SMART AI-BASED INTRUSION DETECTION SYSTEM

by

Sidra Gul

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Master of Science

(Cyber Security)

at the

NATIONAL UNIVERSITY OF SCIENCES & TECHNOLOGY, PAKISTAN

May 2023

Smart AI based Intrusion Detection System

By

Sidra Gul

00000326425

Supervisor

Prof. Dr. Arshad Aziz

Department of Cyber Security

A thesis submitted in partial fulfillment of the requirements for the

degree of Master of Science

In

Pakistan Navy Engineering College, National University of Sciences and Technology Karachi, Pakistan.

May 2023

Copyright Notice

With deep appreciation for the unwavering faith you have shown in me, I dedicate this thesis to my affectionate mother, supportive father, and beloved siblings...

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to are substantial extent has been accepted for the award of any degree or diploma at Department of Electrical Engineering at Pakistan Navy Engineering College (PNEC) or at any other educational institute, accept where due acknowledgement has he made in the thesis. Any contribution made to the research by others, with whom I have worked at Pakistan Navy Engineering College (PNEC) or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: **Sidra Gul**

Signature: Sidra

FORM TH-4

National University of Sciences and Technology

MASTER'S THESIS WORK

We hereby recommend that the dissertation prepared under our supervision by: (Student Name & Regn No.) Sidra Gul (00000326425)Titled: Smart AI based Intrusion Detection System be accepted in partial fulfillment of the requirements for the award of Master's degree.

EXAMINATION COMMITTEE MEMBERS

1. Name: Prof. Dr. Fawad Ahmed Signature: Fawad

2. Name: Dr. Bilal M. Khan Signature:

3. Name: ______________ Signature: ______________

Supervisor's name: Prof. Dr Arshad Aziz Signature:

Date: 04- Apr -2023

QURRATULAIN TI(M)
Lt Cdr Pakistan Navy
HOD C&SE

Head of Department

22-06-2023
Date

COUNTERSIGNED

Date: 22-06-2023

Dean / Principal
M IRFAN NADEEM
Captain Pakistan Navy
Deputy Commandant
PNS JAUHAR

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS thesis written by SIDRA GULRegn No. 00000326425 of NUST- PNEC (College) has been vetted by undersigned, found complete in all respects as per NUST Status/Regulations, is free of plagiarism, errors, and mistakes and is accepted as partial fulfillment for award of MS degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have been incorporated in the said thesis.

Signature: ____________________

Name of Supervisor Prof. Dr. Arshad Aziz

Dated: 22-June-2023

Signature: HoD____________________

QURRATULAIN TI(M)
Lt Cdr Pakistan Navy
HOD CySD

Dated: 22-06-2023

Signature: (Dean/Principal)____________________

M IRFAN NADEEM
Captain Pakistan Navy
Deputy Commandant
PNS JAUHAR

Dated: 22-06-2023

Approval

It is certified that the contents and form of the thesis entitled **"Smart AI Based Intrusion Detection System"** submitted by **Sidra Gul** have been found satisfactory for the requirement of the degree.

Advisor: **Prof. Dr. Arshad Aziz**

Signature: ______________________

Date: 22-06-2023

Committee Member 1: **Prof. Dr. Fawad Ahmed**

Signature: ______________________

Date: 22-06-2023

Committee Member 2: **Dr. Bilal M. Khan**

Signature: ______________________

Date: 22-06-2023

Acknowledgments

Above all, I express my gratitude to the Almighty for enabling me with the opportunity and capability to accomplish the thesis triumphantly.

I wish to extend my genuine thanks to Prof. Dr. Arshad Aziz, my supervisor, for the constant assistance in enabling me to undertake this study and research. He was a tremendous source of inspiration, encouragement, and patience for me as his advice was helpful to me throughout the entire research and writing process for this thesis.

Along with my advisor, I'd like to extend my appreciation to the other members of my thesis committee: Prof Dr. Fawad Ahmed and Dr. Bilal M. Khan, for their support, valuable comments, and challenging inquiries.

I am deeply grateful to every member of my family, especially to my dear parents for their compassion and kindness since the day I was born and strengthening me spiritually, emotionally and financially throughout my life.

I must end by sincerely thanking my dear friend, Sonia Azeem, for her unwavering support throughout this entire journey. Her encouragement and support have been priceless, and I cannot thank her enough for this.

Table of Contents

Page

List of Tables

Table Page

List of Figures

Figure Page

Figure Page

Abstract

As internet usage and connected devices continue to proliferate, network security is increasingly crucial to prevent cyberattacks and associated risks such as data breaches, identity theft, and financial harm. Intrusion detection systems (IDS) have become a vital component of network security, with two types available: signature-based and anomaly-based. IDS can detect and alert administrators of suspicious activity, enabling prompt responses to potential threats. In response to evolving cyber threats, using artificial intelligence (AI) to enhance intrusion detection systems (IDS) has been proposed. This paper presents the design of a Network Intrusion Detection System (NIDS) that uses deep learning algorithms to improve real-time intrusion detection rates by reducing false alarms and enhancing accuracy. The proposed NIDS employs customized dataset which includes updated CSE-CIC-2018 dataset also logs collected from real environments. This approach leads to an impressive accuracy rate of 96% with an Artificial Neural Network (ANN) algorithm in detecting network intrusions accuracy. Upon testing with real-world environment traffic, the model demonstrated a high level of accuracy, correctly classifying 92.1% of traffic flows, with a precision score of 91.3% and a recall score of 1.0. To further improve the model's accuracy, it is recommended to train it on an organization's specific traffic patterns and keep it updated with new attack patterns to maintain its efficacy. Overall, this system has the potential to assist organizations in strengthening their cybersecurity measures and better protect against potential attacks.

Chapter 1

Introduction

The Internet is currently a hot topic among researchers into information technology, software development, and cyber-security. network systems and its interconnected programs usage over the internet has grown significantly. It has now touched all spheres of society and become an important part of our daily lives. Currently, the internet is used for a variety of tasks connected to business, money transfers, interactions, and management of necessary tasks. The majority of enterprises utilise applications that enable a great number of users to access and extract significant amounts of information and data services in a huge range of domains. These applications are continually connected to the internet. As a result, secure media have been carefully used to complete the majority of these duties. The internet also touches on many important aspects of our daily lives, including social networking, education, technology, entertainment, and online services. Due to which the threats landscape for networks is evolving at an ever-faster rate.[1]

Factors including reliance on technology, sophisticated attacks and increased connectivity have contributed to the rise of cybersecurity which is derived from two words that are related to one another: cyber and security. Cyber refers to the corresponding technology, which includes all network hardware, including systems and programs, and security refers to the security methodology, which includes networks security, information security and system security. It is important to protect against intrusions or cyber attacks by using security measures such as firewalls, antivirus software, strong passwords and intrusion detection tools. One effective method for identifying and detecting cyberattacks and suspicious activities, which we will keep focus on in our work is an IDS (Intrusion Detection System) that keeps track of network traffic or system operations to detect any indication of unauthorized entry, malicious activity, or violations of security policies. IDS systems may be configured to monitor network traffic, system logs, or other sources of data, and they can be configured to take various actions when an intrusion is detected, such as alerting the administrator, or collecting evidence for forensic analysis and also taking appropriate action such as blocking the attack when employing Intrusion Prevention System (IPS) technology.[2]

Artificial intelligence (AI) has become a popular method for improving Intrusion Detection Systems (IDS) by using machine learning and deep learning algorithms to develop models that can learn and adapt to new attack patterns. Early work in this domain were centered on utilizing rule-based systems and expert systems, which relied on predefined rules and human expertise to identify intrusions.[3]

By utilizing AI-based IDS in our thesis, we aim to enhance the accuracy and efficacy of network attack detection while minimizing false positives rates. However, it is crucial to carefully consider algorithm selection, dataset choice (an up-to-date, balanced dataset), the feature selection process, the preprocessing steps, the hyperparameter tuning, evaluation metrics, and the selection of an appropriate open-source IDS for implementation while considering the computational resources required to train and deploy the model. Additionally, we need to validate the model's performance on a separate test set to ensure that it can generalize well to new, unseen data. Moreover, it is crucial to monitor and update the model continuously to ensure it adapts to new attack patterns and remains effective over time. This will involve regularly updating the training dataset, retraining the model, and evaluating its performance on new data.

1.1 Chapter Structure

This work is structured into chapters that serve different purposes:

Chapter 1 provides an introduction to the research problem and its significance.

Chapter 2 includes renowned cyber attacks types, highlights Intrusion detection system,its methodologies and classification, and gives an overview of Intrusion prevention system.

Chapter 3 of this work provides an overview of artificial intelligence (AI) in intrusion detection systems (IDS), as well as a literature review on the intersection of AI and IDS.

Chapter 4 outlines the research methodology which is an illustration of the proposed solution's design, including IDS deployment, how data was collected, preprocessed, and used to develop a model.

Chapter 5 provides an account of the outcomes of the conducted experiments and assesses the effectiveness of the intrusion detection system that relies on artificial intelligence.

Finally, ***Chapter 6*** summarizes this work and draws conclusions, including the limitations of the study and suggesting directions for future research.

Chapter 2

Network Security, Threats and Technologies

Network Security

Network security is a critical element of cybersecurity, which, in turn, is a subset of the broader field of information security as shown in Fig 2.1, that encompasses a set of practices, technologies, and policies designed to safeguard computer networks and the services they provide from unauthorized access, modification, or destruction. [4]

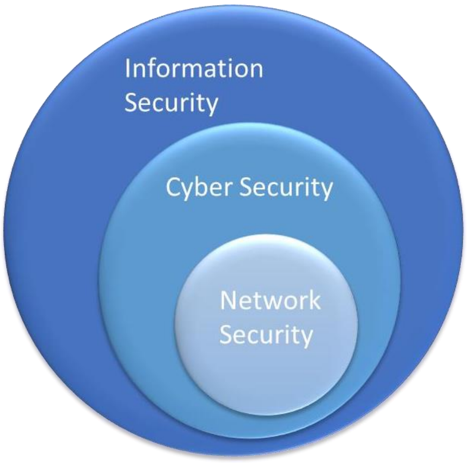


Figure 2.1 Network, Cyber and Information Security

Network security, cybersecurity, and information security are interrelated, with the CIA triad serving as a fundamental concept that underpins their effectiveness. The CIA triad evaluates how well organizations protect their information assets, helping them assess the efficacy of their security measures. [4] A diagram 2.2 demonstrating the connection between information security and the CIA triad is displayed below.

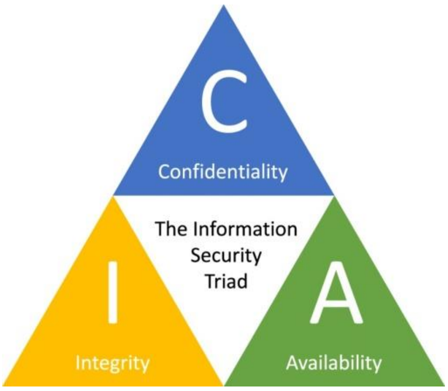


Figure 2.2 Information Security and CIA Triad

Confidentiality

Confidentiality is one of the core principles of the CIA triad, and it refers to the protection of sensitive information from unauthorized access or disclosure. In the context of network security, this involves implementing measures such as access control mechanisms, encryption, and data loss prevention solutions to ensure that only authorized individuals can access sensitive information. [5]

Integrity

Integrity is another key principle of the CIA triad, and it pertains to the preservation of the accuracy and completeness of information, as well as ensuring that it has not been tampered with. In the context of network security, this involves measures such as data backup and recovery, secure data transmission protocols, and data verification mechanisms to maintain the integrity of information. [5]

Availability

Availability is the third principle of the CIA triad, and it pertains to ensuring that authorized users have access to the information when they need it. In the context of network security, this involves measures such as implementing redundant network infrastructure, load balancing, and disaster recovery solutions to ensure that network services remain available even in the event of a network outage or failure.[5]

Network Security Tools

In today's digital age, networks are vulnerable to a wide range of threats, including viruses, malware, ransomware, phishing attacks, and data breaches. Effective network security is vital for organizations as a security breach can result in significant financial losses, legal liabilities, and reputational damage. Hence, it is essential to adopt robust network security measures to protect valuable data and systems.

To mitigate these risks, organizations must implement various measures to prevent and detect unauthorized access, misuse, or modification of information transmitted over the network. These measures can include implementing firewalls, intrusion detection and prevention systems, virtual private networks (VPNs), encryption, and access control mechanisms, among others.

Firewalls

Firewalls are an essential aspect of network security as they offer the first layer of defense against a variety of cyber threats. These security devices are designed to oversee and regulate traffic flow to and from a network by following predetermined security policies and regulations. By screening incoming traffic and preventing dangerous packets from entering, firewalls can prevent unauthorized access and attacks while reducing the likelihood of security incidents, such as data breaches. They act as a barrier between the internal network and the internet, halting potentially malicious traffic and impeding unauthorized access to the network. Firewalls can be implemented in various forms, including as hardware, software, or a combination of both, and can be tailored to implement different security policies depending on the organization's specific requirements. Popular types of firewalls include packet-filtering firewalls, stateful inspection firewalls, and application-level gateways. [6] The figure 2.3 displays traffic that has been identified as harmful and prevented from passing through the firewall.

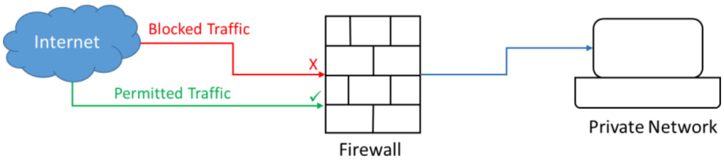


Figure 2.3 Firewall Working

Virtual Private Network (VPN)

A Virtual Private Network (VPN) is a crucial element for network security as shown in Fig 2.4, it creates a secure and encrypted connection between a user's device and a remote server, safeguarding sensitive data from potential attackers.

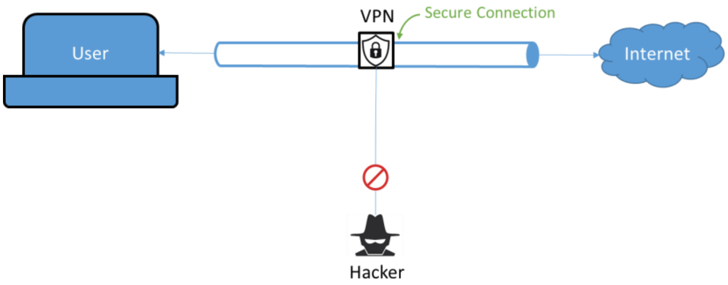


Figure 2.4 VPN

VPNs are available in various types, and each type offers its own benefits and drawbacks. The most common type is a remote access VPN which enables employees to securely access the company's network resources such as email, files, and applications, from any location with an internet connection, making it useful for remote workers and frequent travelers. [7]

Other types of VPNs include site-to-site VPNs that connect two or more networks securely over the internet and client-to-site VPNs that allow individual users to connect securely to a particular network. VPNs can also be classified based on the protocols used to establish and maintain the VPN connection, such as SSL VPNs, IPsec VPNs, and PPTP VPNs.

However, regardless of the type of VPN used, it is vital to select a reputable VPN service provider and implement appropriate security measures to ensure the protection of sensitive data and network resources. These security measures may include using strong encryption, implementing multi-factor authentication, and conducting regular security audits. [7]

Antivirus and anti-malware tools

Antivirus and anti-malware software are critical tools in maintaining network security, as malware infections can compromise the security and stability of the entire network, resulting in costly and damaging outcomes such as system failures and data breaches.

To prevent malware infections from spreading throughout the network, it is crucial to install and keep antivirus and anti-malware software updated on all devices connected to the network. Many of these programs offer network-wide management and monitoring tools, allowing administrators to centrally protect and manage multiple devices on the network.

It is also important to combine antivirus and anti-malware software with other network security measures, such as firewalls, intrusion detection systems, and regular security audits to identify and fix network vulnerabilities.

Furthermore, educating users about the risks of malware infections and best practices for network security is essential. This includes training on how to recognize and avoid common malware vectors, such as phishing emails and malicious websites, as well as instruction on how to use antivirus and anti-malware software effectively.

Intrusion Detection System

Intrusion detection systems (IDS) are dedicated tools that have the capability to automate the task of detecting and reacting to intrusions. [8] An **intrusion** can be characterized as any unauthorized activity or or an act of breaking into a computer system or network without permission which can result in potential damage to the information system, including jeopardizing the confidentiality, integrity, or availability of information. [9] This can be done for various reasons, such as to steal sensitive information, to disrupt the system, or to plant malicious software. Intrusions can be perpetrated by individuals or groups, and they can be carried out using various methods, such as exploiting vulnerabilities in the system, using brute force attacks, or tricking users into giving away their login credentials. The practice of observing computer systems or networks for any such indications of unlawful entry or harmful actions refers to **intrusion detection**. The process involves a range of approaches and tools to identify potential security breaches and notify security personnel of the need to take appropriate measures to avoid or lessen any harm.[10] These systems may be configured to monitor network traffic, system logs, or other sources of data, and they can be configured to take various actions when an intrusion is detected.

IDS are categorized based on two factors, which are their position in the network and methodologies as depicted in Fig 2.5. In terms of position, IDS can be classified into two types: network-based IDS (NIDS) and host-based IDS (HIDS). On the other hand, in terms of methodologies, IDS can be categorized into three types, namely signature-based IDS, anomaly-based IDS, and hybrid IDS.

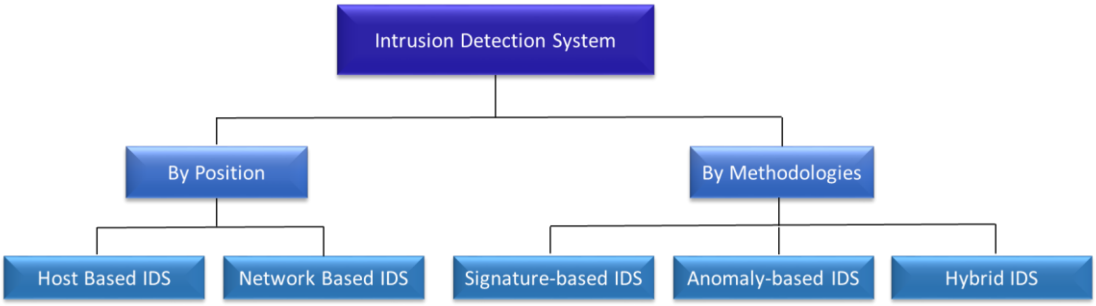


Figure 2.5 Intrusion Detection System Classification

Intrusion Detection Methodologies

A variety of methods can be employed for detecting intrusions, including:

Signature-based detection

Signature-based detection is a method used by some security tools to identify malicious activity or threats. As shown in Fig 2.6. It works by comparing the characteristics of a file or network traffic to a database of known ”signatures” of malicious software or attacks, whereas a signature is a pattern that identifies a certain threat or intrusion. Since prior information is necessary to develop such a database, this approach is also known as knowledge-based.[11] If a match is found, the system can alert the administrator or take other action to block the attack.

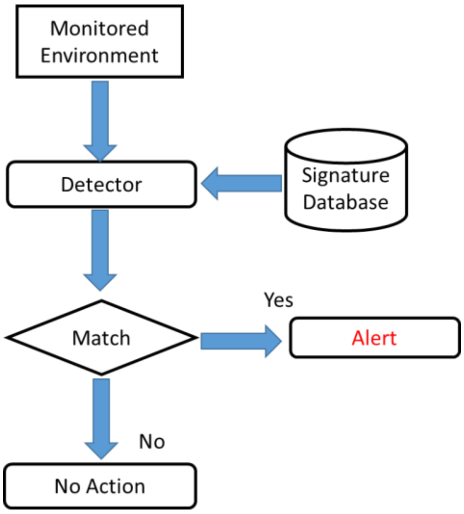


Figure 2.6 Signature based IDS Process

Signature-based detection can be an effective method of detecting known threats, but it has some limitations. One limitation is that it can only recognize threats that have been previously identified and have a specific signature stored in the database. If a new, unknown threat emerges, signature-based detection will not be able to identify it.

Another limitation is that signature-based detection can be resource-intensive, as it requires the constant updating of the signature database and the scanning of all files and traffic against this database.[12] This can lead to slower performance and increased overhead for the system.

Although there are limitations, signature-based detection is still commonly employed as a component of a comprehensive security plan, frequently in conjunction with other techniques like behavior-based detection and machine learning.

Anomaly-based detection

Anomaly-based detection is a security strategy that aims to recognize atypical or abnormal behavior or activity on a network or system. Unlike signature-based detection that uses particular patterns or signatures to identify hazards, anomaly-based detection establishes a baseline of standard behavior and then flagging any deviations from that baseline as potentially suspicious as depicted in Fig 2.7. This type of detection relies on a baseline or ”normal” behavior profile, which can be established through machine learning or other methods. The system continuously monitors the behavior of devices, networks, or systems, and compares this behavior to the normal behavior profile. When the system detects an anomaly or deviation from this normal behavior, it can trigger an alert or take other actions to respond to the potential threat.[13]

Different types of anomalies that an anomaly-based detection system might look for can include Unusual patterns of network traffic: such as an abrupt rise in traffic volume or an unusual pattern of traffic could indicate a cyber attack, Unfamiliar processes or executables: If the system detects the execution of a process or executable that it has not seen before, it could be an indication of an attack, and Unauthorized access or changes: If the system detects unauthorized access to a device, network, or system, or if it detects unauthorized changes to data or configurations, it could be an indication of an attack.

The effectiveness of anomaly-based detection in identifying new or unfamiliar threats lies in its independence from specific signatures or indicators of compromise. However, it may also generate a higher number of false positives because it is based on deviations from normal behavior, which can be caused by a variety of factors. To minimize false positives, it is important to carefully tune the normal behavior profile and to have processes in place for investigating and responding to alerts.

Hybrid Detection System

A hybrid detection system integrates signature and anomaly-based detection methods to improve threat detection and prevention. In order to eliminate known attacks using signature-based mechanisms, hybrid methods first look for a match before performing an anomaly detection check

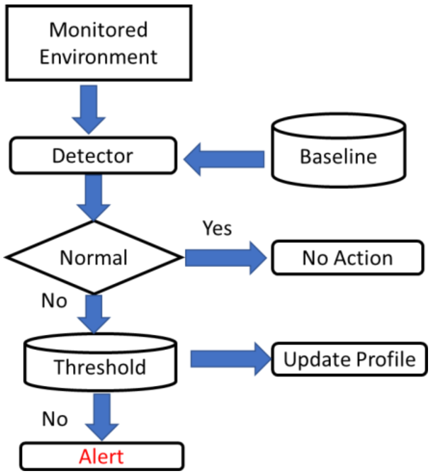


Figure 2.7 Anomaly based IDS Process

to detect any divergence from the typical behavior pattern. This increases the rate of detection and lowers the rate of false alarms. As cyber threats become more complex and sophisticated, the hybrid approach of integrating multiple detection methods is gaining popularity as it enables the system to more effectively recognize and counter a broader range of threats.[14]

Intrusion Detection System Categories

Intrusion Detection System is classified depending on it's placement inside the network where the it is being deployed. The two main categories of IDS are:

Host-based IDS (HIDS)

A HIDS, which stands for host-based intrusion detection system, is a type of security tool that runs on individual devices or systems and monitors events and activities on that device or system. The HIDS analyzes log files, system calls, and other activity on the host to detect patterns or anomalies that could indicate an attack. If the HIDS detects a potential threat, it can generate an alert or take other action to respond to the threat.

HIDS is capable of detecting various types of threats, such as malware infections, unauthorized access to system resources, and unauthorized modifications to system configurations. It can provide a comprehensive view of the activity on the host and can be customized to monitor particular processes or system files. However, HIDS may be less effective at detecting attacks that originate from external sources or that involve network-based attacks such as DoS attacks. A HIDS can be compared to an agent that checks to see if anything or anyone, internal or external, has managed to get through the security measures that the operating system tries to impose.[15]

Network-based IDS (NIDS)

A NIDS, which stands for network-based intrusion detection system is a type of security tool that examines a specific network segment to look for any suspicious activities or anomalies that could indicate an attack. The NIDS analyzes packets of data as they flow across the network and compares them to a set of rules,patterns or profiles that are known to be associated with attacks. If the NIDS detects a match to one of these patterns, it can generate an alert or take other action to respond to the potential threat.

NIDS can be deployed in a variety of locations within a network, including at network perimeter points such as firewalls, or at key locations within a network to analyze the traffic that flows through them. NIDS have the capability to identify various types of threats, including malware infections, network scanning, and Denial of Service attacks. However, NIDS can be susceptible to evasion techniques such as encryption or fragmentation of traffic, and may generate a high number of false positives if the rules or patterns used for detection are not carefully tuned.[16]

Intrusion Prevention System

An IPS is a security mechanism that aims to prevent attacks and minimize the impact of security threats on computer networks. It is commonly viewed as an extension of an intrusion detection system (IDS) and is also known as an Intrusion Detection and Prevention System (IDPS). The IPS system uses predefined rules or signatures to identify known attack patterns, including denial-of-service attacks, worms, Trojans, viruses, and other types of malware. It can also detect anomalies in network traffic and take action based on predefined security policies.

Once a potential threat is detected, the IPS can take several automated actions, such as blocking the offending traffic, sending an alert to security personnel, or reconfiguring firewall rules to prevent future attacks. Advanced IPS systems can also initiate incident response procedures like quarantining infected hosts or blocking access to specific network resources.

Compared to traditional IDS systems, IPS provides several advantages. Firstly, it can actively block malicious traffic, thus reducing the impact of attacks on the network. Secondly, IPS is more proactive in detecting and responding to security threats than IDS, which only alerts security personnel to the presence of a threat. Finally, IPS provides more granular control over network traffic, allowing administrators to implement fine-grained security policies and monitor network activity in real-time.

Despite its benefits, IPS technology is not without limitations. For instance, IPS can be resource-intensive, causing network performance issues. Moreover, IPS systems may generate false positives or false negatives, leading to legitimate network traffic being blocked or malicious traffic going undetected. As such, it is crucial to configure and manage IPS systems carefully to ensure their efficacy and prevent any inadvertent impact on network operations.[17]

Network Attacks

Network attacks are a type of malicious activity that targets vulnerabilities in a network's security protocols or components. These attacks can have a range of objectives, such as stealing valuable information, altering or destroying data, or simply disrupting normal network operations. Some common types of network attacks include:

Malware

Malware, a shortened term for malicious software, pertains to any software or code designed to inflict harm on computer networks, systems, or devices. Malware can manifest itself in different forms, including viruses, worms, trojan horses, spyware, ransomware, and adware. Its negative impacts include stealing sensitive information, disrupting system operations, corrupting files, and even assuming control of a device or network. Malware can be propagated through several means such as email attachments, infected software downloads, malicious websites, or through exploiting security weaknesses in outdated software or operating systems.

To ensure computer and network security, detection and removal of malware is of utmost importance. This can be achieved by utilizing security tools like antivirus and anti-malware software, firewalls, intrusion detection systems, and performing regular security assessments. Additionally, users must practice good security hygiene, like keeping software and systems up to date, avoiding suspicious emails and websites, and using strong passwords and multi-factor authentication.

Denial of service attack (DoS)

A DoS attack is a serious network security threat that can have significant consequences for businesses and organizations. The goal of a DoS attack is to disrupt or deny access to network resources by overwhelming them with traffic or requests. This type of attack can be carried out in various ways, such as UDP flood attacks, SYN flood attacks, HTTP flood attacks, and DNS amplification attacks.

In a UDP flood attack, the attacker sends a large number of User Datagram Protocol (UDP) packets to the target, which can cause network congestion and make the network unavailable to legitimate users. A SYN flood attack involves the attacker sending a large number of SYN requests to the target, which can cause the target to become overwhelmed and unresponsive. [18] In an HTTP flood attack, the attacker sends a large number of HTTP requests to the target, which can cause the target to become overloaded and unable to handle legitimate traffic. A DNS amplification attack involves the attacker using vulnerable DNS servers to generate large amounts of traffic that are directed towards the target. [19]

The effects of a successful DoS attack can be devastating for businesses and organizations. It can result in temporary unavailability of network resources, which can disrupt business operations and cause significant financial loss. In severe cases, it can lead to complete business failure. It is crucial to implement efficient strategies for detecting and avoiding DoS attacks. This may include deploying security measures such as firewalls, intrusion detection and prevention systems, and DoS mitigation services.

Distributed denial of service (DDoS) attacks

A DDoS attack is a type of cyber attack that floods the target with traffic or requests using multiple compromised systems. It is more complex and harder to defend against than a regular DoS attack, as it involves a larger number of systems and traffic sources that are located in different geographic areas.

The attacker typically creates a botnet by infecting computers with malware, which enables them to control the systems and launch synchronized attacks on the target. This makes it difficult for network administrators to distinguish between legitimate and malicious traffic.

DDoS attacks can take various forms, such as amplification attacks, reflection attacks, and application-layer attacks. Amplification attacks exploit vulnerabilities in servers to produce much larger responses to a small request, increasing the traffic directed at the target. Reflection attacks involve sending requests to a vulnerable server that appear to originate from the target, causing the server to flood the target with responses. Application-layer attacks target specific vulnerabilities in web applications or other software to overload the target.

Man-in-the-Middle (MitM) Attack

A Man-in-the-Middle (MitM) attack is a cyber attack whereby an attacker intercepts and modifies communications between two parties who believe they are communicating directly with each other. The attacker achieves this by intercepting and relaying the communication to the intended recipient while also monitoring and altering the transmitted information without either party being aware of the interference. MitM attacks can affect different communication channels such as email, messaging apps, voice calls, and public Wi-Fi networks. These attacks can result in serious consequences such as the theft of sensitive information such as passwords, credit card details, and personal data, and also enable the attacker to impersonate one of the parties and carry out unauthorized actions or transactions. [20]

Spoofing Attack

A type of network attack known as spoofing involves an attacker assuming the identity of a trusted entity, such as a website, email address, or IP address, with the aim of gaining unauthorized access to sensitive information or launching other malicious attacks. Various forms of spoofing attacks exist, including IP, DNS, ARP, and email spoofing.

IP Spoofing involves an attacker altering the source IP address of a packet to conceal their true identity or impersonate a trustworthy entity. DNS Spoofing, on the other hand, involves an attacker modifying the DNS records of a website to divert users to a fraudulent site that is intended to steal confidential information. ARP Spoofing involves an attacker transmitting fake ARP messages to associate their MAC address with the IP address of another device on the network, enabling them to intercept traffic and steal information. Finally, Email Spoofing involves an attacker falsifying the sender's email address to appear to originate from a reliable source, such as a bank or government agency, in order to deceive the recipient into disclosing sensitive information.

Preventing spoofing attacks necessitates the implementation of security measures like encryption, digital certificates, and two-factor authentication. Keeping software and security systems up-to-date is also critical in preventing known vulnerabilities from being exploited.[21]

Brute force attack

It is an attack method used by cyber attackers to exploit the weakness of weak passwords or encryption keys. Attackers utilize automated software or tools to systematically try every possible password or key until the correct one is found. This attack method can be targeted at various systems, including websites, databases, encrypted files, social media platforms, and email services.

To prevent such attacks, strong and unique passwords can be used, limiting login attempts, implementing multi-factor authentication, and using encryption to protect sensitive data in case of password compromise. These security measures can prevent attackers from gaining unauthorized access to systems or accounts. It is crucial to note that attackers are constantly evolving their attack methods, necessitating regular security updates and the implementation of new measures to prevent such attacks. [22]

Zero-day attack

A zero-day attack as shown in Fig 2.8 is a type of cyberattack that exploits a vulnerability in software, networks, or operating systems that has not been discovered or patched by the software developer. The term ”zero-day” refers to the fact that the vulnerability has not been known for any length of time before the attack occurs. These types of attacks can be difficult to defend against as the vulnerability may not be detected until after the attack has taken place. This makes them an attractive option for hackers and other malicious individuals who want to gain access to a system, steal data, or cause harm. To protect against zero-day attacks, organizations can implement various measures including keeping software and operating systems up-to-date with patches, using intrusion detection and prevention systems, monitoring emerging threats with advanced threat intelligence services, educating users on security best practices. [23]

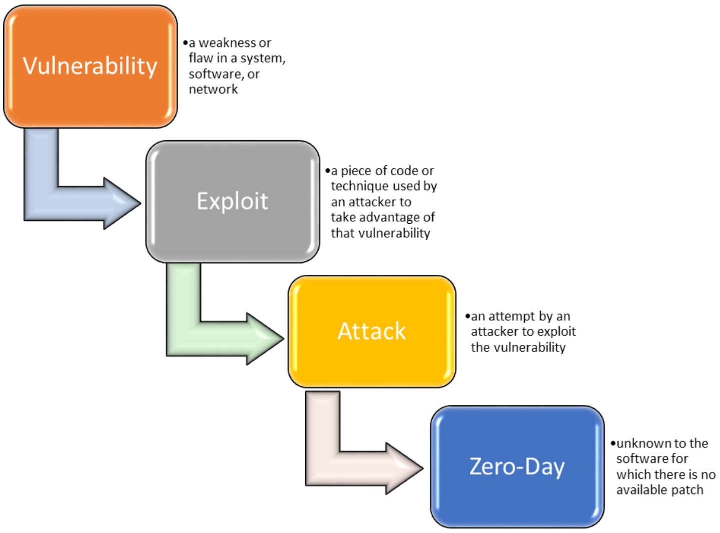

Vulnerability
•a weakness or flaw in a system, software, or network
Exploit
•a piece of code or technique used by an attacker to take advantage of that vulnerability
Attack
•an attempt by an attacker to exploit the vulnerability
Zero-Day
•unknown to the software for which there is no available patch

Figure 2.8 Zero-day Attack

After reviewing the various network security tools available, it becomes clear that An Intrusion Detection System (IDS), and specifically a Network-based IDS (NIDS), is an indispensable tool for effective network security. IDS provides comprehensive security by monitoring network traffic and system activity for potential security breaches or attacks. IDS solutions can identify both internal and external threats, respond to security incidents in real-time, and provide granular visibility into network activity. The use of an IDS also provides a rich research field with many techniques and methodologies available for exploration and study, offering practical applications for organizations seeking to improve their security posture. The benefits of NIDS are especially notable in its ability to monitor network traffic across multiple systems, enabling easier management and response to threats at scale. Overall, IDS is a promising area for thesis work, particularly when evaluated against a well-defined dataset that provides an objective means for assessing the effectiveness of the system.

Chapter 3

Role of AI in Cyber Security

Relation between AI, ML, DL

AI (Artificial Intelligence), ML (Machine Learning), and DL (Deep Learning) are interrelated but distinct concepts in the field of computer science and artificial intelligence a depicted in Fig 3.1. They represent different levels of sophistication in developing intelligent systems, and their integration can lead to significant advancements in various fields, including image and speech recognition, natural language processing, robotics, and automation. AI is a vast field that includes various techniques and algorithms for developing intelligent systems that can perform complex tasks. ML, is a subset of AI, involves the use of algorithms and statistical models to enable machines to learn from data without explicit programming. DL is a specific type of ML that uses artificial neural networks to learn and process data in a hierarchical manner.

AI role in Cyber Security

Artificial intelligence (AI) is playing a crucial role in the field of cybersecurity. By leveraging machine learning (ML) and deep learning (DL) algorithms, to detect security threats with greater speed and accuracy. By automating routine tasks, such as updating security policies and configurations and monitoring systems, AI frees up security personnel to focus on more complex issues. Additionally, AI improves threat intelligence capabilities by analyzing a variety of data sources to provide early warning of emerging threats. As the threat landscape continues to evolve, AI will play an increasingly critical role by providing early warning of emerging threats in protecting organizations and their data to stay ahead of the curve when it comes to new and evolving threats.

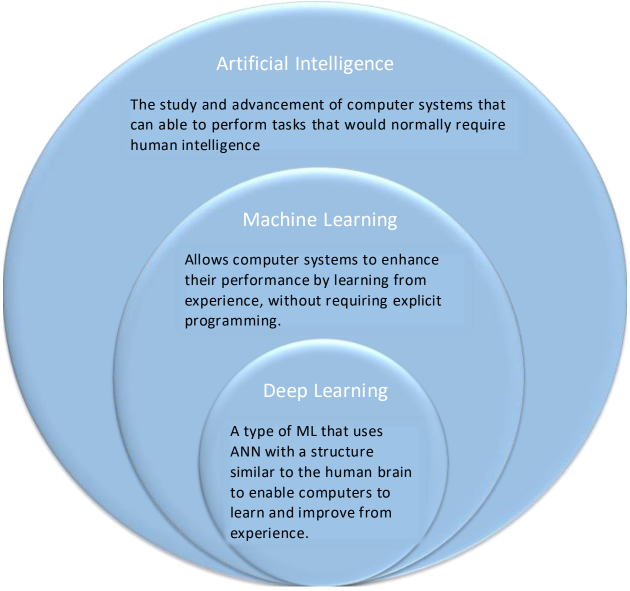

Artificial Intelligence
The study and advancement of computer systems that can able to perform tasks that would normally require human intelligence
Machine Learning
Allows computer systems to enhance their performance by learning from experience, without requiring explicit programming.
Deep Learning
A type of ML that uses ANN with a structure similar to the human brain to enable computers to learn and improve from experience.

Figure 3.1 How AI, ML, and DL are interlinked

AI importance in IDS

Intrusion Detection Systems (IDS) are crucial in recognizing and responding to security threats in networks. The role of AI in IDS is vital, as it employs machine learning algorithms to evaluate network traffic and recognize potential threats, including malware infections, network scans, and denial-of-service attacks. AI-based IDS can learn normal patterns of activity and detect irregularities that may indicate suspicious behavior. This makes them able to recognize previously unknown or zero-day attacks that conventional signature-based IDS may not detect. AI-based IDS can adapt to new threats and upgrade their detection abilities over time, providing better protection against evolving threats. Additionally, AI can help decrease false positives and enhance alert accuracy, enabling security teams to concentrate on the most critical threats.

Machine Learning

Machine learning (ML) is a subfield of artificial intelligence (AI) that is concerned with the development of algorithms and statistical models that enable computer systems to automatically improve their performance on a given task through experience. ML achieves this goal through the training of algorithms on datasets, thereby enabling them to recognize and extract patterns and relationships from the data. The resulting models can then use this knowledge to make predictions or decisions with little to no human involvement. This fundamental principle of ML holds that computers are capable of acquiring knowledge from data, which they can apply to new, unseen data in a generalized fashion.

ML algorithms can be broadly categorized into several groups which serve as a starting point for selecting the appropriate algorithm for a given task and allow practitioners to tailor their approach to specific use cases. As displays in Fig 3.2 this categorization includes supervised learning, unsupervised learning, and reinforcement learning, depending on the nature of the training data and the learning objectives.

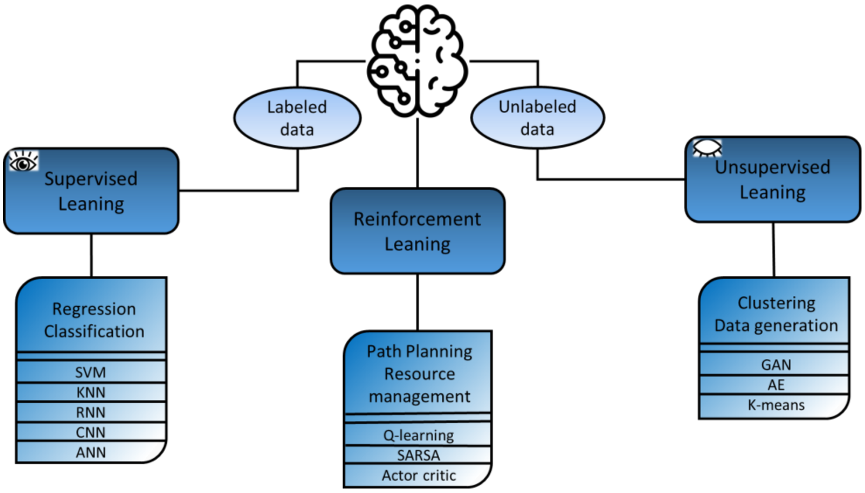


Figure 3.2 Machine Learning Categorization

Supervised Learning

Supervised learning is a core machine learning algorithm that trains a model to make predictions or decisions based on labeled data. The labeled dataset includes input features and output labels, and the algorithm learns the relationship between them during training. [24]

The primary objective of supervised learning is to predict output labels accurately for new, unseen input data. Supervised learning problems are divided into two main categories: regression and classification.

Regression is used to predict a numeric value when the output variable is continuous. Regression algorithms include Linear regression, Polynomial regression, Support Vector Regression (SVR), Decision tree regression, Random forest regression, Gradient Boosting regression, and Neural networks (e.g., Multi-Layer Perceptron). [24]

In contrast, **Classification** is used to predict the class or category of input data when the output variable is categorical. Classification algorithms consist of Logistic regression, K-Nearest Neighbors (KNN), Decision tree classification, Random forest classification, Support Vector Machine (SVM), Naive Bayes classification, and Neural networks (e.g., Convolutional Neural Networks for image classification, Recurrent Neural Networks for text classification). [24]

Unsupervised Learning

Unsupervised learning involves machine learning algorithms that aim to uncover patterns or relationships in unlabeled datasets. Unlike supervised learning, unsupervised learning does not involve a target variable to be predicted. Instead, the goal is to learn the underlying structure of the data and discover meaningful insights. Common unsupervised learning algorithms include clustering, dimensionality reduction, and association rule mining. Unsupervised learning is widely used in various fields such as anomaly detection, market segmentation, and recommendation systems. [24]

Reinforcement Learning

The reinforcement learning algorithm learns through rewards or penalties received during interaction with the environment, with the aim of maximizing cumulative rewards over time. It is useful in scenarios where labeled data is unavailable, and has applications in robotics and game playing. There are model-based and model-free reinforcement learning algorithms, including Q-learning, SARSA, and actor-critic methods. Despite its remarkable performance in challenging tasks, reinforcement learning still faces challenges, such as sample efficiency, scalability, and generalization to new environments. [24]

Neural Networks or ANN

Neural network or artificial neural network (ANN) is a type of computational model that is created to imitate the structure and operation of the human brain. It consists of interconnected processing units called neurons that are arranged in layers. The input layer receives data and passes it to the hidden layers where the data is processed using weights and biases before being transferred to the output layer. The weights and biases are adjusted through a process called backpropagation, which helps the network learn from its errors and enhance its performance over time.[25] The diagram in Figure 3.3 illustrates the typical architecture of ANN.

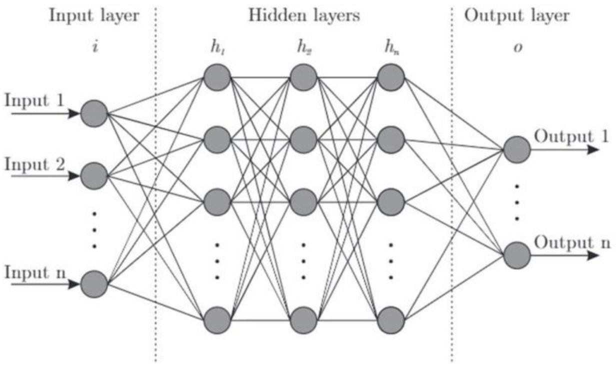


Figure 3.3 General architecture of ANN

There are several types of ANNs, each with its own structure and learning algorithm. Some common types of ANNs include feedforward neural networks, recurrent neural networks, and convolutional neural networks.

Feedforward neural network

Feedforward neural networks (FFNNs) are a type of artificial neural network (ANN) characterized by their layered architecture. They contain multiple layers of interconnected nodes where each node receives input from the preceding layer and passes its output to the following layer, creating a unidirectional flow of information. This property makes FFNNs suitable for supervised learning tasks, where the objective is to map input data to output data.

Each node in an FFNN is associated with a weight that determines the strength of the connection with the nodes in the next layer. During training, the weights are adjusted so that the network can learn to recognize patterns in the input data and generate the desired output.

Multi-Layer Perceptrons

Multi-Layer Perceptrons (MLPs) are a specific type of FFNN that are widely used for supervised learning tasks, such as classification and regression. MLPs contain at least one hidden layer of nodes between the input and output layers, and they use backpropagation algorithms to adjust the weights of the connections between the nodes during training. MLPs are capable of capturing nonlinear relationships between the input and output data. [26]

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) process sequential data, like time-series or text, using feedback connections to maintain a memory of previous inputs. RNNs consist of interconnected neurons in layers that receive input from the previous layer and feedback from the same layer in the previous time step. The Long Short-Term Memory (LSTM) network is a widely used type of RNN that addresses the ”vanishing gradient” problem by selectively forgetting or remembering previous inputs based on their relevance to the current task. LSTM networks are especially useful for handling long sequences of data and are applied in natural language processing, speech recognition, and other sequential data applications. [24]

Convolutional Neural Networks

CNNs are highly effective in image recognition and computer vision tasks. They identify and learn patterns in images through convolutional layers that capture spatial relationships between neighboring pixels. The initial layers extract low-level features while deeper layers extract more complex features, resulting in a higher-level representation of the input image. Fully connected layers perform classification or regression tasks and produce the final output. CNNs have achieved remarkable results in image classification, object detection, and facial recognition and are an active area of research in computer vision. [24]

Chapter 4

Literature Review, Problem Statement and Objective

Literature Review

The initial introduction of intrusion detection systems dates back to 1980 when James P. Anderson wrote a report titled ”Computer Security Threat Monitoring and Surveillance.” [27]

The first IDSs were simple systems that used basic rule-based techniques to identify potential security breaches. These systems relied on manually defined rules that specified the conditions under which an alert should be triggered. For example, an IDS might be configured to alert the system administrator if a user attempted to log in with an incorrect password more than three times in a row. State Transition Analysis (STA) is a rule-based intrusion detection method developed by Ilgun, Kemmerer, and Porras [28] that examines the sequence of states and transitions that occur during computer system operation to detect anomalies. The method employs predefined rules to model the normal behavior of the system, and any deviations from this model are flagged as potential intrusions. STA constructs a finite-state machine (FSM) to model expected behavior and compares it to the observed behavior of the system, making it a versatile intrusion detection approach that does not rely on signatures or patterns of known attacks. STA is less prone to false positives, and more resistant to evasion techniques. Despite its advantages, STA has certain limitations. The method can be computationally demanding as it must monitor and analyze the complete sequence of system calls made by processes.it may not be effective against sophisticated attacks that mimic normal system behavior or evade detection, and requires significant expertise to create and implement the rules to model anticipated system behavior, posing a challenge for some organizations.

As computer networks became more complex and the number of potential security threats increased, rule-based IDSs became less effective. The introduction of machine learning techniques such as decision trees, neural networks, and support vector machines in the late 1990s and early 2000s opened up new possibilities for IDSs. Machine learning algorithms were able to learn from large datasets of network traffic and identify patterns that may not have been apparent to humans. As a result, IDSs that used machine learning techniques were able to detect previously unknown threats and improve their accuracy in identifying known threats. In the mid-2000s, researchers began to explore the use of unsupervised machine learning techniques such as clustering and anomaly detection for intrusion detection. These techniques did not require labeled data and were able to identify unusual patterns in network traffic, making them well-suited to identifying unknown threats. More recently, deep learning techniques such as convolutional neural networks and recurrent neural networks have been applied to intrusion detection. These techniques are capable of learning complex patterns in network traffic and can detect subtle changes that may be missed by traditional rule-based systems or even traditional machine learning models.

Wen Xu et al. [29] present a new method to enhance the efficiency of autoencoder-based network anomaly detection on the NSL-KDD dataset through a 5-layer architecture model. Initially, the authors preprocess the dataset by eliminating duplicate features, transforming categorical features to numerical ones, and scaling the data. Afterward, an autoencoder is trained on the preprocessed dataset to identify network anomalies. To enhance the autoencoder's performance, the authors propose a new approach called ”residual correction.” In this technique, the autoencoder is first trained on the entire dataset, and the reconstruction error is utilized to detect potential anomalies. These possible anomalies are then removed from the dataset and the autoencoder is retrained on the remaining data. Finally, the removed anomalies are included back into the reconstructed data, and the reconstruction error is computed again. The authors demonstrate that this technique enhances the accuracy of the autoencoder in detecting anomalies. The model exhibits potential in identifying unusual network traffic patterns, however, its efficacy in real-world operational network environments needs further evaluation and research.

Xianwei Gao et al. [30] presents an adaptive ensemble learning model for the detection of intrusions, utilizing various machine learning algorithms such as DT, SVM, LR, kNN, Adaboost, RF, and deep neural networks. The model incorporates the selection of five voting classifiers and applies techniques such as sample proportion adjustment, data weighting, and multi-layer detection to enhance algorithm effectiveness. By utilizing an adaptive voting algorithm with different class-weights, the model aims to achieve optimal detection outcomes. However, the paper has limitations, including a lack of detailed information on the feature selection algorithm, limited comparative analysis, unaddressed scalability and real-time adaptability concerns, and a lack of insights into practical deployment and integration. Further research is needed to address these limitations and validate the model's applicability and effectiveness.

Yongkuan Zhu et al. [31] investigate the potential of data mining methods, particularly the FP-growth and Apriori algorithms, to enhance security sustenance and identify network intrusion. The authors highlight the importance of detecting network intrusion in real-time and the challenges associated with doing so. According to them, data mining methods can be utilized to examine data on network traffic and recognize patterns that may suggest a security threat or intrusion. The FP-growth and Apriori algorithms are used to mine frequent patterns in the data and generate rules for identifying potential intrusions. To evaluate the effectiveness of these techniques, the authors used the KDDCUP99 dataset, which contains a set of network traffic data that has been labeled as either normal or malicious. They applied the FP-growth and Apriori algorithms to this dataset and compared their performance with other intrusion detection methods. Data mining techniques, such as the FP-growth and Apriori algorithms, can become computationally expensive when applied to large datasets. As networks become more complex and generate larger volumes of data, there may be limitations in the ability of these techniques to scale and handle the increased data load.

Before applying machine learning algorithms to data, it is necessary to process the data by selecting relevant features and eliminating redundant and noisy features. Chaouki Khammassi et al. [32] have proposed a novel feature selection approach for network intrusion detection, which uses a hybrid of NSGA2 (Non-Dominated Sorting Genetic Algorithm) and logistic regression (LR) to minimize the number of features and maximize classification accuracy. This wrapper approach has been applied to three datasets: NSL-KDD, UNSW-NB15, and CIC-IDS2017, and three classifiers have been used for each dataset: Decision Tree (DT), Random Forest (RF), and Naive Bayes Tree (NBTree). The results demonstrate that binary-class datasets achieve higher accuracy than multi-class datasets, but some attacks, such as U2R in NSL-KDD, Backdoor, Analysis, Exploits, and DoS in UNSW-NB15, and Web-attack-XSS in CIC-IDS2017, are difficult to identify. However, the proposed method may result in overfitting, where the selected features perform well on the training data but poorly on unseen data.

Problem Statement

There has been a lot of research done on intrusion detection systems and Artificial Intelligence, but still a question arises about a comprehensive and reliable dataset that includes both recent and current attacks for utilizing them in intrusion detection systems. Previous studies has employed techniques for selecting features by reducing their number which assume that the input features are independent of each other. However, in practice, some features may be highly correlated, leading to redundant information. In such cases, feature selection techniques that take into account the interdependence between features, such as correlation-based feature selection, could be more effective. One of the significant challenges is the issue of false alarm rates that comprises of false positives and false negatives. These rates must be reduced for better detection of traffic. Apart from the above issues, previous studies that used deep learning for intrusion detection have faced challenges in applying their models in real-world settings. This is mainly because these models were often pre-processed into metadata formats within controlled experimental environments. As a result, only a limited number of studies have demonstrated how to apply these models in real-time situations outside the lab.

Objective

Keeping all of the previous statements in consideration, our goal is to design an advanced deep learning-based NIDS (Network Intrusion Detection System) using open-source tools. This system aims to improve detection rates and minimize false alarms in real-time. To ensure accuracy, the model will be tested using a balanced dataset. By employing this system, organizations can enhance their cybersecurity measures and protect against potential attacks.

Chapter 5

Methodology

The purpose of this chapter is to present a detailed and comprehensive account of each of the various phases involved in the workflow employed during the experiments conducted for this project.

Proposed Methodology

The experiments consist of two primary steps:

1. Setting up an authentic environment that produces logs, which are then detected by a functional IDS (Suricata), and subsequently evaluated by the trained model.

2. Training a deep learning model to categorize network traffic as safe or harmful.

The proposed methodology is illustrated in the accompanying Fig 5.1

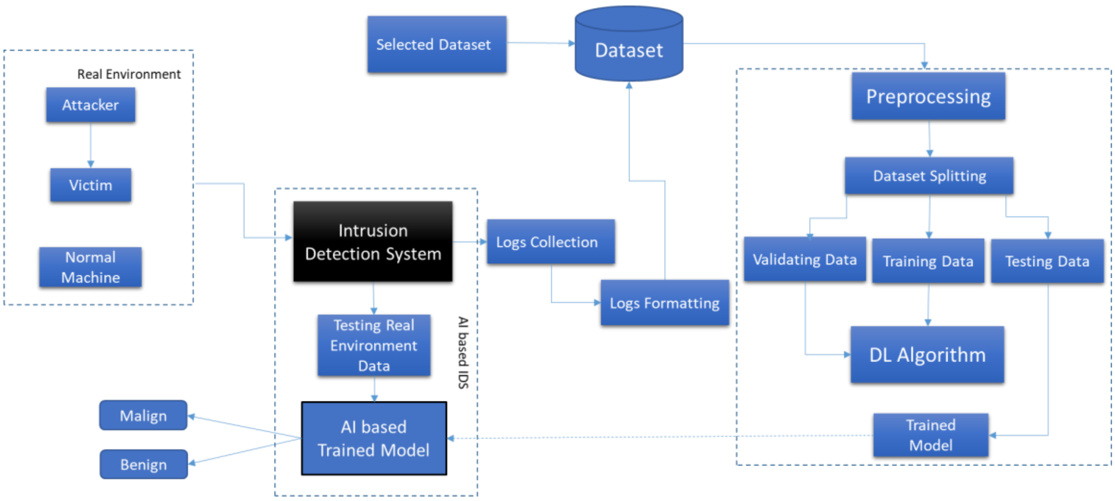


Figure 5.1 Proposed Methodology

By following these steps, an open-source tool can be made intelligent via AI in detecting intrusions. This approach can help enhance the tool's detection capabilities, making it more effective in identifying potential threats and minimizing false positives and false negatives

Preprocessing

This step involves a series of critical tasks, beginning with data collection. The process of collecting data includes the careful selection of an appropriate dataset and the preparation of the data for analysis. Additionally, attention must be given to the distribution of classes within the dataset and to the selection of relevant features. Finally, the dataset must be split appropriately to ensure that the resulting models are properly trained and validated. A graphical representation of these tasks is illustrated in the accompanying Figure 5.2.

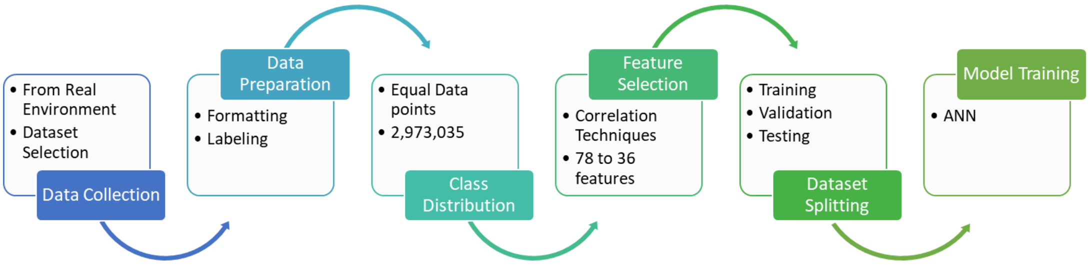


Figure 5.2 Preprocessing

Data Collection and Preparation

The initial stage involves gathering and preparing the data which will be utilized for the deep learning model's training. This involves gathering log data and network traffic data from various sources, cleaning and preprocessing the data to ensure it is consistent and relevant to the task.

Real Environment Creation

The process of creating a real environment requires the establishment of an authentic operating system, network, and software infrastructure that accurately emulates real-world conditions. To this end, we have developed a real environment consisting of multiple machines designed to function as an attacker, victim, and normal machine. Additionally, a network intrusion detection system (NIDS) in the form of Suricata has been installed to provide real-time traffic analysis, alerting, and logging of network activity.

In our study, we conducted a series of experiments to investigate the susceptibility of a victim machine running an Apache server to attacks. To this end, we employed various methods, including the Hulk tool and exploiting vulnerabilities through the Metasploit framework, using a Kali machine as the attacker. Subsequently, we launched denial-of-service (DoS) attacks through the Metasploit framework, which were found to be undetectable by intrusion detection systems (IDS). To further analyze and train machine learning algorithms for intrusion detection, we included the logs from these attacks as part of our dataset. This methodology is widely used in the field of machine learning for intrusion detection as it allows researchers to evaluate the effectiveness of their algorithms on real-world attack data.

As part of our research, we intend to utilize a distinct machine to conduct standard activities, such as web browsing, email communication, and file transfers. The resulting logs generated from these activities will be employed to assess the efficiency of the trained model.

This evaluation process is a critical aspect in determining the model's capability to differentiate between normal activities and potential attacks accurately. By utilizing a thorough dataset that comprises both attack logs and normal activity logs, we can ensure that the trained model can detect and categorize attacks while minimizing the incidence of false positives, which is a significant hurdle in intrusion detection.

Our goal through this evaluation process is to develop an intrusion detection system that is highly accurate and effective, capable of adapting to various attack methodologies, and offering resilient protection to network systems. 5.3

Suricata

Suricata is an IDS that is open-source and freely available, created to be fast and effective in processing vast quantities of network traffic. Key features of Suricata is illustraded in Figure

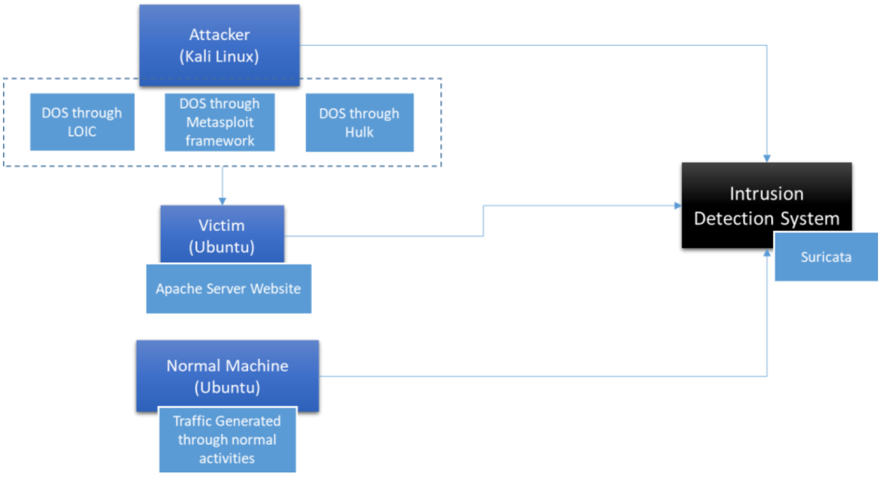


Figure 5.3 Real Environment

5.4. The IDS has the ability to perform instantaneous inspection of network actvities, generate alerts, and maintain logs of network activities. Moreover, it possesses the capability to identify different types of security threats, including malware, attempts of intrusion, and abnormalities in the network.

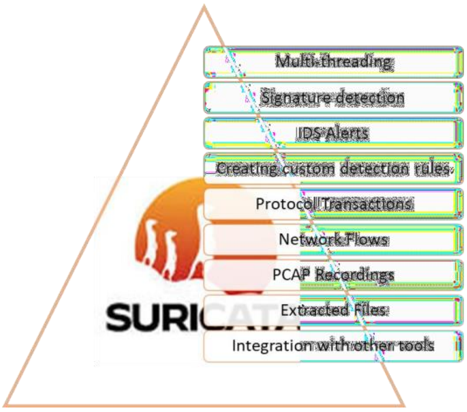


Figure 5.4 Suricata Key Features

Suricata is a multi-threaded, which means that it can take advantage of multiple CPU cores to improve its performance that allows Suricata to process multiple packets and events simultaneously, reducing the time required to process large amounts of network traffic. Suricata is also a rule-based IDS that uses the Emerging Threats and VRT rule sets to detect known and unknown threats. It also supports a wide range of protocols, including HTTP, SMTP, and SSH.[33]

Additionally, Suricata has a feature called ’multi-processing’ which allows to run multiple instances of Suricata on the same host, this way it can scale the performance by using multiple CPU cores. This feature allows to increase the packet processing capacity of Suricata. [34]

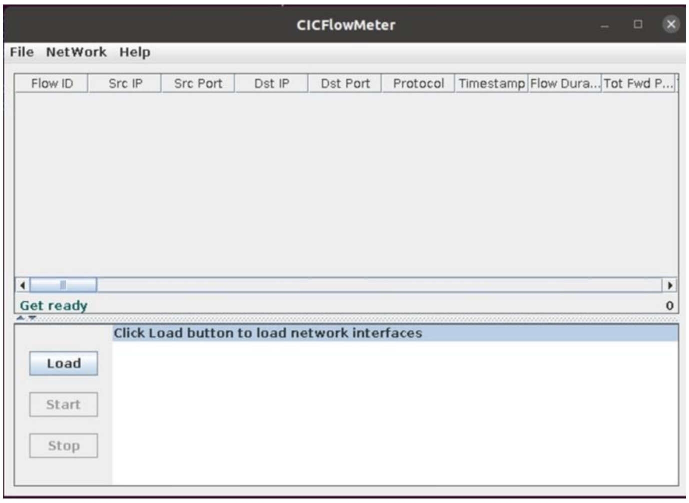
After Suricata was installed, we added rulset providers and get them updated. For the network traffic detection in the pcap format, we have enabled pcap capturing in the suricata.yaml file.

Selection of Dataset

After capturing the network traffic in ”pcap” format, we utilized the CICFlowmeter tool to extract relevant network data, which were then saved in multiple CSV files. To prepare the dataset for our experiments, we labeled the network traffic in these CSV files as either ”Benign” or ”Malign”, based on their characteristics. We also included the dataset from CSE-CIC-IDS2018 which were downloaded from Kaggle website. This dataset contains various types of attacks. To simplify the analysis, we replaced the attack labels in the CSE-CIC-IDS2018 dataset with the label ”Malign” to create a binary classification system. After removing the ’Flow ID’, ’Src IP’, ’Src Port’, and ’Dst IP’ features from the captured logs, our resulting dataset contains a total of 79 features. Of these, 78 features have a data type of ”int64” or ”float64”, and one feature has a data type of ”object”. The data types of our features now closely resemble those found in the CSE-CIC-IDS2018 dataset.

CIC FlowMeter

CIC FlowMeter as depicted in Figure 5.5 is a proficient and dependable tool for analyzing network traffic data, which offers a user-friendly interface. This open-source software is equipped with the ability to convert packet capture (pcap) files into CSV format, as well as extract almost 80 distinct features from the network traffic data. These features include aspects such as packet size, duration, protocol, source, and destination IP addresses, among others. By utilizing its extensive features, CIC FlowMeter enables a detailed examination of network traffic data and proves to be an essential resource for network administrators, cybersecurity experts, and researchers alike.



Get ready

Click Load button to load network interfaces

Figure 5.5 CICFlowmeter

The final combined dataset is illustrated in the provided Figure 5.6

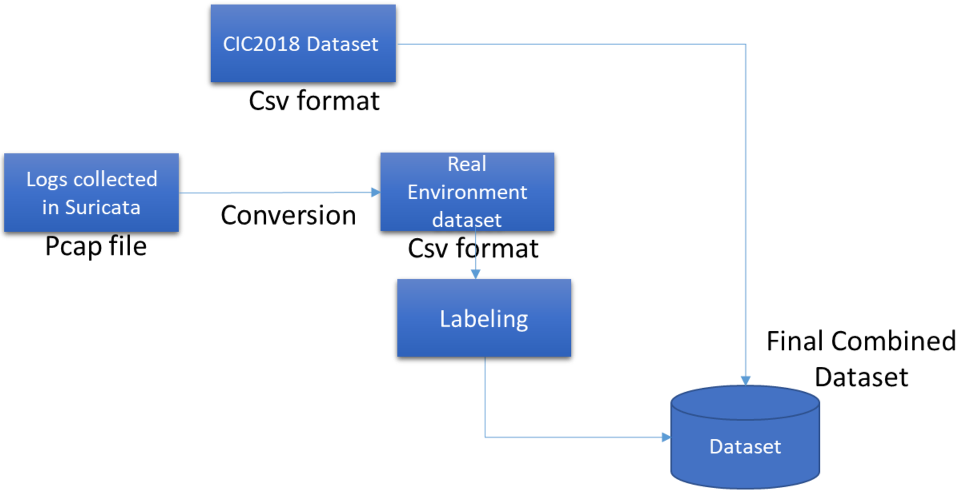


Figure 5.6 Combined Dataset

Table 5.1 presents a comprehensive list of all 79 features in our dataset, along with their corresponding data types.

Table 5.1: Total number of features in our combined Dataset

No	Features	Dtype	No	Features	Dtype
0	Dst Port	Int64	40	Pkt Len Max	Int64
1	Protocol	Int64	41	Pkt Len Mean	Float64
2	Flow Duration	Float64	42	Pkt Len Std	Float64
3	Tot Fwd Pkts	Int64	43	Pkt Len Var	Float64
4	Tot Bwd Pkts	Int64	44	FIN Flag Cnt	Int64
5	TotLen Fwd Pkts	Int64	45	SYN Flag Cnt	Int64
6	TotLen Bwd Pkts	Int64	46	RST Flag Cnt	Int64
7	Fwd Pkt Len Max	Int64	47	PSH Flag Cnt	Int64
8	Fwd Pkt Len Min	Int64	48	ACK Flag Cnt	Int64
9	Fwd Pkt Len Mean	Float64	49	URG Flag Cnt	Int64
10	Fwd Pkt Len Std	Float64	50	CWE Flag Count	Int64
11	Bwd Pkt Len Max	Int64	51	ECE Flag Cnt	Int64
12	Bwd Pkt Len Min	Int64	52	Down/Up Ratio	Int64
13	Bwd Pkt Len Mean	Float64	53	Pkt Size Avg	Float64
14	Bwd Pkt Len Std	Float64	54	Fwd Seg Size Avg	Float64
15	Flow Byts/s	Float64	55	Bwd Seg Size Avg	Float64
16	Flow Pkts/s	Float64	56	Fwd Byts/b Avg	Int64
17	Flow IAT Mean	Float64	57	Fwd Pkts/b Avg	Int64
18	Flow IAT Std	Float64	58	Fwd Blk Rate Avg	Int64
19	Flow IAT Max	Float64	59	Bwd Byts/b Avg	Int64
20	Flow IAT Min	Float64	60	Bwd Pkts/b Avg	Int64
21	Fwd IAT Tot	Float64	61	Bwd Blk Rate Avg	Int64
22	Fwd IAT Mean	Float64	62	Subflow Fwd Pkts	Int64
23	Fwd IAT Std	Float64	63	Subflow Fwd Byts	Int64

24	Fwd IAT Max	Float64	64	Subflow Bwd Pkts	Int64
25	Fwd IAT Min	Float64	65	Subflow Bwd Byts	Int64
26	Bwd IAT Tot	Int64	66	Init Fwd Win Byts	Int64
27	Bwd IAT Mean	Float64	67	Init Bwd Win Byts	Int64
28	Bwd IAT Std	Float64	68	Fwd Act Data Pkts	Int64
29	Bwd IAT Max	Int64	69	Fwd Seg Size Min	Int64
30	Bwd IAT Min	Int64	70	Active Mean	Float64
31	Fwd PSH Flags	Int64	71	Active Std	Float64
32	Bwd PSH Flags	Int64	72	Active Max	Int64
33	Fwd URG Flags	Int64	73	Active Min	Int64
34	Bwd URG Flags	Int64	74	Idle Mean	Float64
35	Fwd Header Len	Int64	75	Idle Std	Float64
36	Bwd Header Len	Int64	76	Idle Max	Float64
37	Fwd Pkts/s	Float64	77	Idle Min	Float64
38	Bwd Pkts/s	Float64	78	Label	int64
39	Pkt Len Min	Int64			

Class Distribution

For our model to be unbiased, we need a class distribution of 50% for each class. However, in our dataset, the class distribution is imbalanced, with 1486518 data points of Malicious and 10625443 data points of Benign, as depicted in Fig 5.7 as well as in Table 5.2.

Table 5.2 Imbalanced Class Distribution

Class	Data points	Total Data Points
Benign	10625443	12,111,961
Malign	1486518	

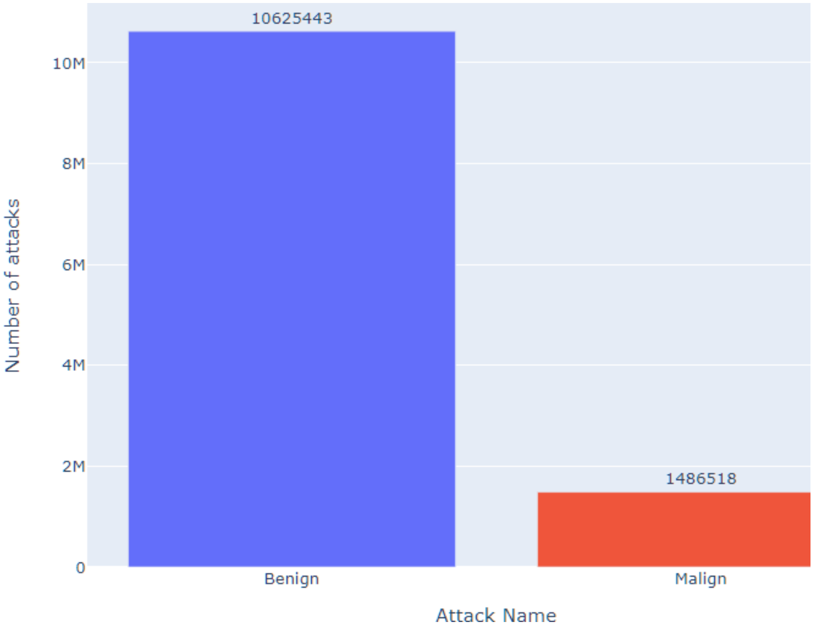


Figure 5.7 Imbalanced Class Distribution

Resampling methods are one of the most common approaches for dealing with imbalanced data, as mentioned in [35] Haixiang et al. paper. The process of resampling entails modifying the initial dataset to generate a new dataset that is more evenly distributed by either augmenting the quantity of samples in the minority class using oversampling or diminishing the quantity of samples in the majority class using undersampling.

To ensure a balanced class distribution in our model, we utilized Min class count. As a result, our final dataset now contains a total of 2973035 data points, as indicated in Table 5.3.

Table 5.3 Balanced Class Distribution

Class	Data points	Total Data Points
Benign	1486518	2,973,036
Malign	1486518	

Features selection

Once the data preparation stage is completed, the subsequent step is to pick out the appropriate features for the purpose of training the model. The process of selecting relevant features is known as ”feature selection” and involves extracting a subset of features from the original set based on a predefined selection criterion that identifies the most pertinent features for the dataset.

Correlation is a feature selection technique that is used to evaluate the degree and direction of the association between two variables. By analyzing correlations, it’s possible to identify patterns and dependencies between features in a dataset, which can be helpful in selecting the most relevant features for a machine learning model or identifying potential causal relationships between variables. The correlation coefficients most usually utilized include:

Pearson’s Correlation Coefficient

It mainly measures the linear relationship between two variables. It assumes that both variables have a normal distribution and that there is a linear relationship between them. Pearson’s coefficient ranges from -1 to 1.

Spearman’s Correlation Coefficient

It is non-parametric method for calculating correlation that assesses the monotonic relationships between two attributes. It is calculated by ranking the values of each variable and then computing the correlation between the ranks. Spearman’s coefficient ranges from -1 to 1.

Kendall’s Correlation Coefficient

It represents the ordinal association between two variables. It is calculated by comparing the number of concordant and discordant pairs of observations between the two variables. Kendall’s tau ranges from -1 to 1, where -1 indicates a perfect negative correlation, 0 indicates no correlation, and 1 indicates a perfect positive correlation. [36]

The final dataset, which consists of several CSV files, underwent a correlation analysis using these techniques. The feature columns with correlation values falling below -3% or above +3% were subsequently eliminated from the dataset.

So, the final dataset, as presented in Table 5.4, has 36 features and roughly 3 million data points.

Table 5.4 Dataset after Feature Selection

Data points	Features
2973035	36

Dataset Splitting

In this section, the dataset has been partitioned into three sets: Training Dataset, Validation Dataset, and Testing Dataset. The Training Dataset will be employed to train the model, while the Validation Dataset will be utilized for cross-validation to tackle the Overfitting problem. Once the model has been trained, the Testing Dataset will be utilized to assess its performance.

Table 5.5 demonstrates that the Training set constitutes 2,018,439 data points, which corresponds to 65% of the complete dataset. Similarly, the Validation set encompasses 356,196 data points, equivalent to 15% of the entire dataset, and the Testing set comprises 593,659 data points, which amounts to 20% of the total dataset.

Table 5.5 Data points of 3 DataSets after Splitting Dataset

DataSet	Data Points	Percentage of Dataset
Training Set	2,018,439	65%
Validation Set	356,196	15%
Testing Set	593,659	20%

Deep Learning Model Training

After setting up the data and features, the subsequent step is to train the deep learning model with the aim of developing a model that can effectively detect intrusions with minimal false positives and false negatives. For this purpose we have used ANN. A detailed depiction of the entire process involved in our model training can be observed in Fig 5.8.

Model decisions we made in our experiments are in line with earlier works in the area. Our approach utilized a MLP with five fully connected hidden layers, as illustrated in Figure 5.9. The choice of used in the hidden layers affect the behavior of the neurons, and they can range from linear to nonlinear. The activation functions used are tanh, relu and sigmoid. The accuracy of the model was measured using cross-entropy loss, and we used ADAM, a powerful optimization algorithm, with a batch size of 500, 20 epochs, and accuracy as the metric.

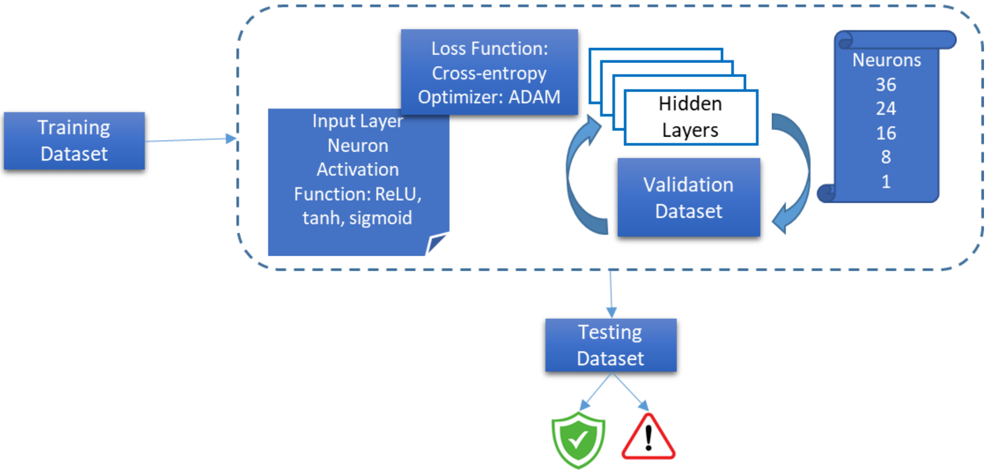

Training Dataset
Loss Function: Cross-entropy Optimizer: ADAM
Input Layer Neuron Activation Function: ReLU, tanh, sigmoid
Hidden Layers
Validation Dataset
Neurons 36 24 16 8 1
Testing Dataset

Figure 5.8 Steps of the Model Training

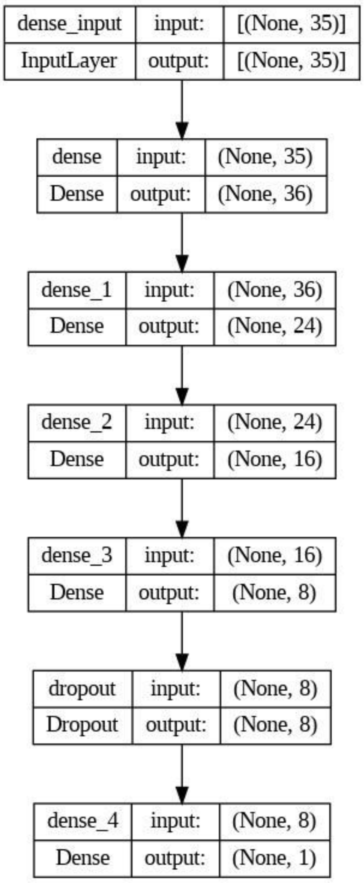

dense_input | input: | [(None, 35)]
InputLayer | output: | [(None, 35)]
dense | input: | (None, 35)
Dense | output: | (None, 36)
dense_1 | input: | (None, 36)
Dense | output: | (None, 24)
dense_2 | input: | (None, 24)
Dense | output: | (None, 16)
dense_3 | input: | (None, 16)
Dense | output: | (None, 8)
dropout | input: | (None, 8)
Dropout | output: | (None, 8)
dense_4 | input: | (None, 8)
Dense | output: | (None, 1)

Figure 5.9 Hidden Layers of our Model

In the following section, a brief explanation is provided for the terms mentioned above.

Hyperbolic Tangent (tanh) Function

It is similar to the sigmoid function but maps the input values to a range between -1 and 1. The tanh function has the same problem of vanishing gradients as the sigmoid function, but it is more suitable for multi-class classification problems.

Rectified Linear Unit (ReLU) Function

It is one of the most popular activation functions used in deep learning. The ReLU function maps all negative input values to zero, which allows for faster training and convergence in deep neural networks. However, it suffers from the problem of dying ReLU, where the gradients become zero for negative inputs, making the corresponding neurons useless for training.

Sigmoid Function

It is one of the earliest activation functions used in ANNs. The sigmoid function maps the input values to a range between 0 and 1, making it suitable for binary classification problems. However, it suffers from the problem of vanishing gradients, which can slow down the training process.[37]

Cross-entropy loss function

It is a stochastic optimization method which measures the distance between the predicted probabilities and the true probabilities of the output classes.

Adaptive Moment Estimation - ADAM

It is an adaptive learning rate optimization algorithm that estimates the first and second moments of the gradients to determine the learning rate for each weight. The method uses a moving average of the gradient and its square to calculate the first and second moments of the gradient, respectively. The algorithm then uses these moments to calculate an adaptive learning rate for each weight. [38]

Chapter 6

Results

The systems that were proposed in this project were created using the Python programming language and were implemented using several libraries, including Keras/Tensorflow, Seaborn, Sklearn, and pandas. These libraries provided important functionalities that were utilized throughout the development of the systems. The project was carried out using the Jupyter Notebook application within the Anaconda environment, which allowed for efficient and interactive development and testing of the models.

Implemented Methodology

The proposed methodology has been successfully implemented and is depicted in Figure 6.1

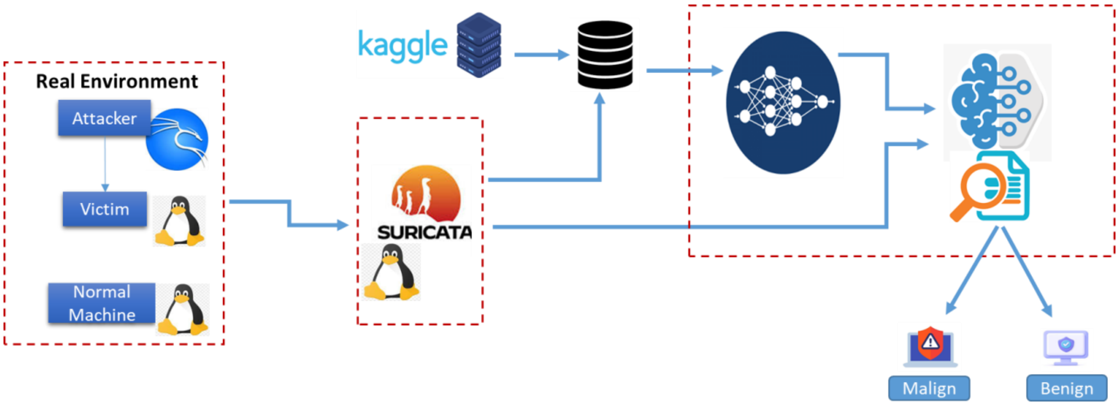


Figure 6.1 Implemented Methodology

Performance metrics were evaluated by conducting experiments on a workstation with the specifications indicated in Table 6.1

Table 6.1: Hardware Requirements

Components	**Operating System**	**RAM**	**Processors**	**Storage HDD**
Victim	Ubuntu 20.04	3GB	4	200GB
Attacker	Kali Linux	3GB	4	200GB
Normal Operations	Ubuntu 20.04	2GB	2	100GB
Suricata	Ubuntu 20.04	4GB	4	200GB
Training Machine	Windows 11	48GB	8	1TB

Loss vs Epoch Graph

The loss vs epoch graph as shown in Figure 6.2 is a visual representation of a machine learning model's performance during training, where the number of epochs is plotted on the x-axis and the model's loss (error) on the training set is plotted on the y-axis. The goal is to minimize the loss function as the number of epochs increases to achieve better performance on the training set. However, it is crucial to avoid overfitting, which happens when the model fits the training data too well and performs poorly on new data. The graph usually displays a decreasing trend of the loss over epochs.6.2

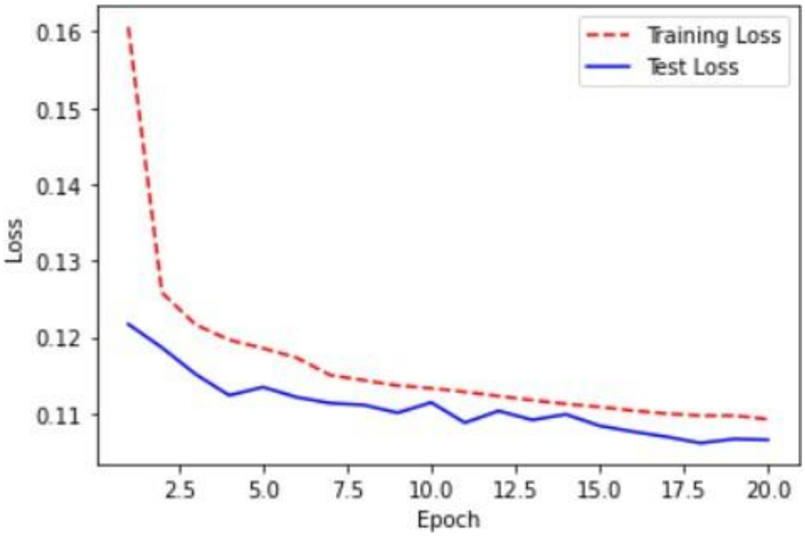


Figure 6.2 Loss vs Epoch Graph

Once the model was trained, it was evaluated using a Testing Dataset, and the resulting accuracy of the model was 96%. Additionally, a loss of 1% was recorded during this evaluation.

ROC Curve

The plot depicted in Figure 6.3 shows how effectively the binary classifier distinguishes between two classes (Benign or Malign) by comparing the True Positive rate with the False Positive rate at different threshold points. This classifier model operates on a dataset of 2973035 records, each with 36 features, and generates the ROC curve.[39]

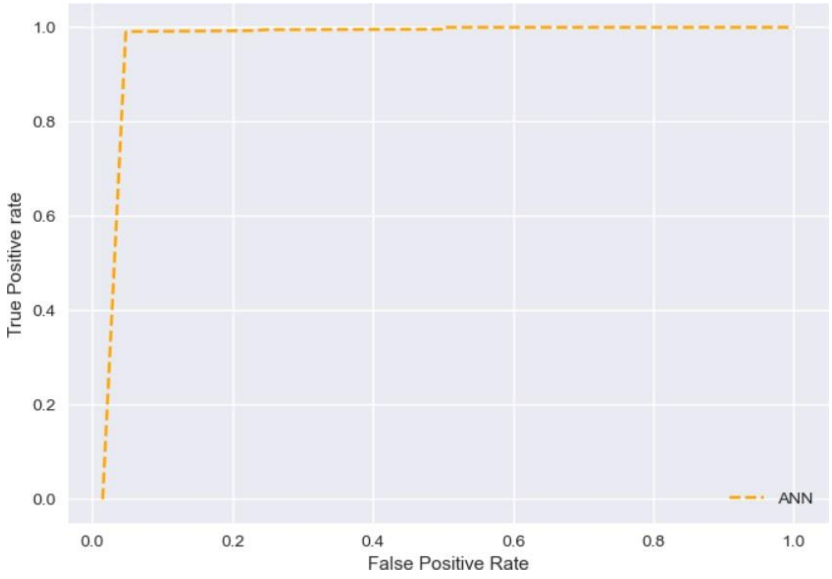


Figure 6.3 ROC Curve

Confusion Matrix

A confusion matrix summarizes the model’s predictions on a set of data for comparison with the actual outcomes. The matrix is made up of two dimensions - actual outcomes and predicted outcomes. The actual outcomes are listed in the rows, while the predicted outcomes are listed in the columns. Each cell in the matrix displays the number of times a specific actual outcome was predicted as a particular predicted outcome. [40] The four cells of the confusion matrix represent:

True positives

indicate the number of cases where the model correctly predicted the positive class.

False positives

indicate the number of cases where the model predicted the positive class, but the actual outcome was negative.

False negatives

indicate the number of cases where the model predicted the negative class, but the actual outcome was positive.

True negatives

indicate the number of cases where the model correctly predicted the negative class.

Table 6.2: Confusion Matrix

	Predicted Negative	Predicted Positive
Actual Negative	115 (TN)	95 (FP)
Actual Positive	0 (FN)	1000 (TP)

Based on the data presented in the table 6.2, we can conclude that the model accurately the model accurately identified 1000 network traffic flows as attack traffic (true positives), while it incorrectly classified 95 network traffic flows as attack traffic when they were actually normal traffic (false positives). Additionally, the model did not misclassified any attack traffic as normal traffic (false negatives), and it correctly identified 115 network traffic flows as normal traffic (true negatives). In addition to the table, a visual representation of the confusion matrix is also presented in Figure 6.4.

Performance Metrics

Using the values in the confusion matrix, various performance metrics such as accuracy, precision, recall, and F1-score can be calculated to assess the model's performance on the classification task.

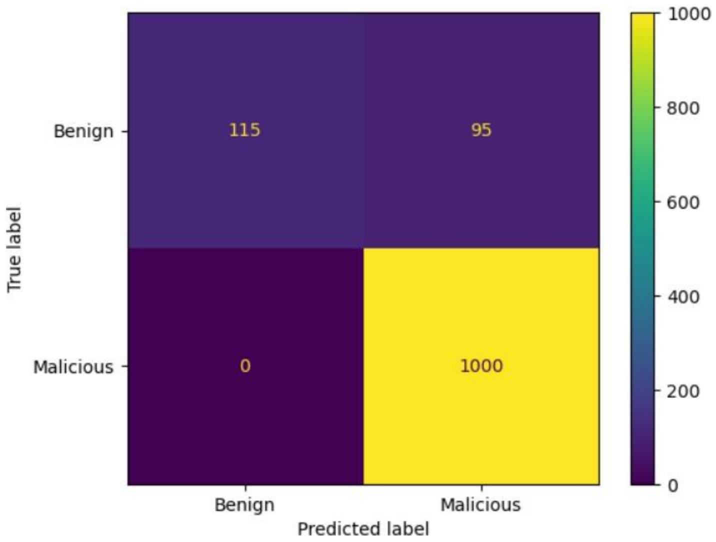


Figure 6.4 Confusion Matrix

Precision

It measures the proportion of true positive predictions out of all positive predictions made by the model.[40]

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

Recall

Recall or Detection rate calculates the ratio of correct positive predictions made by the model to the total number of actual positive cases in the dataset.[40]

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

Accuracy

Accuracy refers to the ratio of correct predictions made by the model to the total number of predictions made.[40]

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

F1-score

It is the harmonic mean of precision and recall, which gives equal weight to both metrics.[40]

$$F1\ score = 2(\frac{Precision\ x\ Recall}{Precision + Recall})$$

Table 6.3: Test data performance metrics

Accuracy	Precision	Recall	F1 score
0.92148	0.91324	1.0	0.95465

According to the performance metrics presented in Table 6.3, the model accurately predicted 92.1% of the traffic flows (accuracy score). The precision score of 0.913 suggests that when the model predicted an attack traffic, 91.3% of the time it was correct. The recall score of 1.0 implies that the model identified all the actual attack traffic flows. Moreover, the F1 score of 0.955, which is a combined measure of the model's precision and recall scores, indicates its overall effectiveness in classifying traffic flows.

Comparison with related work

We conducted a comprehensive comparison of our proposed model with other similar models as shown in Table 6.4 in order to evaluate their performance. The comparison was based on two key factors: accuracy and the implications of the models in real environment testing. Wen Xu et al. [29] proposed a 5-layer architecture model using the autoencoder algorithm to improve the efficiency of autoencoder-based network anomaly detection on the NSL-KDD dataset, achieving an accuracy of 90.61%. Their approach demonstrates enhanced anomaly detection capabilities. However, the model's effectiveness in real-world operational network environments requires further evaluation and research. Xianwei Gao et al. [30] presented an adaptive ensemble learning model for intrusion detection, achieving an accuracy of 85.20%. However, the paper has limitations regarding feature selection, comparative analysis, scalability, real-time adaptability, and practical deployment. Yongkuan Zhu et al. [31] investigated the use of data mining methods, specifically the FP-growth and Apriori algorithms, for enhancing security and detecting network intrusion. The authors emphasize the importance of real-time intrusion detection and acknowledge the challenges associated with it. They achieve an accuracy of 90.57% using the FP-growth and Apriori algorithms on the KDDCUP99 dataset. Chaouki Khammassi et al. [32] proposed a feature selection approach for network intrusion detection using a hybrid of NSGA2 and logistic regression. The method achieves high accuracy, with 99.65% on NSL-KDD and 94.90% on UNSW-NB15 datasets. However, certain attacks pose challenges in identification. The proposed method may be susceptible to overfitting, impacting performance on unseen data. Further research is needed to validate the applicability of the approach in real-world scenarios. Lastly, our proposed Network Intrusion Detection System (NIDS) utilizes a customized dataset that combines the updated CSE-CIC-2018 dataset with logs collected from real environments. The model achieves an impressive accuracy rate of 96% using an Artificial Neural Network (ANN) algorithm for detecting network intrusions. In testing with real-world environment traffic, the NIDS demonstrates a high level of accuracy, correctly classifying 92.1% of traffic flows. The model also achieves a precision score of 91.3% and a recall score of 1.0, indicating its effectiveness in accurately detecting network intrusions.

Table 6.4 Comparison with related work

Author	Year	Dataset	Algorithm	Accuracy
W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei, and F. Sabrina [29]	2021	NSL-KDD	Autoencoder	90.61%
X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu [30]	2019	NSL-KDD	Ensemble adaptive voting	85.20%
Y. Zhu, G. Gaba, F. Almansour, R. Alroobaea, and M. Masud [31]	2021	KDDCUP99	FP-growth Apriori	90.57%
C. Khammassi and S. Krichen [32]	2020	NSL-KDD	Decision tree	99.65%
		UNSW-NB15	Random Forest	94.90%
		CIC-IDS2017	Naive Bayes Tree	-
Proposed Model	2023	Customized (CIC-IDS2018 + other)	Artificial Neural Network	96%
		Real Environment logs		92.10%

Chapter 7

Conclusion

After successfully training our model and achieving an accuracy rate of 96% on the testing dataset, we proceeded to test it on real-world data. The results showed that the model was able to correctly classify 92.1% of the traffic flows with an accuracy score. We also calculated the precision score, which indicated that when the model predicted an attack traffic, it was correct 91.3% of the time. The recall score was 1.0, indicating that all actual attack traffic flows were correctly identified by the model.

Based on these statistics, it is recommended that the model should be trained on an organization's normal routine traffic to avoid false positives in the future. This would help to increase the accuracy of the model by training it on the specific patterns and characteristics of the organization's network traffic. Additionally, it is important to continuously update the model with new attack patterns as they emerge, to ensure that it stays up-to-date with the latest threats and can detect them effectively.

To ensure the optimal performance of the model, it is necessary to monitor its accuracy regularly and make adjustments as needed. This involves retraining the model with new data and optimizing its parameters, so that it can continue to accurately detect attacks and minimize false positives. By continuously improving the model's performance, organizations can improve the overall security of their networks and protect against potential threats.

In the future, an additional avenue for development would be to write a source code for the pre-trained model to integrate it with Suricata could be a potential area of development in the future. This integration would enable the model to act as an additional layer of defense against attacks in a network environment. By utilizing the capabilities of the model, the organization could enhance the overall security of their system. This would involve implementing the necessary code to integrate the model with Suricata, allowing for its seamless use in a network security setting. By doing so, the organization could take advantage of the model's pre-existing capabilities and strengthen their overall security posture.

Bibliography

[1] M. Meyen, S. Pfaff-Rüdiger, K. Dudenhöffer, and J. Huss, “The internet in everyday life: a typology of internet users,” *Media, Culture & Society*, vol. 32, no. 5, pp. 873–882, 2010. 1

[2] S. P.S, N. Sundaresan, and S. M, “Overview of cyber security,” *IJARCCE*, vol. 7, pp. 125–128, 11 2018. 1

[3] A. R. b. Gupta and J. Agrawal, “A comprehensive survey on various machine learning methods used for intrusion detection system,” in *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 282–289, 2020. 1

[4] R. von Solms and J. van Niekerk, “From information security to cyber security,” *Computers & Security*, vol. 38, pp. 97–102, 2013. Cybercrime in the Digital Economy. 2.1, 2.1

[5] S. Samonas and D. Coss, “The cia strikes back: Redefining confidentiality, integrity and availability in security.,” *Journal of Information System Security*, vol. 10, no. 3, 2014. 2.1.1, 2.1.2, 2.1.3

[6] K. Ingham, S. Forrest, *et al.*, “A history and survey of network firewalls,” *University of New Mexico, Tech. Rep*, 2002. 2.2.1

[7] A. A. Jaha, F. B. Shatwan, and M. Ashibani, “Proper virtual private network (vpn) solution,” in *2008 The Second International Conference on Next Generation Mobile Applications, Services, and Technologies*, pp. 309–314, 2008. 2.2.2

[8] A. S. Ashoor and S. Gore, “Importance of intrusion detection system (ids),” *International Journal of Scientific and Engineering Research*, vol. 2, no. 1, pp. 1–4, 2011. 2.2.4

[9] S. E. Smaha *et al.*, “Haystack: An intrusion detection system,” in *Fourth Aerospace Computer Security Applications Conference*, vol. 44, p. 37, Orlando, FL, USA, 1988. 2.2.4

[10] R. G. Bace, *Intrusion detection.* Sams Publishing, 2000. 2.2.4

[11] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Review: Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, p. 16–24, jan 2013. 2.2.4.2

[12] N. Maharaj and P. Khanna, "Article: A comparative analysis of different classification techniques for intrusion detection system," *International Journal of Computer Applications*, vol. 95, pp. 22–26, June 2014. Full text available. 2.2.4.2

[13] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in internet of things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017. 2.2.4.3

[14] Akashdeep, I. Manzoor, and N. Kumar, "A feature reduced intrusion detection system using ann classifier," *Expert Systems with Applications*, vol. 88, pp. 249–257, 2017. 2.2.4.4

[15] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: Methods, systems and tools," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 303–336, 2014. 2.2.5.1

[16] S. M. Othman, N. T. Alsohybe, F. M. Ba-Alwi, and A. T. Zahary, "Survey on intrusion detection system types," *International Journal of Cyber-Security and Digital Forensics*, vol. 7, no. 4, pp. 444–463, 2018. 2.2.5.2

[17] A. A. Abdelkarim and H. H. Nasereddin, "Intrusion prevention system," *International journal of academic research*, vol. 3, no. 1, p. 201, 2011. 2.2.6

[18] G. Carl, G. Kesidis, R. R. Brooks, and S. Rai, "Denial-of-service attack-detection techniques," *IEEE Internet computing*, vol. 10, no. 1, pp. 82–89, 2006. 2.3.2

[19] G. Mantas, N. Stakhanova, H. Gonzalez, H. H. Jazi, and A. A. Ghorbani, "Application-layer denial of service attacks: taxonomy and survey," *International Journal of Information and Computer Security*, vol. 7, no. 2-4, pp. 216–239, 2015. 2.3.2

[20] A. Mallik, "Man-in-the-middle-attack: Understanding in simple words," *Cyberspace: Jurnal Pendidikan Teknologi Informasi*, vol. 2, no. 2, pp. 109–134, 2019. 2.3.3

[21] J. R. v. d. Merwe, X. Zubizarreta, I. Lukčin, A. Rügamer, and W. Felber, "Classification of spoofing attack types," in *2018 European Navigation Conference (ENC)*, pp. 91–99, 2018. 2.3.4

[22] D. Stiawan, M. Idris, R. F. Malik, S. Nurmaini, N. Alsharif, R. Budiarto, *et al.*, "Investigating brute force attack patterns in iot network," *Journal of Electrical and Computer Engineering*, vol. 2019, 2019. 2.3.5

[23] A. Kamruzzaman, S. Ismat, J. C. Brickley, A. Liu, and K. Thakur, "A comprehensive review of endpoint security: Threats and defenses," in *2022 International Conference on Cyber Warfare and Security (ICCWS)*, pp. 1–7, IEEE, 2022. 2.3.6

[24] M. A. Lahmeri, M. Kishk, and M.-S. Alouini, "Artificial intelligence for uav-enabled wireless networks: A survey," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 1015–1040, 01 2021. 3.4.1, 3.4.2, 3.4.3, 3.5.2, 3.5.3

[25] H. Kukreja, N. Bharath, C. Siddesh, and S. Kuldeep, "An introduction to artificial neural network," *Int J Adv Res Innov Ideas Educ*, vol. 1, pp. 27–30, 2016. 3.5

[26] Q. Chen, W. Zhang, and Y. Lou, "Forecasting stock prices using a hybrid deep learning model integrating attention mechanism, multi-layer perceptron, and bidirectional long-short term memory neural network," *IEEE Access*, vol. 8, pp. 117365–117376, 2020. 3.5.1.1

[27] J. P. Anderson, "Computer security threat monitoring and surveillance," *Technical Report, James P. Anderson Company*, 1980. 4.1

[28] K. Ilgun, R. Kemmerer, and P. Porras, "State transition analysis: a rule-based intrusion detection approach," *IEEE Transactions on Software Engineering*, vol. 21, no. 3, pp. 181–199, 1995. 4.1

[29] W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei, and F. Sabrina, "Improving performance of autoencoder-based network anomaly detection on nsl-kdd dataset," *IEEE Access*, vol. 9, pp. 140136–140146, 2021. 4.1, 6.6, 6.4

[30] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An adaptive ensemble machine learning model for intrusion detection," *IEEE Access*, vol. 7, pp. 82512–82521, 2019. 4.1, 6.6, 6.4

[31] Y. Zhu, G. S. Gaba, F. M. Almansour, R. Alroobaea, and M. Masud, "Application of data mining technology in detecting network intrusion and security maintenance," *Journal of Intelligent Systems*, vol. 30, no. 1, pp. 664–676, 2021. 4.1, 6.6, 6.4

[32] C. Khammassi and S. Krichen, "A nsga2-lr wrapper approach for feature selection in network intrusion detection," *Computer Networks*, vol. 172, p. 107183, 2020. 4.1, 6.6, 6.4

[33] 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*, pp. 122–127, 2012. 5.2.1.2

[34] D. Fadhilah and M. I. Marzuki, "Performance analysis of ids snort and ids suricata with many-core processor in virtual machines against dos/ddos attacks," in *2020 2nd International Conference on Broadband Communications, Wireless Sensors and Powering (BCWSP)*, pp. 157–162, 2020. 5.2.1.2

[35] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: Review of methods and applications," *Expert Systems with Applications*, vol. 73, pp. 220–239, 2017. 5.2.3

[36] G. P. Dubey and D. R. K. Bhujade, "Optimal feature selection for machine learning based intrusion detection system by exploiting attribute dependence," *Materials Today: Proceedings*, vol. 47, pp. 6325–6331, 2021. SI: TIME-2021. 5.2.4.3

[37] A. D. Rasamoelina, F. Adjailia, and P. Sinčák, "A review of activation function for artificial neural network," in *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pp. 281–286, 2020. 5.2.6.3

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. 5.2.6.5

[39] V. Kanimozhi and T. P. Jacob, "Artificial intelligence based network intrusion detection with hyper-parameter optimization tuning on the realistic cyber dataset cse-cic-ids2018 using cloud computing," *ICT Express*, vol. 5, no. 3, pp. 211–214, 2019. 6.3

[40] Z. Ahmad, A. Shahid Khan, C. Shiang, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, 01 2021. 6.4, 6.5.1, 6.5.2, 6.5.3, 6.5.4