

Privacy-Aware Intrusion Detection System

submitted by

Meghasri Saha

Class Roll No: 002210503031

Examination Roll No: MCA244030

Master of Computer Application

under the supervision of

Prof. (Dr.) Chandreyee Chowdhury

in the

Department of Computer Science and Engineering

Faculty of Engineering and Technology

Jadavpur University

2024

Faculty of Engineering and Technology
Jadavpur University

CERTIFICATE OF RECOMMENDATION

To Whom It May Concern

I hereby recommend that the project report entitled "Privacy-Aware Intrusion Detection System" has been carried out by Meghasri Saha (Reg. No.: 163623 of 2022-23, Roll No.: 002210503031, Exam Roll: MCA244030), under my guidance and supervision and be accepted in partial fulfilment of the requirement for the degree of Master of COMPUTER APPLICATION in DEPARTMENT of COMPUTER SCIENCE and ENGINEERING, JADAVPUR UNIVERSITY during the academic year 2023-2024.

Prof. (Dr.) Chandreyee Chowdhury
Project Supervisor
Dept. of Computer Science and
Engineering

Prof. (Dr.) Nandini Mukhopadhyay
Head of the Department
Dept. of Computer Science & Engineering
Jadavpur University

Prof. Dipak Laha
Dean, Faculty Council
of Engineering & Technology
Jadavpur University, Kolkata-032

Faculty of Engineering and Technology
Jadavpur University

CERTIFICATE OF APPROVAL

This is to certify that the project report entitled “**Privacy-Aware Intrusion Detection System**” is a bonafide record of work carried out by MEGHASRI SAHA, in fulfillment of the requirements for the award of the degree of Master of Computer Application in the Department of Computer Science and Engineering, Jadavpur University. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the project report only for the purpose for which it has been submitted.

Signature of Examiner 1

Date :

Signature of Examiner 2

Date :

Faculty of Engineering and Technology
Jadavpur University

**DECLARATION OF COMPLIANCE WITH
ACADEMIC ETHICS**

I hereby declare that this project report entitled "Privacy-Aware Intrusion Detection System" is an outcome of my genuine and independent work, as part of my Master of Computer Application program.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, all materials and results that are not original to this work have been fully cited and referenced.

MEGHASRI SAHA

Registration No.: 163623 of 2022-23

Examination Roll No.: MCA244030

Thesis Title :- Privacy-Aware Intrusion Detection System

Signature with Date

ACKNOWLEDGEMENT

With my most sincere and gratitude, I would like to thank Prof. (Dr.) Chandreyee Chowdhury, Department of Computer Science & Engineering, my supervisor, for her overwhelming support throughout the duration of the project. Her motivation always gave me the required inputs and momentum to continue with my work, without which the project work would not have taken its current shape. Her valuable suggestions and numerous discussions have always inspired new ways of thinking. I feel deeply honored that I got this opportunity to work under her.

I would like to express my sincere thanks to all my teachers for providing sound knowledge base and cooperation.

I would like to thank all the faculty members of the Department of Computer Science & Engineering of Jadavpur University for their continuous support.

Last, but not the least, I would like to thank my batch mates for staying by my side when I needed them.

Meghasri Saha

Class Roll No.: 002210503031

Examination Roll No.: MCA244030

Department of Computer Science & Engineering

Jadavpur University

ABSTRACT

Intrusion Detection System (IDS) are vital for monitoring computer system and networks to detect unauthorized access or attacks. As cyber threats from insiders and outsiders become more sophisticated, safeguarding digital information is crucial. IDS use two main methods: Signature-based (SIDS), which detect known attack patterns, and Anomaly-based (AIDS), which identify unusual activities. By analyzing network traffic for suspicious behavior, IDS quickly raise alerts and log detailed records for investigation, maintaining network integrity and confidentiality.

In this study, we evaluated the effectiveness of various machine learning models for IDS. We tested Support Vector Machine (SVM), k-Nearest Neighbor (kNN), Random Forest, and eXtreme Gradient Boosting (XGBoost) with a dataset of attack types. Random Forest outperformed the other models, achieving an accuracy of 99.9958%. Its ability to handle many features and avoid overfitting makes it highly suitable for IDS.

Table of Contents

1. Introduction.....	1
1.1 Aim.....	1
1.2 Types of Intrusion Detection System.....	1
1.3 Key Functions of IDS.....	2
1.4 Importance of IDS.....	2
1.5 Challenges.....	2
1.6 Motivation.....	2
1.7 Contribution.....	2
2. Literature Review.....	3
3. Methodology.....	5
3.1 Intrusion Detection Datasets.....	5
3.1.1 Comparison of Public IDS Datasets.....	6
3.1.2 Feature Selection For IDS.....	7
3.1.3 Data Visualization.....	7
3.1.4 Types of Computer Attacks.....	9
3.2 Data Preprocessing.....	9
3.2.1 Data Normalization.....	9
3.3 Classification	10
3.3.1 Binary Classification of Train Data.....	10
3.3.2 Multi-Class Classification of Train Data.....	11
3.3.3 Model Selection and Model Training.....	11
3.3.4 Description of Different Models	12
4. Experimental Result.....	13
4.1 Parameter Table.....	13
4.1.1 Parameter Table of SVM.....	13
4.1.2 Parameter Table of kNN.....	14
4.1.3 Parameter Table of RF.....	14
4.1.4 Parameter Table of XGBoost.....	15
4.2 Accuracy with Combination of Various Attributes.....	16
5. Experimental Setup.....	17
5.1 Confusion Matrix.....	17
5.1.1 Evaluating Trained SVM Using Confusion Matrix.....	18
5.1.2 Evaluating Trained kNN Using Confusion Matrix.....	19
5.1.3 Evaluating Trained RF Using Confusion Matrix.....	20
5.1.4 Evaluating Trained XGBoost Using Confusion Matrix.....	21
5.2 Accuracy of Different Models.....	22
5.3 Visual Representation of Different Models.....	23
6. Conclusion.....	24
7. References.....	25

LIST OF ABBREVIATIONS AND SYMBOLS

IDS	Intrusion Detection System
SVM	Support Vector Machine
kNN	k-Nearest Neighbor
RF	Random Forest
XGBoost	eXtreme Gradient Boosting

Chapter 1

Introduction

In today's interconnected world, securing network infrastructure and data is most important. An IDS plays a critical role in safeguarding computer networks from unauthorized access, misuse, and attacks. An IDS monitors network traffic and system activities for malicious activities, raising alerts when potential threats are detected.

1.1 Aim

The aim is to create a Privacy-Aware Intrusion Detection System (IDS) that effectively detects and mitigates security threats while preserving user privacy. The objectives include designing a system that accurately identifies intrusions, incorporates privacy-preserving techniques, enables real-time detection, and minimizes false positives. Additionally, the IDS should be scalable, user-friendly, compliant with legal regulations, and perform well in various network privacy environments. This IDS seeks to balance robust security measures with the need to maintain data privacy and confidentiality.

1.2 Types of Intrusion Detection System

IDS is broadly classified into two categories as shown in Fig1:

Network-based Intrusion Detection System (NIDS):

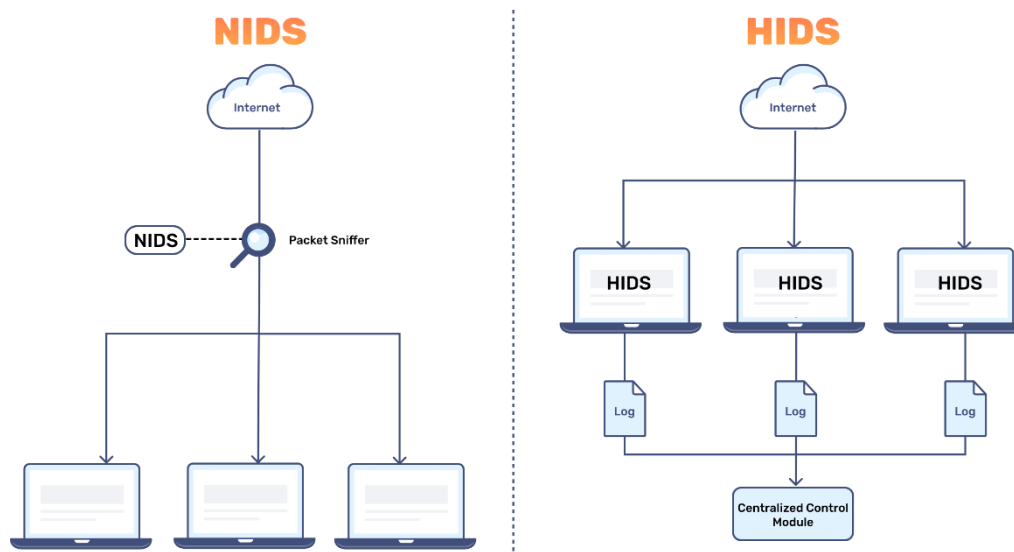
NIDS is deployed at strategic points within a network to monitor traffic to and from all devices on the network.

They analyze passing network traffic for signs of intrusions, such as suspicious patterns or known attack signatures.

Host-based Intrusion Detection System (HIDS):

HIDS is installed on individual hosts or devices to monitor the inbound and outbound packets from the device only.

They examine system logs, file integrity, and other host-specific actions to detect unusual behavior.



[Figure.1](#): Intrusion Detection System – Types [2]

	Advantages	Disadvantages
Technology	HIDS • HIDS can check end-to-end encrypted communications behaviour. • No extra hardware required. • Detects intrusions by checking hosts file system, system calls or network events. • Every packet is reassembled • Looks at the entire item, not streams only	• Delays in reporting attacks • Consumes host resources • Needs to be installed on each host. • It can monitor attacks only on the machine where it is installed.
	NIDS • Detects attacks by checking network packets. • Not required to install on each host. • Can check various hosts at the same period. • Capable of detecting the broadest ranges of network protocols	• Challenge is to identify attacks from encrypted traffic. • Dedicated hardware is required. • It supports only identification of network attacks. • Difficult to analysis high-speed network. • The most serious threat is the insider attack.

Table 1: Comparison of IDS technology types based on their positioning within the computer system[1]

1.3 Key Functions of IDS :-

1. **Monitoring:** Continuously observes network traffic and system activity to detect unusual patterns.
2. **Analysis:** Evaluates the data collected to identify potential threats by comparing them against known attack signatures and behavioral patterns.
3. **Alerting:** Notifies administrators of detected suspicious activities or policy violations in real-time, enabling quick response and mitigation.
4. **Response:** Some advanced IDS can automatically take predefined actions to counter detected threats, such as blocking traffic from suspicious IP addresses.

1.4 Importance of IDS

1. **Early Threat Detection:** An IDS can detect intrusions at an early stage, allowing for timely intervention before significant damage occurs.
2. **Enhanced Security:** By monitoring network and system activities, IDS adds an additional layer of defense, complementing other security measures like firewalls and antivirus software.
3. **Compliance:** Many regulations and standards require the implementation of IDS to ensure data protection and secure network practices.
4. **Incident Response:** Provides valuable data and alerts that aid in the rapid response and investigation of security incidents.

1.5 Challenges

1. **False Positives/Negatives:** IDS may sometimes raise false alarms or miss actual threats, necessitating fine-tuning and regular updates.
2. **Performance:** High traffic volumes can affect the performance of IDS, requiring robust hardware and optimized configurations.

IDS is indispensable in the modern cybersecurity landscape. By continuously monitoring and analyzing network and system activities, IDS helps in maintaining the integrity, confidentiality, and availability of critical information assets.

1.6 Motivation

IDS is important because they help to protect against cyberattacks, keep sensitive information safe, and meet legal requirements. IDS can quickly spot and respond to threats, giving a clear view of what is happening on a network. It helps catch problems early, provide useful information for investigations, and prevent future attacks. By using IDS, organizations can avoid the high costs of data breaches and improve their overall security.

1.7 Contribution

IDS contribute to cybersecurity by quickly identifying and responding to threats, protecting sensitive information. They provide real-time monitoring, help prevent data breaches, and offer valuable information for investigating security incidents. Overall, IDS enhances the security and resilience of an organization's network.

Chapter 2

Literature Review

IDS is a tool that monitor computer system and analyze events to detect any signs of unauthorized access or attacks[4].Today's computer system are at risk from both insiders who misuse their access and outsiders who break in. It's impossible to close all security gaps in these systems, and no technology can fully stop users with permission from doing bad things. That's why auditing, or keeping track of what users do, is the last line of defense.[1]

This document explains different types of computer intrusions and briefly describes the main types of computer attacks. It also explains the common structure and basic features of intrusion detection system. Then, it provides a classification of IDS based on five criteria: information source, analysis strategy, timing, architecture, and response. Finally, it classifies IDS according to these criteria and briefly describes the most notable research prototypes in the field[4].

A host-based anomaly IDS is difficult because it often triggers too many false alarms. This paper presents a new approach to host-based anomaly intrusion detection that uses non-sequential system call patterns. This method aims to improve detection rates while reducing false alarms. The main idea is to organize kernel-level system calls in a way that reflects the actual activities of high-level programming languages, helping to better understand abnormal program behavior[7].

We tested this approach using various decision engines and evaluated it with the KDD98 and UNM datasets, as well as a new, modern dataset called the ADFA Linux dataset. This dataset was created as part of our research using a modern operating system and contemporary hacking techniques, and it is now publicly available. Additionally, this new semantic method is naturally resistant to mimicry attacks and works well across different versions of operating system[7].

Over the years, experts have been creating tools that automatically check audit logs for unusual user behavior. This document explains how these tools help find intrusions in computer system and explores new technologies that could improve intrusion detection in the future.[1]

The open-source tool Snort is used for intrusion prevention. It is set up to monitor network traffic and block any suspicious activity. When something suspicious is found, Snort alerts the administrator and drops the bad data packets. These dropped packets are then logged using Splunk. This solution is cost-effective, easy to customize, and can grow with the needs of the organization [8].

The growth of the internet and computer system, a lot of data has gone digital, leading to major issues like security, privacy, and confidentiality. While computer system security has improved significantly, these issues remain serious concerns. Using just a firewall is not enough to address these Cloud security issues. This paper focuses on using other methods, like intrusion detection and prevention system (IDS/IPS), to enhance Cloud security[9].This document provides an overview of IDS used in various areas, including Web Applications, Cloud Environments, the Internet of Things (IoT), Mobile Ad-Hoc Networks (MANET), Wireless Sensor Networks (WSN), and Voice over Internet Protocol (VoIP) [3].

In today's world cloud computing uses the internet to provide virtual servers that offer software, infrastructure, platforms, and other resources to customers on a daily basis. As cloud usage grows, so does the risk of security breaches. An IDS is a key strategy for enhancing security. The paper, propose a fast, efficient, and secure IDS that combines a multi-threaded Network Intrusion Detection System (NIDS) with a Host Intrusion Detection System (HIDS)[5].

In the current system, Cloud-IDS captures network packets, analyzes them, and sends reports to the Cloud Administrator based on the analysis. The analysis uses a hybrid classifier combining kNN and Neural Network (KNN-NN). It uses the NSL-KDD dataset for training and testing. Once the Cloud-IDS generates a report, the Cloud Service Provider (CSP) alerts the user and keeps a log of malicious IP addresses. The proposed model efficiently handles large data flows, analyzes them, and generates reports by integrating both anomaly detection and misuse detection methods[5].

A Wireless Sensor Network (WSN) is a large network that can range from dozens to thousands of tiny devices. WSNs are used in many fields, such as the military, healthcare, and smart homes, and their use is growing rapidly. Security is a major concern for WSNs, and they have significant security weaknesses. An IDS acts as a second layer of security for networks and is crucial for maintaining integrity, confidentiality, and availability[6].

In today's world, nothing is completely flawless. So, the search for better solutions continues. In this effort, we present an IDS that uses a genetic algorithm (GA) to efficiently detect various types of network intrusions. This method uses the principles of evolution to filter traffic data, reducing complexity. To test and measure our system's performance, it used the KDD99 benchmark dataset and achieved a good detection rate.[10]

CHAPTER 3-

Methodology

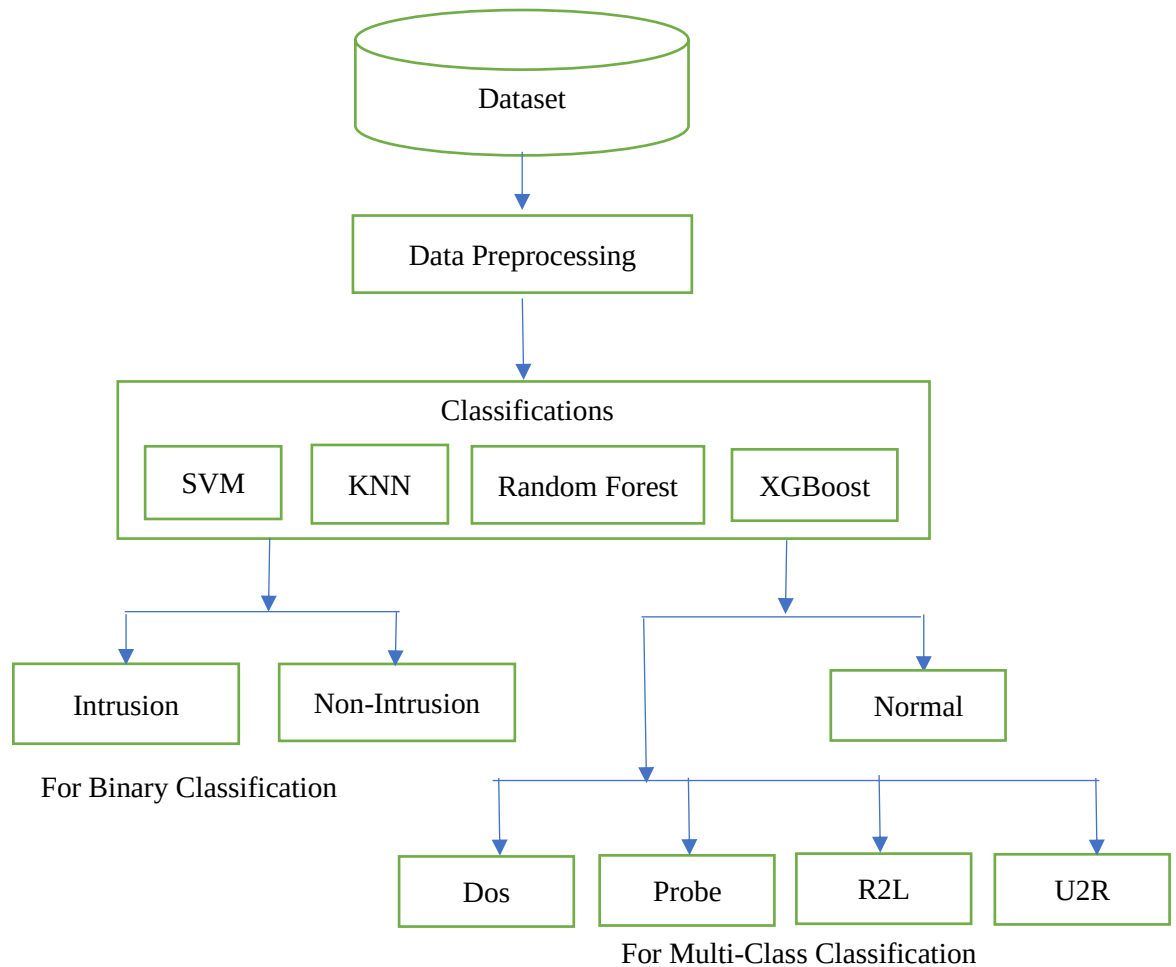


Fig 2: Workflow For The Methodology

3.1 Intrusion Detection Datasets

Evaluation datasets are crucial for validating the effectiveness of any IDS approach, as they allow researchers and practitioners to assess the system's capability in detecting intrusive behavior. However, obtaining suitable datasets for network packet analysis poses challenges, primarily due to privacy concerns associated with real-world network traffic. Among them are datasets such as DARPA, KDD, NSL-KDD, which are widely utilized in the research community. These datasets offer diverse features and characteristics that mimic real-world network traffic scenarios, helps researchers to test and compare the performance of different IDS techniques in detecting various types of attacks.

DARPA / KDD Cup99^[1]

The DARPA (Defense Advanced Research Projects Agency) made one of the earliest attempts to create a dataset for IDS in 1998, resulting in the KDD98 (Knowledge Discovery and Data Mining) dataset. This dataset was compiled using multiple computers connected to the Internet to simulate a small US Air Force base. The data collection process involved capturing both network packets and host log files. The LAN to mimic real-world conditions, they intentionally injected simulated intrusions to create a diverse set of network activities. The resulting dataset comprises a series of TCP sessions with clearly defined start and end times, representing data flows between source and target IP addresses. The 1998 DARPA Dataset was used as the basis to derive the KDD Cup99 Dataset.

CAIDA^[2]

The dataset mentioned contains network traffic traces from Distributed Denial-of-Service (DDoS) attacks and was compiled in 2007. DDoS attacks aim to disrupt the normal operation of a targeted computer or network by overload with a flood of network packets, thereby preventing legitimate traffic from reaching its intended destination. However, one limitation of the CAIDA dataset is its lack of diversity in representing various types of attacks. Additionally, the collected data lacks comprehensive features from the entire network, making it challenging to differentiate between abnormal and normal traffic flows effectively.

NSL-KDD^[3]

The NSL-KDD dataset, developed from the earlier KDD Cup'99 dataset, addresses a significant issue present in the original dataset: widespread of duplicate packets. In the KDD dataset, both the training and test sets suffered from a large number of duplicated network packets, with approximately 78% and 75% duplication rates, respectively. To resolve this issue, the NSL-KDD dataset was created in 2009 by eliminating these duplicated records.

Comprising 125,973 records in the training set and 22,544 records in the test set, the NSL-KDD dataset offers a sufficient size for practical use without the need for random sampling. This characteristic has led to consistent and comparable results across various research works. The dataset includes 22 different types of intrusion attacks in the training set and features 41 attributes (or features). Among these, 21 attributes pertain to the connection itself, while the remaining 19 attributes describe the nature of connections within the same host. This refined dataset has become a valuable resource for evaluating IDS, providing researchers with a standardized and reliable benchmark for their experiments.

Among all of these datasets I have used NSL-KDD dataset in my project.

3.1.1 Comparison of Public IDS Datasets

The selection of datasets for machine learning techniques applied in Anomaly-based Intrusion Detection System (AIDS) plays a crucial role in assessing the effectiveness of these techniques for realistic evaluation. Table 2 outlines the key characteristics of these datasets, helping researchers in understanding the scope, diversity, and complexity of the data available for training and testing machine learning models. By leveraging these datasets, researchers can gain a comprehensive understanding of the strengths and limitations of various machine learning techniques in the context of intrusion detection, enabling informed decision-making and further advancements in the field.

Dataset	Realistic Traffic	Label data	IoT traces	Zero-day attacks	Full packet captured	Year
DARPA 98	✓	✓	✗	✗	✓	1998
KDDCUP 99	✓	✓	✗	✗	✓	1999
CAIDA	✓	✗	✗	✗	✗	2007
NSL-KDD	✓	✓	✗	✗	✓	2009
ISCX 2012	✓	✓	✗	✗	✓	2012
ADFA-WD	✓	✓	✗	✓	✓	2014
ADFA-LD	✓	✓	✗	✓	✓	2014
CICIDS2017	✓	✓	✗	✓	✓	2017
Bot-IoT	✓	✓	✓	✓	✓	2018

Table 2: Comparison between datasets [1]

3.1.2 Feature Selection For IDS

Feature selection is a crucial step in developing an effective IDS using the NSL-KDD dataset. This dataset has many features related to network traffic that can help detect intrusions as shown in the Table 3.

Type	Name	Description	Numerical Type
Basic	duration	connection duration	continuous
	protocol_type	protocol type	discrete
	service	targeted network service type	discrete
	src_bytes	number of bytes sent from source to destination	continuous
	dst_bytes	number of bytes sent from destination to source	continuous
	flag	the connection is normal or not	discrete
	land	whether the connection is from/to the same host/port	discrete
	wrong_fragment	number of "wrong" fragment	continuous
Traffic	urgent	number of urgent packets	continuous
	count	number of connections to the same host in the first two seconds	continuous
	serror_rate	"SYN" error on the same host connection	continuous
	rerror_rate	"REJ" error on the same host connection	continuous
	same_srv_rate	number of of same service connected to the same host	continuous
	diff_srv_rate	number of of different services connected to the same host	continuous
	srv_count	number of connections to the same service in the first two seconds	continuous
	srv_error_rate	"SYN" error on the same service connection	continuous
	srv_rerror_rate	"REJ" error on the same service connection	continuous
	srv_diff_host_rate	number of different targeted host connected to the same service	continuous

Table 3: Features of NSL-KDD dataset [1]

Choosing the most important features improves the IDS's accuracy and speed. Techniques for feature selection include using statistical methods, machine learning, and expert knowledge. By focusing on the most relevant features, we make the model simpler, reduce the chance of errors, and make it run faster and better.

3.1.3 Data Visualization

Here is a table from the NSL-KDD dataset, which is widely used for evaluating IDS. This dataset includes 41 features, such as 'duration', and 'src_bytes', each providing crucial information about network connections. The 'label' column indicates whether each connection is normal or an attack type like DoS, R2L, U2R, or probing.

In this graph there is a column named 'difficulty' has been removed as because column provided a

predefined measure of the complexity of detecting certain attacks. Machine learning models might rely on this column instead of learning the underlying patterns of the data, leading to biased results. Examining these features helps in identifying patterns that distinguish normal behavior from malicious activity.

duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_same_srv_rate	dst_host_diff_srv_rate	dst_host_same_src_port_rate
0	0	tcp	ftp_data	SF	491	0	0	0	0	0	...	0.17	0.03
1	0	udp	other	SF	146	0	0	0	0	0	...	0.00	0.60
2	0	tcp	private	SO	0	0	0	0	0	0	...	0.10	0.05
3	0	tcp	http	SF	232	8153	0	0	0	0	...	1.00	0.00
4	0	tcp	http	SF	199	420	0	0	0	0	...	1.00	0.00
...	...	...	...	...	...	...	...	...	...	...	...	...	...
125968	0	tcp	private	SO	0	0	0	0	0	0	...	0.10	0.06
125969	8	udp	private	SF	105	145	0	0	0	0	...	0.06	0.01
125970	0	tcp	smtp	SF	2231	384	0	0	0	0	...	0.12	0.06
125971	0	tcp	klogin	SO	0	0	0	0	0	0	...	0.03	0.05
125972	0	tcp	ftp_data	SF	151	0	0	0	0	0	...	0.30	0.03

125973 rows × 43 columns

Fig 3: Dataset with its features

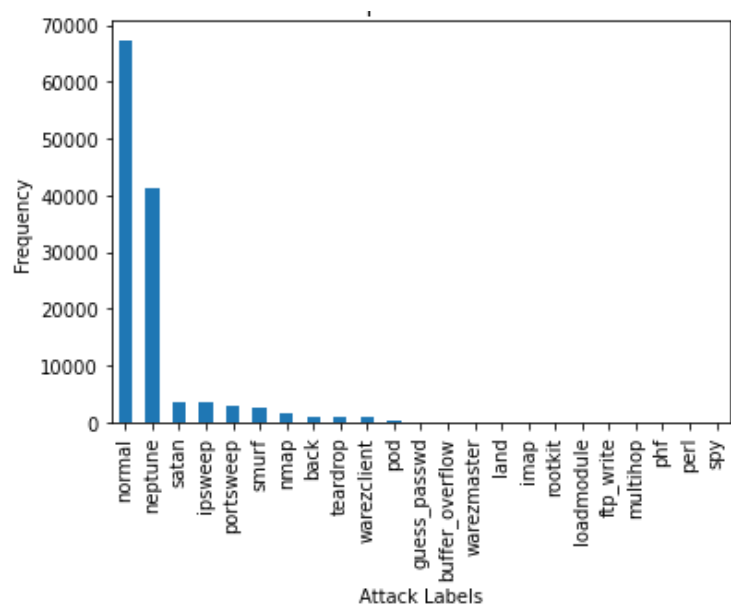


Fig 4- Number of samples for each attack

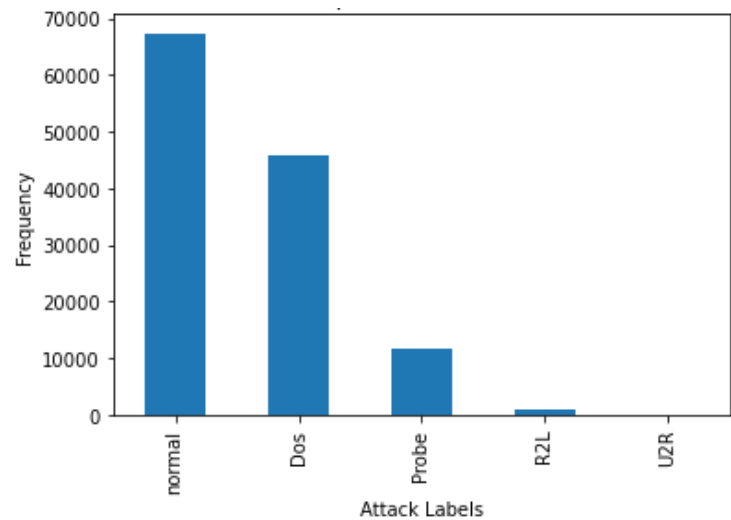


Fig 5: Number of samples for each attack class

3.1.4 Types of Computer Attacks

Cyber-attacks encompass a variety of strategies and objectives, leading to their classification into distinct categories based on the activities and targets of the attacker.

Each attack type can be classified into one of the following four classes:

Denial-of-Service (DoS) Attacks: These malicious activities are aimed at disrupting or restricting services provided by a network or computer system. Attackers overwhelm the target with a flood of network traffic or requests, rendering it inaccessible to legitimate users.

Probing Attacks: Probing attacks are focused on gathering information about the target network or computer system. Cybercriminals employ reconnaissance techniques to identify vulnerabilities, potential entry points, and system configurations for subsequent exploitation.

User-to-Root (U2R) Attacks: In U2R attacks, non-privileged users attempt to escalate their privileges to gain root or admin-level access on a specific computer or system. These attackers leverage vulnerabilities or weaknesses in the system to elevate their privileges and gain control over critical resources.

Remote-to-Local (R2L) Attacks: R2L attacks involve sending packets or malicious payloads to the victim machine from a remote location. The attacker's objective is to exploit vulnerabilities in the target system to gain unauthorized access, typically starting from a non-local or external position.

Within these broad categories, there are many different forms of computer attacks. A summary of these attacks with a brief explanation, characteristics, and examples are presented in Table 4:

Class	Subclass	Description
DOS	udpstrom, pod, neptune, land, smurf, drop, tear, snmpgetattack, apache2, back, mailbomb, processtable	denial-of-service, e.g., syn flood:
R2L	warezclient, warezmater, multihop, phf, imap, named, sendmail, xlock, xsnoop, worm, ftp_write, gues_passwd, snm, pgues	unauthorized access from a remote machine, e.g., guessing password;
U2R	xterm, sqlattack, httptunnel, ps, buffer_overflow, perl, root, kit, loadmodule	unauthorized access to local superuser (root) privileges, e.g., "buffer overflow" attack;
Probe	satan, saint, mscan, ipsweep, nmap, portsweep	surveillance and other probing, e.g., port scanning
Normal	Normal	benign record

Table 4: Different types of attacks [1]

3.2 Data Preprocessing

3.2.1 Data Normalization

Data normalization is a preprocessing technique used to rescale numerical features in a dataset to a standard range, typically between 0 and 1 or with a mean of 0 and a standard deviation of 1. It involves adjusting the values of features so that they fall within a similar scale, preventing features with larger scales from dominating the analysis or model training process.

The normalization process typically involves two main steps:

Centering: Subtracting the mean value of the feature from each data point. This centers the data around zero, ensuring that the mean of the normalized feature is zero.

Scaling: Dividing each centered data point by a measure of scale, such as the standard deviation or the range of the feature values. This scales the data to a standard range, ensuring that the variance of the normalized feature is consistent across different features.

duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_same_srv_rate	dst_host_diff_srv_rate	dst_host_same_src_port_rate	dst_host_diff_src_port_rate
0	0	top	ftp_data	SF	491	0	0	0	0	0	...	0.17	0.03	0.17
1	0	udp	other	SF	148	0	0	0	0	0	...	0.00	0.60	0.88
2	0	top	private	SO	0	0	0	0	0	0	...	0.10	0.05	0.00
3	0	top	http	SF	232	8153	0	0	0	0	...	1.00	0.00	0.03
4	0	top	http	SF	199	420	0	0	0	0	...	1.00	0.00	0.00
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
125968	0	top	private	SO	0	0	0	0	0	0	...	0.10	0.06	0.00
125969	8	udp	private	SF	105	145	0	0	0	0	...	0.98	0.01	0.01
125970	0	top	smtp	SF	2231	384	0	0	0	0	...	0.12	0.06	0.00
125971	0	top	klogin	SO	0	0	0	0	0	0	...	0.03	0.05	0.00
125972	0	top	ftp_data	SF	151	0	0	0	0	0	...	0.30	0.03	0.30

125973 rows × 43 columns

Fig 6: Before data normalization

duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	dst_host_same_srv_rate	dst_host_diff_srv_rate	dst_host_same_src_port_rate
0	-0.110249	top	ftp_data	SF	-0.007879	-0.004919	-0.014089	-0.089486	-0.007736	-0.095078	...	-0.818890	-0.782367	-0.280282
1	-0.110249	udp	other	SF	-0.007737	-0.004919	-0.014089	-0.089486	-0.007736	-0.095078	...	-1.035688	-1.161030	2.736852
2	-0.110249	top	private	SO	-0.007782	-0.004919	-0.014089	-0.089486	-0.007736	-0.095078	...	-0.809857	-0.938287	-0.174417
3	-0.110249	top	http	SF	-0.007723	-0.002891	-0.014089	-0.089486	-0.007736	-0.095078	...	1.258754	1.066401	-0.439078
4	-0.110249	top	http	SF	-0.007728	-0.004814	-0.014089	-0.089486	-0.007736	-0.095078	...	1.258754	1.066401	-0.439078

Fig 7: After data normalization

3.3 Classification

Under classification, there are two types: binary and multi-class. Binary splits the data into two groups. Multi-class means sorting things into more than two groups. For model selection and training, we create a dataset containing only numerical features and binary labels for intrusion detection. Then, we split it into two parts: one for training (70%) and one for testing (30%). We use different models to train dataset.

3.3.1 Binary Classification of Train Data

In the data preprocessing stage, the attack labels in the dataset are first transformed into a binary classification: 'normal' for network traffic and 'abnormal' for malicious traffic. This is achieved by mapping the original labels to these two categories.

Next, the binary labels are encoded numerically using label encoding, where 'normal' is represented as 1 and 'abnormal' as 0. This numerical representation is added as a new column named 'intrusion'.

To further facilitate the use of categorical data in machine learning models. This encoding creates separate binary columns for each category, allowing the model to process these categorical features effectively. The final dataset includes the original features, the columns ('normal' and 'abnormal'), the original 'label' column with binary categories, and the numerical 'intrusion' column. This comprehensive preprocessing ensures the data is ready for training robust and accurate intrusion detection models.

3.3.2 Multi-Class Classification of Train Data

In the data preprocessing stage, the attack labels in the dataset are transformed into multi-class categories: 'Dos', 'Probe', 'R2L', 'U2R', and 'normal'. This is achieved by copying the original data and extracting the 'label' column into a separate DataFrame.

Next, the multi-class labels are encoded numerically using label encoding, where each unique category is assigned a unique integer value. For instance, 'Dos' is encoded as 0, 'Probe' as 1, 'R2L' as 2, and 'U2R' as 3 and 'normal' as 4. This numerical representation is added as a new column named 'intrusion' in the DataFrame.

The final dataset includes the original features, and the numerical 'intrusion' column. This comprehensive preprocessing ensures the data is ready for training robust and accurate intrusion detection models that can distinguish between multiple types of attacks and normal network traffic. We then split the data into training and testing sets, ensuring a 70-30 split to balance model training and evaluation.

3.3.3 Model Selection and Model Training

We aim to create a dataset that includes only numeric features along with binary labels for intrusion detection.

First, we select all the numeric columns from the original training data to ensure that the dataset contains only numerical attributes. This selection is crucial for certain machine learning algorithms that require numeric input data.

- **Binary Classification**

In our dataset we add an 'intrusion' column to this numeric dataset. This column contains the encoded binary labels, which were previously generated. These labels represent different types of network traffic, with 'normal' traffic encoded as 1 and 'abnormal' traffic encoded as 0. The inclusion of these labels facilitates binary classification tasks, allowing the model to distinguish between normal and malicious network activities.

we preprocess the dataset, split it into training and testing sets, and train model for binary classification. we define our feature set (X) and label set (Y).

The feature set includes all relevant attributes, while the label set contains binary labels indicating 'normal' (1) or 'abnormal' (0). We then split the data into training and testing sets, ensuring a 70-30 split to balance model training.

- **Multi Class Classification**

We creates a dataset that includes only numeric features and multi-class labels, useful for feature selection. For the binary class dataset, we isolate only the numeric columns and add an encoded intrusion label. These multi-class labels in the intrusion column are represented by the following codes: 'Dos' = 0, 'Probe' = 1, 'R2L' = 2, 'U2R' = 3, and 'normal' = 4. This dataset is useful for categorizing the model into normal and different malicious attack class.

To prepare for model training, we define the feature matrix X by selecting all relevant columns except the target attribute. The target variable Y is the intrusion label. We then split the dataset into training and testing sets, allocating 75% of the data for training and 25% for testing.

3.3.4 Description of Different Models

- **Linear SVM**

An SVM model with a linear kernel is chosen for its effectiveness in handling binary classification problems. The model is trained on the training data and evaluated on the test data. The accuracy score and classification report provide detailed metrics on the model's performance, including its ability to correctly classify normal and abnormal network traffic.

- **kNN**

The kNN algorithm is a simple, yet powerful supervised learning model used for both classification and regression tasks.

The kNN model predicts the class of a given data point by examining the 'k' closest points (neighbors) in the training dataset. For classification, the data point is assigned the most common class among its kNN. For regression, the prediction is typically the average (or sometimes the median) of the values of its k nearest neighbors.

- **RF**

The RF algorithm is a smart method for solving classification and regression problems. It works by creating many decision trees during training, where each tree learns from a different part of the data and considers only some of the features at each decision. When it comes to making predictions, RF combines the results from all these trees to give a final prediction. This approach helps improve accuracy and reduces the risk of making mistakes.

- **XGBoost**

XGBoost is a powerful machine learning algorithm. It is an optimized implementation of gradient boosting trees designed for speed and performance. XGBoost works by sequentially building a series of decision trees, where each subsequent tree corrects the errors made by the previous ones. This iterative process results in a highly accurate and robust predictive model.

Chapter 4

Experimental Result

The experimental results include a parameter table that compares the performance of different models: SVM, kNN, RF, and XGBoost. Tables show the accuracy of each model with various combinations of attributes.

4.1 Parameter Table

A parameter table in machine learning is a structured representation of the hyperparameters used in a model and the resulting performance metrics for each combination of these hyperparameters. It helps in comparing different configurations to find the optimal settings for the model.

4.1.1 Parameter Table of SVM

Kernel: This column shows the type of kernel used in the SVM model (either 'linear' or 'rbf').

Gamma: This column indicates the gamma value used. 'auto' lets the SVM automatically select the value, while specific numeric values (like 0.1, 0.01) are provided manually.

Accuracy: This column displays the accuracy score of the SVM model with the corresponding kernel and gamma settings.

By analyzing these Table 5 and Table 6, we can identify which combination of kernel and gamma values yields the highest accuracy, helps us choosing the best parameters for our SVM model.

Kernel	Gamma	Accuracy
Linear	auto	0.935924
Linear	0.1	0.935924
Linear	0.01	0.935924
Rbf	Auto	0.977075
Rbf	0.1	0.978790
Rbf	0.01	0.969423

Table 5: Different Accuracy for Different Parameter value of SVM model for Binary Classifier

Kernel	Gamma	Accuracy
Linear	auto	0.935035
Linear	0.1	0.935035
Linear	0.01	0.935035
Rbf	Auto	0.975646
Rbf	0.1	0.978126
Rbf	0.01	0.968597

Table 6: Different Accuracy for Different Parameter value of SVM model for Multi-Class Classifier

4.1.2 Parameter Table of kNN

n_neighbors: This column shows the number of neighbors considered in the KNN model.

Metric: This column indicates the distance metric used to compute the nearest neighbors.

Accuracy: This column displays the accuracy score of the KNN model for each combination of n_neighbors and metric.

By analyzing the output Table 7 and Table 8, one can determine which combination of n_neighbors and metric results in the highest accuracy, helping to select the optimal parameters for the KNN classifier.

N_neighbors	Metric	Accuracy
3	euclidean	0.990570
3	minkowski	0.990570
3	manhattan	0.990824
4	euclidean	0.989077
4	minkowski	0.989077
4	manhattan	0.989712

Table 7: Different Accuracy for Different Parameter value of kNN model for Binary Classifier

N_neighbors	Metric	Accuracy
3	euclidean	0.989427
3	minkowski	0.989427
3	manhattan	0.989839
4	euclidean	0.987839
4	minkowski	0.987839
4	manhattan	0.988728

Table 8: Different Accuracy for Different Parameter value of kNN model for Multi-Class Classifier

4.1.3 Parameter Table of RF

n_estimators: This column shows the number of trees in the RF .

max_depth: This column indicates the maximum depth of the trees. A value of None means there is no maximum depth.

Accuracy: This column displays the accuracy score of the RF model for each combination of n_estimators and max_depth.

By analyzing the output Table 9 and Table 10 , one can determine which combination of n_estimators and max_depth results in the highest accuracy, helping to select the optimal parameters for the RF classifier. This allows for fine-tuning the model to achieve the best possible performance on the given dataset.

N_estimators	Max_depth	Accuracy
50	10.0	0.995269
50	20.0	0.997809
50	NaN	0.997841
100	10.0	0.995332
100	20.0	0.997936
100	NaN	0.997968

Table 9: Different Accuracy for Different Parameter value of RF model for Binary Classifier

N_estimators	Max_depth	Accuracy
50	10.0	0.994348
50	20.0	0.997682
50	NaN	0.997777
100	10.0	0.994475
100	20.0	0.997777
100	NaN	0.997714

Table 10: Different Accuracy for Different Parameter value of RF model for Multi-Class Classifier

4.1.4 Parameter Table of XGBoost

Max Depth: This column shows the maximum depth of the trees in the XGBoost model.

Learning Rate: This column indicates the learning rate, which controls the step size at each iteration while moving toward a minimum of the loss function.

N Estimators: This column displays the number of trees in the ensemble.

Accuracy: This column presents the accuracy score of the XGBoost model for each combination of max_depth, learning_rate, and n_estimators.

By analyzing the output Table 11 and Table 12, one can identify which combination of max_depth, learning_rate, and n_estimators yields the highest accuracy, aiding in the selection of the optimal parameters for the XGBoost classifier. This enables fine-tuning of the model to achieve the best possible performance on the given dataset.

Max Depth	Learning Rate	N Estimators	Accuracy
3	0.1	50	0.987204
3	0.1	100	0.992507
3	0.1	150	0.994570
3	0.2	50	0.992570
3	0.2	100	0.996126
3	0.2	150	0.997047
3	0.4	50	0.996349
3	0.4	100	0.997365
3	0.4	150	0.997809

Table 11: Different Accuracy for Different Parameter value of XGBoost model for Binary Classifier

Max Depth	Learning Rate	N Estimators	Accuracy
3	0.1	50	0.985235
3	0.1	100	0.995809
3	0.1	150	0.996793
3	0.2	50	0.995332
3	0.2	100	0.997269
3	0.2	150	0.997682
3	0.4	50	0.997428
3	0.4	100	0.997714
3	0.4	150	0.998063

Table 12: Different Accuracy for Different Parameter value of XGBoost model for Multi-Class Classifier

4.2 Accuracy with Combination of Various Attributes

The table represent's the accuracy achieved by model with varying combinations of attributes, showcasing the importance of each attribute in enhancing overall accuracy. As we add one attribute at a time, we notice that the accuracy of the models improves. Each attribute plays a crucial role for improving the accuracy.

S.No	Attributes	Accuracy
1	'src_bytes'	53
2	'src_bytes', 'dst_bytes'	53.2641138
3	'src_bytes','dst_bytes', 'same_srv_rate'	86.40693465
4	'src_bytes','dst_bytes','same_srv_rate', 'dst_host_same_srv_rate'	86.40693465
5	'src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count'	86.40693465
6	'src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in'	89.49641201
7	'src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in', 'diff_srv_rate'	89.54404013
8	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate', 'srv_error_rate'	89.55991617
9	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate', 'error_rate'	89.55356576
10	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate', 'error_rate', 'count'	90.54105544
11	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate','error_rate','count', 'dst_host_same_src_port_rate'	92.76687623
12	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate','error_rate','count', 'dst_host_same_src_port_rate','dst_host_diff_srv_rate'	92.91928621
13	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate', 'error_rate','count', 'dst_host_same_src_port_rate','dst_host_diff_srv_rate', 'dst_host_srv_diff_host_rate'	93.30666159
14	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate', 'error_rate','count', 'dst_host_same_src_port_rate','dst_host_diff_srv_rate','dst_host_srv_diff_host_rate', 'srv_count',	93.42414428
15	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate','error_rate','count', 'dst_host_same_src_port_rate','dst_host_diff_srv_rate','dst_host_srv_diff_host_rate', 'srv_count', 'dst_host_error_rate'	93.34158887
16	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate','error_rate','count', 'dst_host_same_src_port_rate','dst_host_diff_srv_rate','dst_host_srv_diff_host_rate', 'srv_count', 'dst_host_error_rate', 'dst_host_count'	93.52575094
17	src_bytes','dst_bytes','same_srv_rate','dst_host_same_srv_rate','dst_host_srv_count', 'logged_in','diff_srv_rate','srv_error_rate','error_rate','count', 'dst_host_same_src_port_rate','dst_host_diff_srv_rate','dst_host_srv_diff_host_rate', 'srv_count', 'dst_host_error_rate', 'dst_host_count', 'dst_host_srv_error_rate'	93.5924303

Table 13: Accuracy Of Combination Of Different Attributes

Chapter 5

Experimental Setup

In experiment setup, confusion matrix is used to see how well each model classified attacks. Additionally, tables to show the accuracy, mean absolute error and values of each model, which helped us understand the performance of models. And, graphs to visually represent how each model performed. This allowed us to compare the models easily and see which one worked best for detecting intrusions.

5.1 Confusion Matrix

Explanation of the Confusion Matrix

- True Positive (TP): The model correctly predicts the positive class (Intrusion).
- True Negative (TN): The model correctly predicts the negative class (Not Intrusion).
- False Positive (FP): The model incorrectly predicts the positive class (Intrusion) when it is actually negative (Not Intrusion).
- False Negative (FN): The model incorrectly predicts the negative class (Not Intrusion) when it is actually positive (Intrusion).

Each row of the matrix represents the actual class, while each column represents the predicted class. The diagonal elements of the matrix indicate the number of correct predictions for each class, while the off-diagonal elements show the misclassifications.

The confusion matrix is a 2×2 matrix for binary classification and helps in assessing the accuracy, precision, recall, and overall performance of the classification model.

The confusion matrix is also helpful for multi-class classification, showcasing the performance of the classification model across multiple classes. After training model, predictions were made on the test dataset, and the confusion matrix was computed. This multi-class confusion matrix has a 5×5 dimension, reflecting the five distinct classes involved in the classification: 'Dos', 'Probe', 'R2L', 'U2R', and 'normal'.

By visualizing this matrix using a heatmap, we can quickly identify how many predictions were correct and how many were incorrect, providing insight into the model's strengths and areas for improvement.

5.1.1 Evaluating Trained SVM Using Confusion Matrix

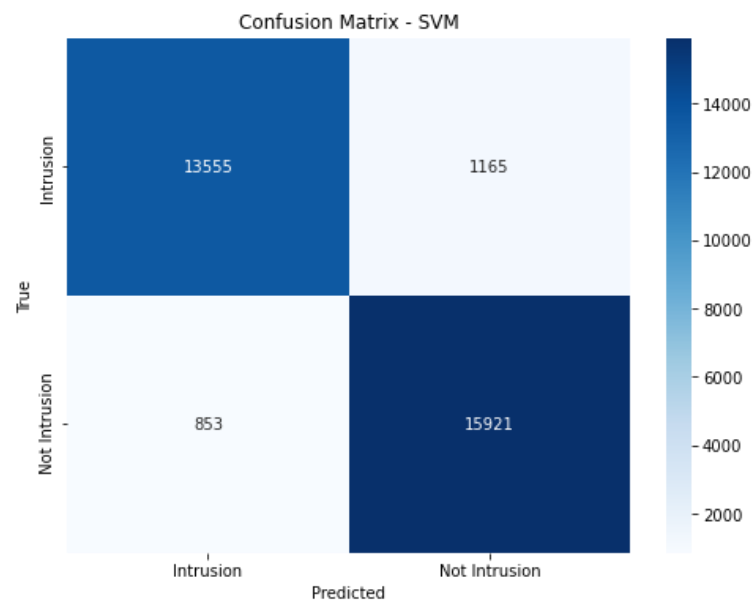


Fig 8: Confusion Matrix – SVM for Binary Classification

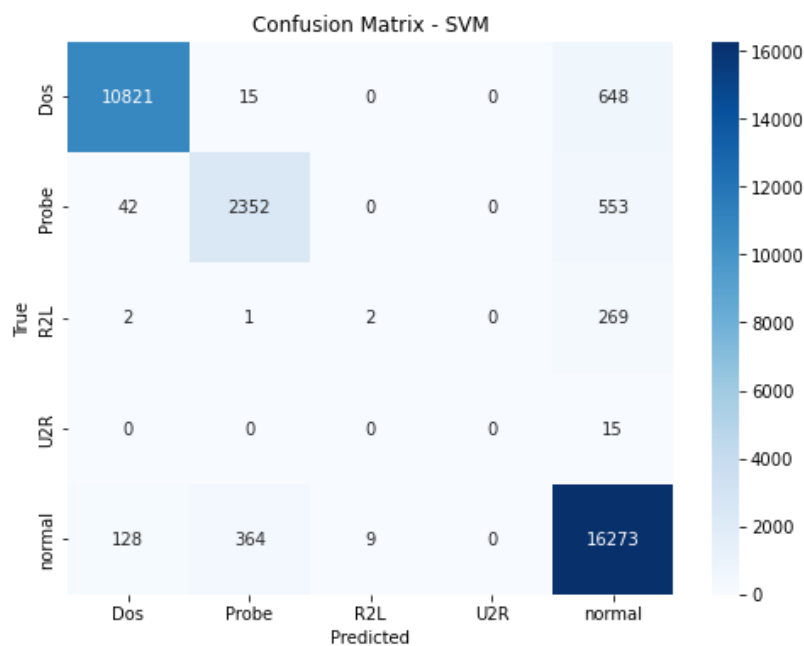


Fig 9: Confusion Matrix – SVM for Multi-Class Classification

This process of computing and visualizing the confusion matrix is crucial for evaluating the performance of the SVM model as shown in Fig 8 for Binary Classification and in Fig 9 for Multi-Class Classification. It provides a clear and concise way to understand how well the model distinguishes between the different classes and highlights any potential issues with misclassification.

5.1.2 Evaluating Trained kNN Model Using Confusion Matrix

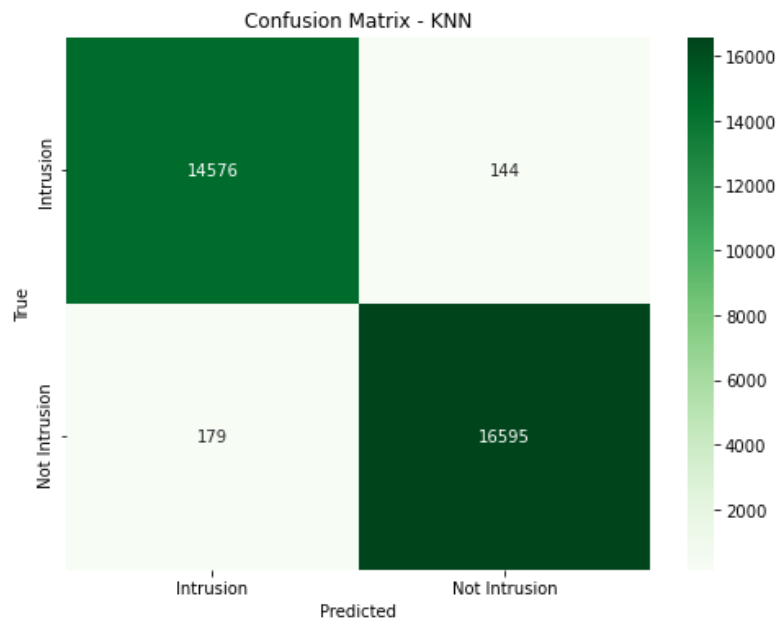


Fig 10: Confusion Matrix – kNN for Binary Classification

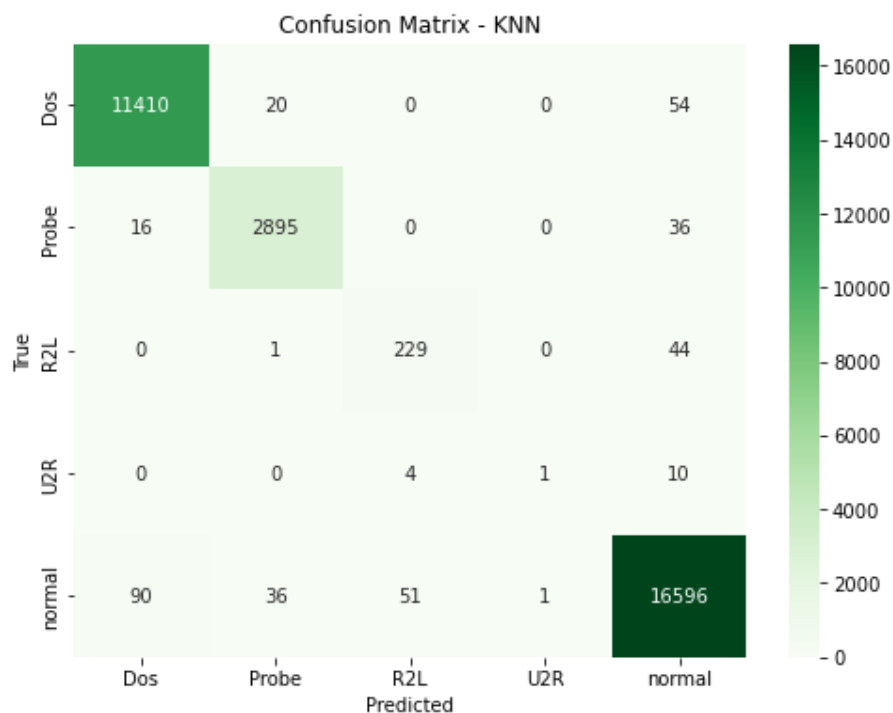


Fig 11: Confusion Matrix – kNN for Multi-Class Classification

The process of calculating and displaying the confusion matrix is essential for assessing the KNN model's performance as shown in Fig 10 for Binary Classification and in Fig 11 for Multi-Class Classification.. It offers a straightforward and clear method to comprehend how effectively the model differentiates between various classes and identifies any possible misclassification problems.

5.1.3 Evaluating Trained RF Model Using Confusion Matrix

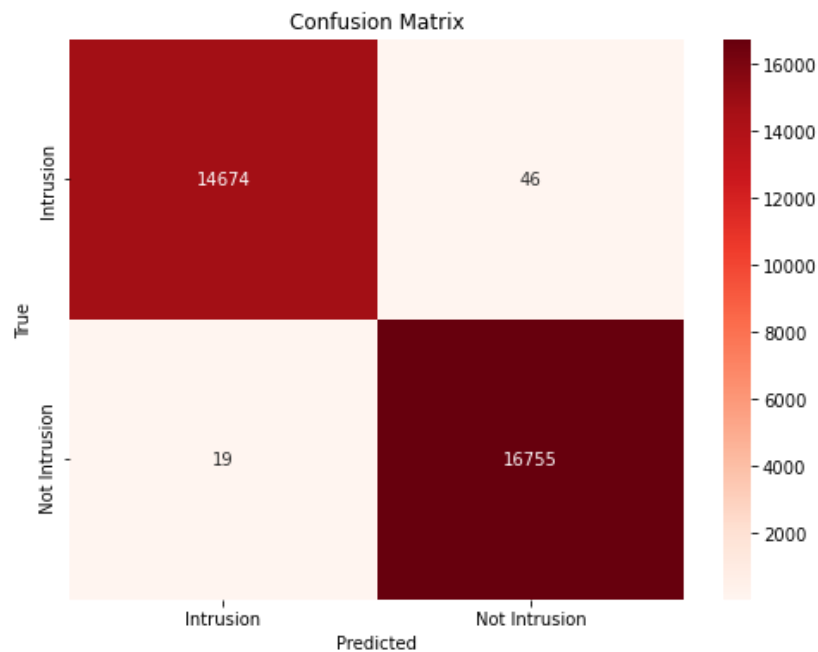


Fig 12: Confusion Matrix – RF for Binary Classification

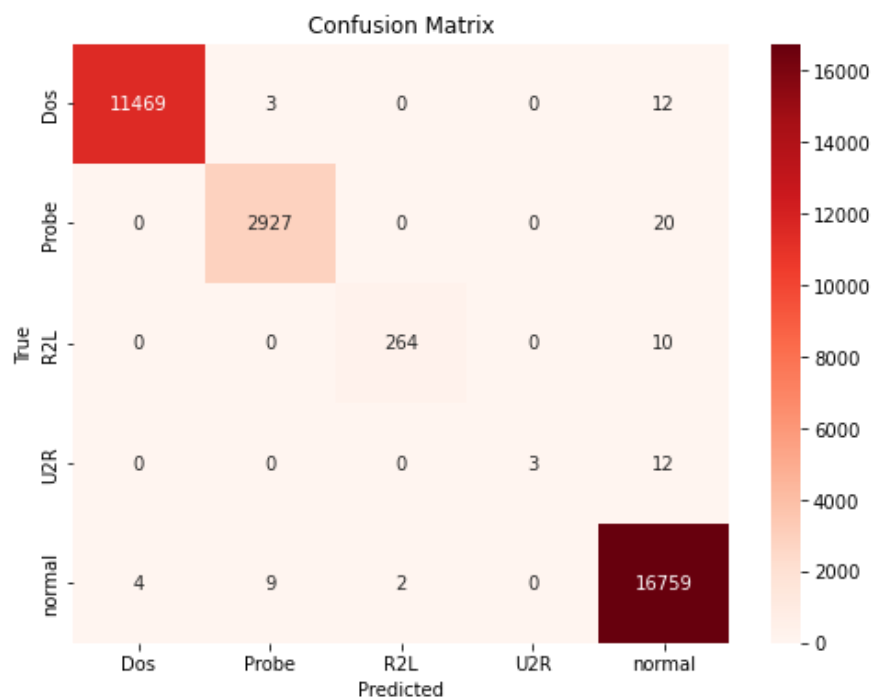


Fig 13: Confusion Matrix – RF for Multi-Class Classification

Calculating and presenting the confusion matrix is vital for evaluating the performance of the RF model as shown in Fig 12 for Binary Classification and in Fig 13 for Multi-Class Classification. It provides an easy-to-understand way to see how well the model distinguishes between different classes and highlights any potential misclassification issues.

5.1.4 Evaluating Trained XGBoost Model Using Confusion Matrix

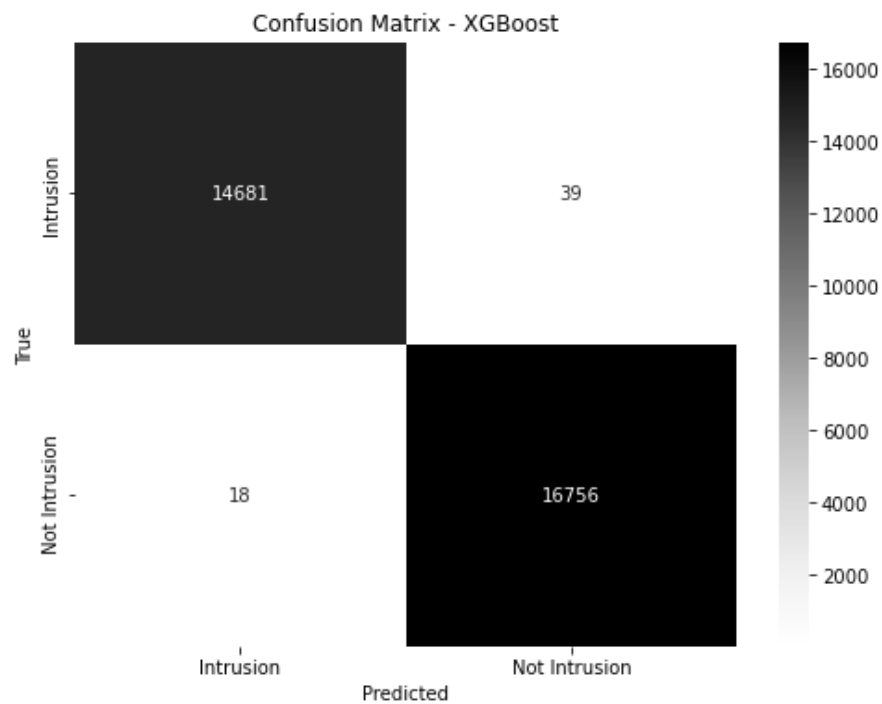


Fig 14: Confusion Matrix – XGBoost for Binary Classification

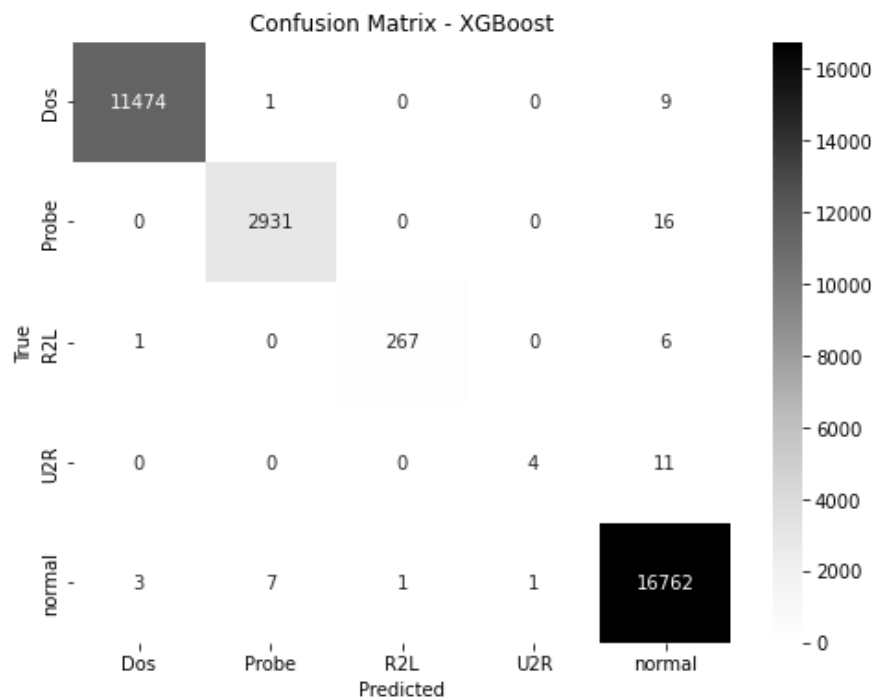


Fig 15: Confusion Matrix – XGBoost for Multi-Class Classification

For XGBoost model evaluation, computing and presenting the confusion matrix is essential. This matrix offers a clear and intuitive representation of the model's performance by showcasing its ability to differentiate between various classes and identifying any potential misclassification challenges as shown in Fig 14 for Binary Classification and in Fig 15 for Multi-Class Classification.

5.2 Accuracy of Different Models

In the assessment of intrusion detection algorithms, namely SVM, KNN, and RF, XGBoost it was found that the RF model yielded the most accurate results. Among these algorithms, RF consistently demonstrates higher accuracy rates compared to SVM. This finding underscores the effectiveness of RF as a robust and reliable method for intrusion detection tasks.

Models	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error	R2 Score	Accuracy
SVM	0.06407569695215075	0.06407569695215075	0.25313177785128194	74.29966180838876	93.59243030418493
kNN	0.010255921762875469	0.010255921762875469	0.1012715249360622	95.88060350581387	98.97440782371245
RF	4.2337450650409084e-05	4.2337450650409084e-05	0.006506723495770286	99.98298047337671	99.99576625493496
XGboost	0.00013759671461382954	0.00013759671461382954	0.011730162599633031	99.94468653847433	99.98624032853863

Table 14: Final result of all models for Binary Classification

Models	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error	R2 Score	Accuracy
SVM	0.20600749349082365	0.694163967739887	0.8331650303150553	81.12182270349892	93.5035244808535
kNN	0.03283165047310599	0.10744903791198324	0.3277942005465979	97.03576168941206	98.84739950466755
RF	0.0001481810772764318	0.0005292181331301136	0.023004741535825036	99.98541886869351	99.99576625493496
XGboost	0.00023285597857724996	0.0008044115623577726	0.028362150171624377	99.97783668041414	99.99259094613619

Table 15: Final result of all models for Multi-Class Classification

5.3 Visual Representation of Different Models

These figures below display the accuracy of various models, the results indicate that RF is the most accurate, followed by XGBoost, kNN, and SVM. RF consistently outperforms the other models, demonstrating its robustness and reliability for intrusion detection tasks.

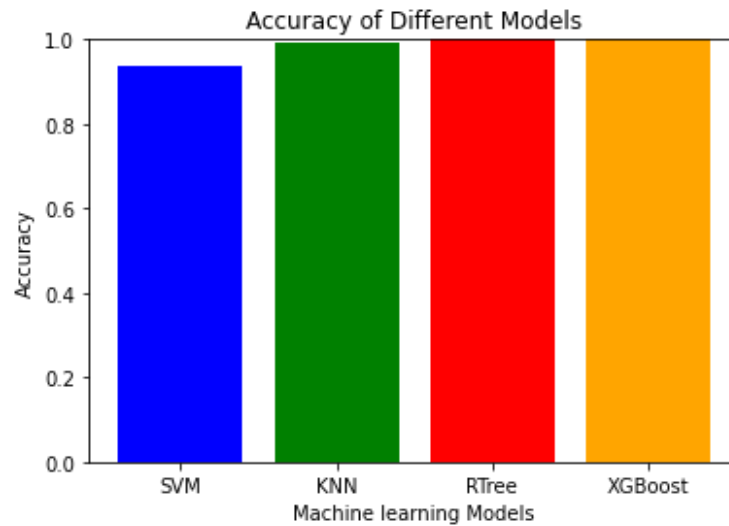


Fig 16: Accuracy of Different Models for Binary Classification

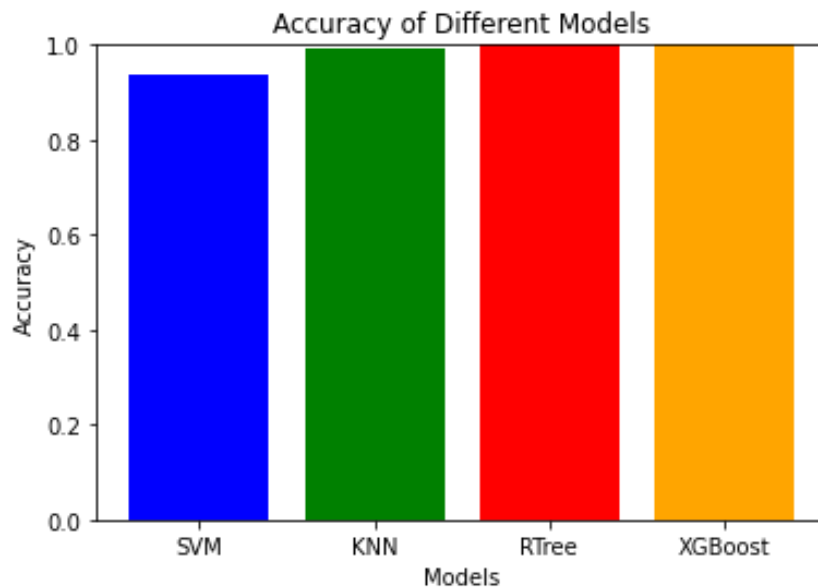


Fig 17: Accuracy of Different Models for Multi-Class Classification

Chapter 6

Conclusion

Cybercriminals are evolving their techniques, using advanced methods and social engineering tactics to target computer users, making it hard for IDS to work effectively. The study examines various IDS methods and machine learning techniques used to detect attacks. It finds that using outdated information can cause many false alarms and miss real threats. The study also points out that the public datasets used in research are limited and don't reflect current malware. Additionally, attackers use clever techniques to evade IDS, posing a big challenge for developing better IDS systems. More advanced IDS and up-to-date datasets are needed to keep up with these evolving threats.

In the future, IDS will become more advanced in identifying strange behaviors and threats by using machine learning and anomaly-based detection. It will analyze behavior and big data more quickly to catch cyberattacks early. And, also become better at safeguarding Internet-connected devices and automatically responding to threats. Overall, IDS will keep evolving to stay ahead of cyber threats and protect our digital world.

REFERENCES

- [1] Khraisat, Ansam and Gondal, Iqbal and Vamplew, Peter and Kamruzzaman, Joarder, Survey of intrusion detection system: techniques, datasets and challenges, Cybersecurity, vol-2, num-1, pg-1-22, 2019, publ-Springer
- [2] <https://bunny.net/academy/security/what-is-network-intrusion-detection-nids/> _last visited on:21-05-2024
- [3] Wanda, Putra and Jie, Huang J, A survey of intrusion detection system, International Journal of Informatics and Computation, vol-1, pg-1-10, 2020
- [4] Lazarevic, Aleksandar, Vipin Kumar, and Jaideep Srivastava, "Intrusion detection: A survey", Managing Cyber Threats, Springer US, 2005, Pp. 19-78, 2005.
- [5] P. Ghosh, A.K. Mandal and R. Kumar, "An Efficient Cloud Network Intrusion Detection System," Chapter Information System Design and Intelligent Applications, vol. 339 of the series Advances in Intelligent System and Computing, 2015, pp 91-99.
- [6] O. Can and O. K. Sahingoz, "A survey of intrusion detection system in wireless sensor networks," in 2015 6th international conference on modeling, simulation, and applied optimization (ICMSAO), 2015, pp. 1–6
- [7] Creech G, Hu J (2014a) A semantic approach to host-based intrusion detection system using Contiguous and Discontiguous system call patterns. IEEE Trans Comput 63(4):807–819
- [8] Kenkre PS, Pai A, Colaco L (2015b) Real Time Intrusion Detection and Prevention System. Springer International Publishing, Cham, pp 405–411
- [9] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," J Netw Comput Appl, vol. 36, no. 1, pp. 42–57, 2013/01/01/ 2013
- [10] C. R. Meiners, J. Patel, E. Norige, E. Torng, and A. X. Liu, "Fast regular expression matching using small TCAMs for network intrusion detection and prevention system," presented at the Proceedings of the 19th USENIX conference on security, Washington, DC, 2010
- [11] T. H. Ptacek and T. N. Newsham, "Insertion, evasion, and denial of service: eluding network intrusion detection," DTIC Document 1998
- [12] Sadotra P, Sharma C (2016) A survey: intelligent intrusion detection system in computer security. Int J Comput Appl 151(3):18–22
- [13] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in ICISSP, 2018, pp. 108–116
- [14] Vigna G, Kemmerer RA (1999) NetSTAT: a network-based intrusion detection system. J Computer Security 7:37–72
- [15] D. Wagner and P. Soto, "Mimicry attacks on host-based intrusion detection system," presented at the Proceedings of the 9th ACM conference on computer and communications security, Washington, DC, USA, 2002
- [16] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection system," Computers & Security, vol. 24, no. 4, pp. 295–307, 6// 2005
- [17] G. Creech, "Developing a high-accuracy cross platform host-based intrusion detection system capable of reliably detecting zero-day attacks," University of New South Wales, Canberra, Australia, 2014
- [18] Axelsson, Stefan, "Intrusion detection system: a survey and taxonomy," technical report 2000, publ- Citeseer