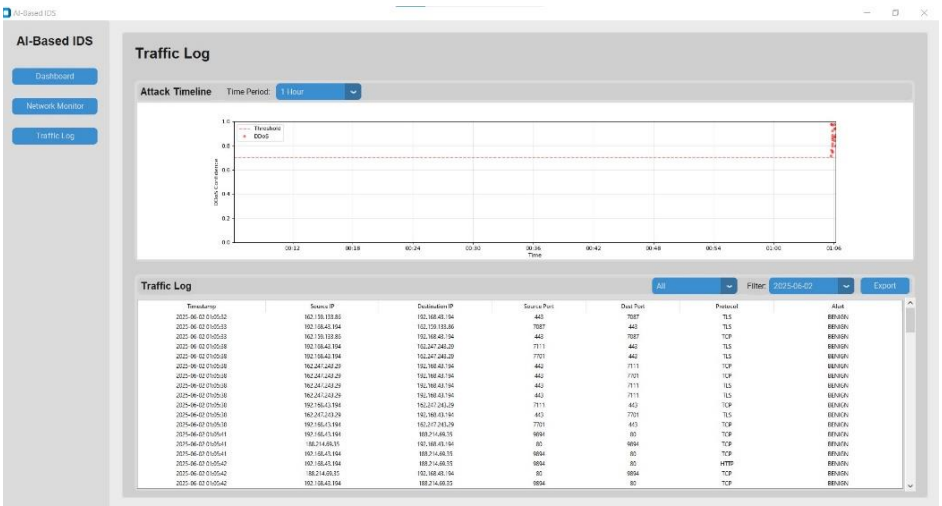

Fig. 13 *Traffic Log Page of AI-based IDS*

5.2 Model Training

The model uses a lightweight LSTM architecture optimized for real-time DDoS detection, based on 15 selected traffic features. The dataset is first loaded and cleaned to remove missing, infinite or duplicate values. Labels are mapped to binary values: 0 for Benign and 1 for DDoS. To address imbalance, RandomOverSampler is applied to balance the classes. Next, the data is normalized using StandardScaler, reshaped for LSTM input, and split into 60% training, 20% validation, and 20% test sets. The LSTM model includes two LSTM layers (64 and 32 units) with batch normalization and dropout, followed by a dense output layer with sigmoid activation.

Training is performed with binary cross-entropy loss, Adam optimizer, and metrics including accuracy, precision, recall, and AUC. EarlyStopping and ModelCheckpoint callbacks monitor validation AUC to avoid overfitting. After training, the best model is evaluated on the test set using standard evaluation metrics and the confusion matrix. The final trained model and scaler are saved for future real-time deployment in the AI-based IDS.

5.3 Evaluation Metrics Overview

The tool performance was measured using key evaluation metrics, including accuracy, precision, recall, and F1-score. These metrics are calculated based on the confusion matrix, which compares the predicted labels with the actual labels. The four basic components of the confusion matrix are True Positive (TP), Ture Negative (TN), False Positive (FP), and False Negative (FN). TP is correctly predicted positive cases (DDoS detected as DDoS). TN is correctly predicted negative cases (Benign detected as Benign). FP is incorrectly predicted positive cases (Benign detected as DDoS), and FN is incorrectly predicted negative cases (DDoS detected as Benign). The formula and their meanings are as follows:

Accuracy measures the overall correctness of the model as shown in Equation (1); it indicates how many predictions (both DDoS and Benign) the model got right.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision measures how many of the predicted positive (DDoS) cases were actually correct as shown in Equation (2); higher precision means fewer false alarms.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall measures how many actual positive cases were correctly identified, as shown in Equation (3); higher recall means fewer missed attacks.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

F1-Score is the harmonic mean of precision and recall, as shown in Equation (4); it balances precision and recall, and is useful when the dataset is imbalanced.

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4}$$

These metrics help quantify the model's ability to correctly identify both normal and anomalous network behaviors while minimizing false positives and false negatives.

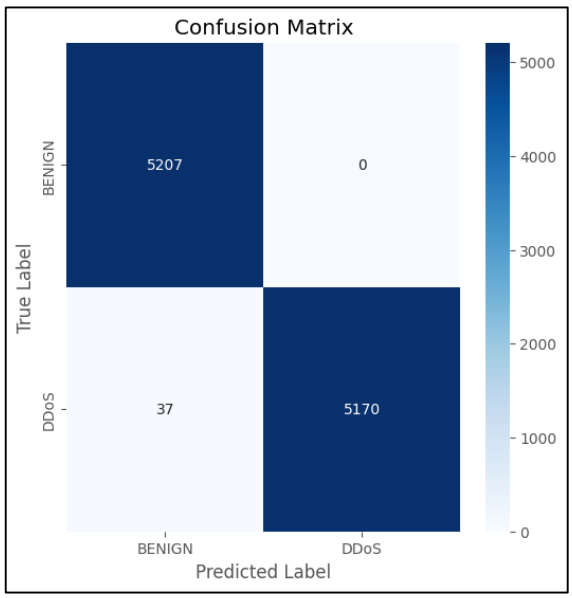

Fig. 14 *Confusion Matrix of the LSTM detection model*

After testing, the lightweight LSTM model was having an excellent performance with test loss of 0.0069, accuracy of 99.64%, precision of 100%, recall of 99.29% and AUC of 1.000. The classification report further confirms strong detection ability with both 'BENIGN' and 'DDoS' class achieving precision, recall, and F1-scores close to or at 1.00. The confusion matrix in Fig. 14 shows very few misclassifications, indicating that the model is highly effective at distinguishing between normal traffic and DDoS attack. Overall, these results suggest the LSTM IDS is well-suited for real-time DDoS detection with minimal false positives and high detection rates.

5.4 Testing and Verification

This section describes the testing and verification process of AI-based IDS based on its overall performance and user acceptance.

5.4.1 Testing Tool

The AI-based IDS was thoroughly tested following the prepared test plan shown in Table 7 which included checks for user interface functionality, real-time network traffic monitoring, anomaly detection, usability, performance, and compatibility across environments. Each test ensured that the tool worked as intended, aligned with project goals.

Table 7 *Application development workflow*

No	Test List	Description	Actual Result
1	User Interface Functionality Test	Verifies the usability and accessibility of the interface.	Pass
2	Real-Time Network Traffic Monitoring Test	Evaluate the tool's ability to capture and display live network data accurately	Pass
3	Anomaly Detection Test	Evaluates the effectiveness of the detection model in identifying suspicious activities.	Pass
4	Usability Test	Ensures the interface is intuitive and meets user expectations.	Pass
5	Performance Test	Measures the tool's responsiveness and efficiency under different workloads.	Pass
6	Compatibility Test	Checks whether the tool functions correctly in various environments or configurations.	Pass

In addition, the AI-based IDS is tested under two controlled scenarios to observe its performance in deployment. In the first scenario presented in Fig. 15, the tool monitored only normal traffic. It captured 257 packets and correctly identified all as benign, achieving 100% accuracy. In the second scenario displayed in Fig. 16, a DDoS attack was launched using the Low Orbit Ion Cannon (LOIC) tool, where the attacker machine (192.168.43.158) targeted the victim host (192.168.43.194). Out of 93 packets (34 actual DDoS and 59 benign), the AI-based IDS misclassified 5 benign packets as DDoS and 4 DDoS packets as benign, resulting in an overall accuracy of 90.32%. Despite a few misclassifications, the IDS demonstrated strong performance in both normal and attack conditions.

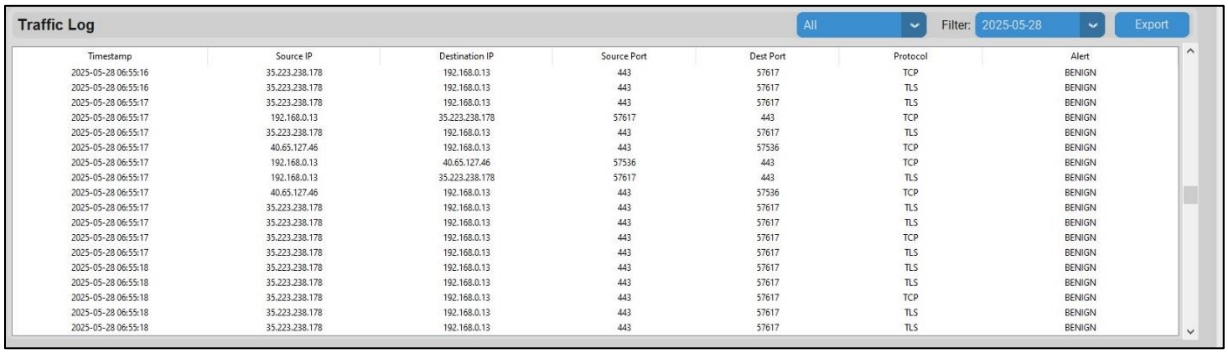

Fig. 15 *Testing Result (Normal Traffic)*

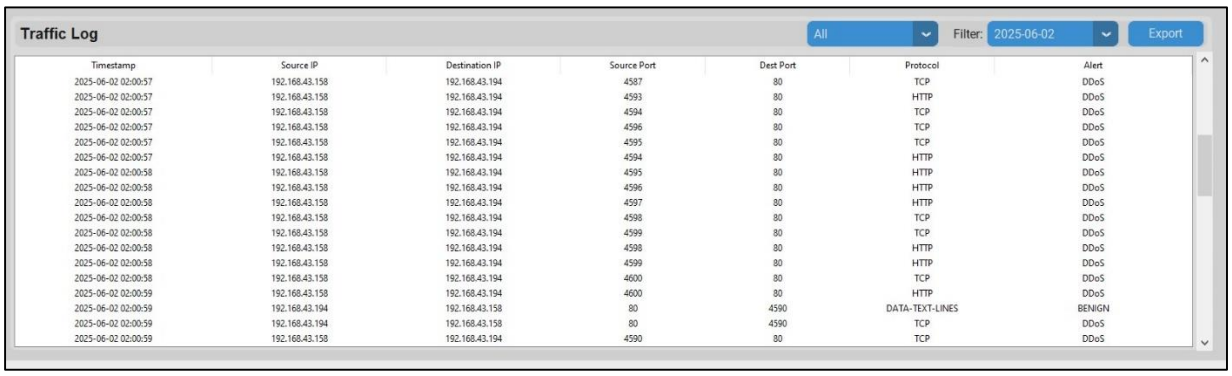

Fig. 16 *Testing Result (Under DDoS Attack)*

5.4.2 User Acceptance Testing

A user acceptance test was carried out where 15 users interacted with the tool and provided feedback through a Google Form, confirming that the tool met their expectations and was ready for deployment. The user acceptance testing form was shown in Table 8 Overall, the feedback from the 15 respondents was very positive, with most features receiving average scores above 4.8 out of 5. Users rated the interface, alert system, and traffic detection highly, with some questions scoring a perfect 5.0. The system's design, network and log pages, and overall effectiveness were also well-received. The only slightly lower score (4.53) was related to system responsiveness under high traffic, suggesting a potential area for further optimization. These results indicate the tool is user-friendly, effective, and suitable for deployment.

Table 8 *User Acceptance Testing (UAT) Form for AI-based IDS*

Question	Description	Average Score (1-5)
1	The tool interface is intuitive and easy to navigate.	5
2	All main functions (Start System, Alert Notification, Logs Export) work as expected and respond correctly.	4.93
3	The tool effectively display live network traffic in real time.	4.8
4	Clear and understandable notifications is delivered when suspicious activities are detected.	5
5	The tool accurately distinguishes between normal (benign) traffic and anomalous (DDoS) traffic.	4.87
6	The tool's performance and responsiveness during monitoring feel smooth even under high traffic loads.	4.53
7	The tool UI design looks nice and interesting.	4.8
8	The network monitor page provides useful and meaningful information about the historical/current network usage.	4.93
9	The traffic log page provide crucial and detailed information about the real-time traffic through attack timeline graph and traffic log table.	4.93
10	Overall, I am satistied with the tool's ability to enhance network security and provide reliable intrusion detection.	4.8

6. Conclusion

In conclusion, this project presents an AI-based IDS that combines LSTM networks and behavior analysis to effectively detect and mitigate advanced cyber threats in real time. Based on the testing conducted, the tool demonstrates strong detection accuracy, real time alerting capabilities, and easy-to-use UI., It ensures continuous protection for users of all backgrounds against evolving threats. Its adaptive design and strong compatibility will make it an excellent choice for enhancing network security.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *T. A. Goh, I. R. A. Hamid;* ***data collection:*** *T. A. Goh, I. R. A. Hamid,* ***analysis and interpretation of results*** *T. A. Goh, I. R. A. Hamid****; draft manuscript preparation:*** *T. A. Goh, I. R. A. Hamid. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] "Cyber Incident Quarterly Summary Report - Q2 2024," Sep. 2024. [Online]. Available: https://www.mycert.org.my/portal/advisory?id=SR-027.092024

[2] S. Ho, S. Al jufout, K. Dajani, and M. Mozumdar, "A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Network," IEEE Open Journal of the Computer Society, vol. PP, p. 1, Dec. 2021, doi: 10.1109/OJCS.2021.3050917.

[3] Y. Otoum and A. Nayak, "AS-IDS: Anomaly and Signature Based IDS for the Internet of Things," Journal of Network and Systems Management, vol. 29, no. 3, p. 23, 2021, doi: 10.1007/s10922-021-09589-6.

[4] P. Israelsson, J. Karlsson, and G. Giamarchi, "A quick overview of Snort," Oct. 2005. [Online]. Available: https://www2.it.uu.se/itwiki.php?page=edu/course/homepage/sakdat/ht05/assignments/pm/programme/Introduction_to_snort.pdf&action=browse

[5] E. Albin and N. C. Rowe, "A Realistic Experimental Comparison of the Suricata and Snort Intrusion-Detection Systems," in 2012 26th International Conference on Advanced Information Networking and Applications Workshops, 2012, pp. 122–127. doi: 10.1109/WAINA.2012.29.

[6] A. Waleed, A. F. Jamali, and A. Masood, "Which open-source IDS? Snort, Suricata or Zeek," Computer Networks, vol. 213, p. 109116, 2022, doi: https://doi.org/10.1016/j.comnet.2022.109116.

[7] M. Aamir, S. S. H. Rizvi, M. A. Hashmani, M. Zubair, and J. Ahmad, "Machine learning classification of port scanning and DDoS attacks: A comparative analysis," Mehran University Research Journal Of Engineering & Technology, vol. 40, no. 1, pp. 215–229, Jan. 2021, [Online]. Available: https://search.informit.org/doi/10.3316/informit.752436663509423

[8] D. Akgun, S. Hizal, and U. Cavusoglu, "A new DDoS attacks intrusion detection model based on deep learning for cybersecurity," *Comput Secur*, vol. 118, p. 102748, 2022, doi: https://doi.org/10.1016/j.cose.2022.102748.

[9] O. I. Falowo, M. Ozer, C. Li, and J. B. Abdo, "Evolving Malware and DDoS Attacks: Decadal Longitudinal Study," IEEE Access, vol. 12, pp. 39221–39237, 2024, doi: 10.1109/ACCESS.2024.3376682.

[10] Pathmahn and Nur Ziadah Harun, "Distributed Denial of Service (DDoS) Detection Tool Using Artificial Neural Network (ANN) for Homelab," *Applied Information Technology And Computer Science*, vol. 4, no. 2, pp. 139–158, [Online]. Available: https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/view/12074

[11] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," Transactions on Emerging Telecommunications Technologies, vol. 32, no. 1, p. e4150, Jan. 2021, doi: https://doi.org/10.1002/ett.4150.

[12] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," Neural Comput, vol. 9, pp. 1735–1780, Dec. 1997, doi: 10.1162/neco.1997.9.8.1735.

[13] C. Do Xuan and M. H. Dao, "A novel approach for APT attack detection based on combined deep learning model," Neural Comput Appl, vol. 33, no. 20, pp. 13251–13264, 2021, doi: 10.1007/s00521-021-05952-5.

[14] E. Zhao, N. Zhou, C. Liu, H. Su, Y. Liu, and J. Cong, "Time-aware MADDPG with LSTM for multi-agent obstacle avoidance: a comparative study," Complex & Intelligent Systems, vol. 10, no. 3, pp. 4141–4155, 2024, doi: 10.1007/s40747-024-01389-0.

[15] P. TS and P. Shrinivasacharya, "Evaluating neural networks using Bi-Directional LSTM for network IDS (intrusion detection systems) in cyber security," Global Transitions Proceedings, vol. 2, no. 2, pp. 448–454, 2021, doi: https://doi.org/10.1016/j.gltp.2021.08.017.

[16] H. N. Ooi and N. H. Ab Rahman, "A Comparative Study between Deep Learning Algorithm and Bayesian Network on Advanced Persistent Threat(APT) Attack Detection.," *Applied Information Technology And*

Computer Science, vol. 2, no. 2, pp. 219–235, [Online]. Available: https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/view/2324

[17] J. Datta, R. Dasgupta, S. Dasgupta, and K. R. Reddy, "Real-Time Threat Detection in UEBA using Unsupervised Learning Algorithms," in 2021 5th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech), 2021, pp. 1–6. doi: 10.1109/IEMENTech53263.2021.9614848.

[18] N. Clarke, S. Furnell, G. Tjhai, and M. Papadaki, Investigating the problem of IDS false alarms: An experimental study using Snort, vol. 278. 2008. doi: 10.1007/978-0-387-09699-5_17.

[19] S. A. R. Shah and B. Issac, "Performance comparison of intrusion detection systems and application of machine learning to Snort system," Future Generation Computer Systems, vol. 80, pp. 157–170, 2018, doi: https://doi.org/10.1016/j.future.2017.10.016.

[20] H. Alaidaros and M. Mahmuddin, "Flow-Based Approach on Bro Intrusion Detection," Journal of Telecommunication, Electronic and Computer Engineering, vol. 9, pp. 139–145, Jun. 2017.

[21] R. C. Martin, Agile Software Development, 2nd ed. Pearson Education, 2003. [Online]. Available: https://books.google.com.my/books/about/Agile_Software_Development.html?id=0HYhAQAAIAAJ&redir_esc=y

[22] O. Anurina, "Agile SDLC: Skyrocketing Your Project with Agile Principles," MLSDev. [Online]. Available: https://mlsdev.com/blog/agile-sdlc

[23] S. Racherla, P. Sripathi, N. Faruqui, M. A. Kabir, M. Whaiduzzaman, and S. A. Shah, "Deep-IDS: A Real-Time Intrusion Detector for IoT Nodes Using Deep Learning," *IEEE Access*, vol. 12, pp. 63584–63597, 2024, doi: 10.1109/ACCESS.2024.3396461.