

PROJECT REPORT ON

**NETWORK INTRUSION DETECTION SYSTEM  
USING STATISTICAL CLASSIFICATION AND AI  
ALGORITHMS**

Submitted by

<b>Devashree Chandrashekhar Limaye</b>	<b>B120203113</b>
<b>Sagarika Satish Limaye</b>	<b>B120203114</b>
<b>Ankita Sanjay Moholkar</b>	<b>B120203128</b>

in partial fulfillment for the award of degree of

Bachelor of Engineering in

**ELECTRONICS AND TELECOMMUNICATION of**

**SAVITRIBAI PHULE PUNE UNIVERSITY**

Under the Guidance of

**Dr. Mrudul Dixit**

Sponsored by

Self



**MKSSS's CUMMINS COLLEGE OF ENGINEERING FOR  
WOMEN, KARVENAGAR, PUNE - 411052**

2017-18

- **Project Name: Network Intrusion Detection System using Statistical Classification and AI algorithms**
- **Subject Area: Networking**
- **Nature of the Project: Software**

# CERTIFICATE

This is to certify that

<b>Devashree Chandrashekhar Limaye</b>	<b>B120203113</b>
<b>Sagarika Satish Limaye</b>	<b>B120203114</b>
<b>Ankita Sanjay Moholkar</b>	<b>B120203128</b>

have successfully completed the work on their Project Topic

**NETWORK INTRUSION DETECTION SYSTEM USING STATISTICAL  
CLASSIFICATION AND AI ALGORITHMS**

In partial fulfillment for the award of the degree of

**BACHELOR OF ENGINEERING  
IN ELECTRONICS AND TELECOMMUNICATION  
OF SAVITRIBAI PHULE PUNE UNIVERSITY**

In

**CUMMINS COLLEGE OF ENGINEERING FOR WOMEN,  
KARVENAGAR, PUNE-52.**

Internal Guide  
Dr. Mrudul Dixit

Head of the Department  
Dr. Prachi Mukherji

Director  
Dr. M. B. Khambete

## **Acknowledgement.**

We take this opportunity to express our deep sense of gratitude towards our esteemed guide Dr. Mrudul Dixit for giving us this opportunity to select and present this project and also providing advices for successful completion. We consider ourselves fortunate to work under her guidance.

We thank Dr. Prachi Mukherji, Head, Electronics and Telecommunication Department, for her guidance, support and valuable suggestions throughout our project work.

We also thank all the teaching and non-teaching staff members of Electronics and Telecommunications Department for their help whenever required. Finally, we express our gratitude to all those who helped us directly or indirectly in completion of this project.

# **Abstract**

In today's world, due to increased usage and easy access to the internet the ways to attack a network system have also increased widely. Network security has become more prominent due to popularization of information technology. As a consequence of this, development of a faster and efficient Network Intrusion Detection System is required. With an exponential growth of network bandwidths this task demanded improvements in speed and accuracy. Hence port-based packet classification and deep packet inspection methods for intrusion detection systems were developed. These methods faced issues like low detection rate, high false positive rates.

In our project, using statistical packet classification and Artificial Intelligence we have designed a more accurate and efficient, having a faster response time Network Intrusion Detection and Prevention System. In this project, real-time data from the network is captured in Wireshark and is used for the analysis and implementation of the system is done using two algorithms Naïve Bayes and Support Vector Machines for classification.

# INDEX

<b>Sr. No.</b>	<b>Title</b>	<b>Page No.</b>
<b>1.</b>	<b>Introduction</b> .....	<b>10</b>
<b>2.</b>	<b>Literature Survey</b> .....	<b>13</b>
2.1	What is Packet Classification .....	14
2.2	Traditional Methods of Packet Classification .....	18
2.3	Statistical Classification .....	19
<b>3.</b>	<b>Network Intrusion detection and prevention system</b> .....	<b>22</b>
<b>4.</b>	<b>Distributed Denial Of Service Attacks</b> .....	<b>27</b>
<b>5.</b>	<b>Design Specifications</b> .....	<b>34</b>
<b>6.</b>	<b>The Proposed System</b> .....	<b>36</b>
6.1	Source-Destination IP pair and number of packets approach .....	37
6.2	Inter-arrival Time-based approach .....	39
6.3	Block Diagram .....	41
6.4	Explanation .....	42

<b>7.</b>	<b>Algorithm for Implementation</b>	.....	<b>43</b>
<b>8.</b>	<b>Machine Learning</b>	.....	<b>47</b>
8.1	Support Vector Machine Classifier	.....	50
8.2	Naïve Bayes Classifier	.....	53
<b>9.</b>	<b>Flowchart of The Algorithm</b>	.....	<b>55</b>
<b>10.</b>	<b>Practical Implementation</b>	.....	<b>57</b>
10.1	Software Used	.....	60
<b>11.</b>	<b>Testing</b>	.....	<b>66</b>
<b>12.</b>	<b>Results and Conclusion</b>	.....	<b>70</b>
<b>13.</b>	<b>Future Scope</b>	.....	<b>73</b>
	<b>References</b>	.....	<b>75</b>

# LIST OF FIGURES

<b>Fig. No.</b>	<b>Title</b>	<b>Page.</b>
1	Routing .....	16
2	A network Intrusion Detection System .....	25
3	A DOS Attack scenario .....	29
4	A normal TCP connection .....	32
5	A TCP-SYN Flooding .....	33
6	DDOS Scenario .....	37
7	TCP Flooding .....	39
8	Bursts/Sec .....	40
9	Block Diagram of the System .....	41
10	SVM in our system .....	52
11	Wireshark window .....	61
12	HyeaneFE Packet Generator .....	63
13	Spyder Window For Python .....	65
14	Test Case 1 .....	67
15	Test Case 2 .....	68
16	Test Case 3 .....	69



# LIST OF TABLES

<b>Table No.</b>	<b>Title</b>	<b>Page No.</b>
1.	Source and Destination IP .....	38
2.	Results of the system .....	70

# **CHAPTER 1**

## **INTRODUCTION**

# 1.Introduction

Today, with unrestrained and ever-increasing number of electronic devices connected to the internet, attackers now have large surface for intrusion. Therefore, it is necessary to protect our systems from known and unknown intrusion attacks; for this there is a need for an advanced Intrusion Detection System. Traditional Intrusion Detection System (IDS) were done using the Port based classification methods. But there were certain disadvantages with the above stated method.

Problems:

- Some port numbers may be the dynamic ones that is not registered with IANA.
- Easy to cheat by changing the port number in the system.
- IP layer encryption may also obscure the TCP and UDP port numbers thus making it unable to identify the actual port numbers.
- It has significant complexities and processing on the traffic devices.
- Direct payload analysis may lead to violation of privacy policies.
- It is also difficult to maintain bit stream sequences having high hit ratios.

Traditional methods may provide high accuracy in the results as entire packet is analyzed and not just the header but the problem is that it consumes large amounts of memory for pattern matching in the network applications. This results into larger costs.

This approach fails in case of encrypted packets as the packet cannot be decrypted by the system and garbage value appears.

If there is a new virus in the system the network may not have the already stored signature and in that case the DPI method fails to analyze the incoming packet. Updating the sequences create a tedious and error prone task. Increasing traffic congestion may lead to develop latency by the DPI system.

Today, increasing demand for large network bandwidths leads to the growth in the data traffic. As a consequence of this packet classification is required for handling the drastically increasing the traffic on both the edge and the core devices.

The complexity of network applications is an ever-increasing factor. Previously the primary application where classification was used was firewalls for restricting unwanted traffic. In today's world, due to increasing usage of cloud services, large networks are formed which carry immense traffic. There is a need to develop a system to protect the large networks from malicious traffic and abnormalities but having a faster response time so that the system will not be damaged.

In this project, we have implemented a Network Intrusion Detection and Prevention system with a new approach of using statistical parameters of the flows in order to detect the anomalies in the network and protect the network from harmful attacks initiated by the hackers. This system is designed to provide a faster response time to the network attacks as compared to existing systems.

## **CHAPTER 2**

# **LITERATURE SURVEY**

## **2.Literature Survey**

### **2.1 A brief on Packet Classification**

Packet Classification is a process of classifying or processing internet packets for meeting certain objectives such as routing and filtering.

Access control, traffic engineering, intrusion detection, software defined networking and many other network services require the discrimination of packets based on the multiple fields of packet headers, which is called packet classification.

Packet classification features provide the capability to partition network traffic into multiple priority levels or classes of service.

The objective of the packet classification is to classify the internet packets by applying a set of predefined rules to the header fields. Each rule consists of some set of components in a range matching expression. A packet is said to be matched if and only if a desired rule matches the field in the header of the packet and satisfies the matching expression.

#### **Different types of solution for packet classification:**

Mathematical based: Consists of algorithms focusing more on the mathematical analysis of the packets. They are rarely seen implemented in the real-life networks though due to number of special conditions required to simplify the problems.

Observations based: These algorithms are based on the statistical characteristics observed in the rules for more optimization and efficient classification.

Hardware based: One more kind of algorithms is proposed to enhance packet classification based on application specific hardware. This kind of solutions are FPGA/ASIC based algorithms having extremely high performance. But they suffer from poor scalability and portability.

### **Applications of packet classification:**

**Routing:** The Internet is comprised of packet-processing nodes, called routers, that route packets towards their destinations, and physical links that transport packets from one router to another. Owing to advances in optical technologies, such as Wavelength Division Multiplexing, the data rates of links have increased rapidly over the years. However, routers have failed to keep up with this pace because they must perform expensive per-packet processing operations. Every router is required to perform a forwarding decision on an incoming packet to determine the packet's next-hop router. This is achieved by looking up the destination address of the incoming packet in a forwarding table. Besides increased packet arrival rates because of higher speed links, the complexity of the forwarding lookup mechanism and the large size of forwarding tables have made routing lookups a bottleneck in the routers that form the core of the Internet. Traditionally, the Internet provides only a "best-effort" service, treating all packets going to the same destination identically, and servicing them in a first come-first-served manner. However, Internet Service Providers are seeking ways to provide differentiated services (on the same network infrastructure) to different users based on their different requirements and expectations of quality from the Internet. For this, routers need to have the capability to distinguish and isolate traffic belonging to different flows. The ability to classify each incoming packet to determine the flow it belongs to is called packet classification and could be based on an arbitrary number of fields in the packet header.

### IP Routing Process

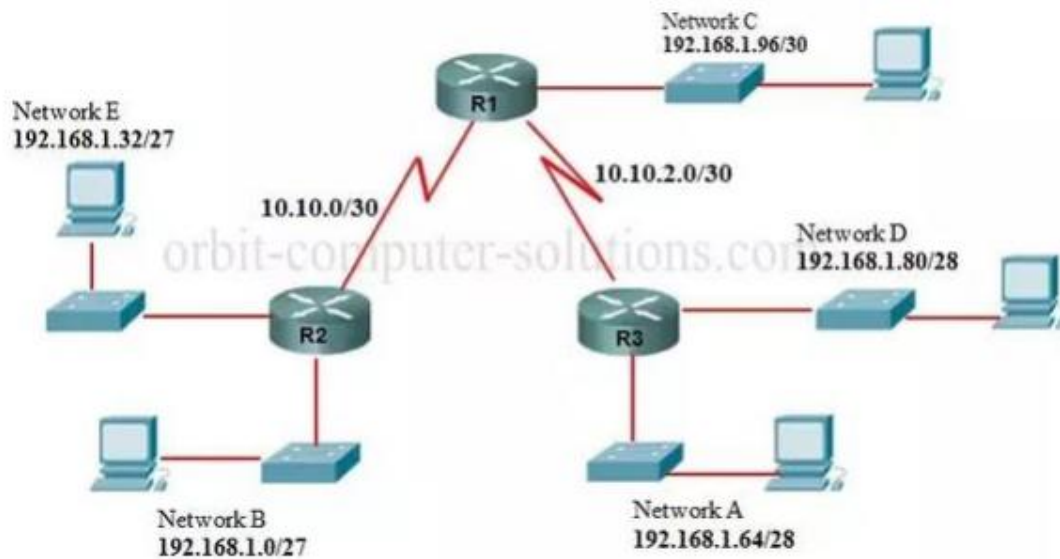


Figure No. 1: Routing

### Firewall:

Packet filtering: a type of packet classification, is often part of a firewall program for protecting a local network from unwanted intrusion. On the Internet, packet filtering is the process of passing or blocking packets at a network interface based on source and destination addresses, ports, or protocols. The process is used in conjunction with packet mangling and Network Address Translation. In a software firewall, packet filtering is done by a program called a packet filter. The packet filter examines the header of each packet based on a specific set of rules, and on that basis, decides to prevent it from passing (called DROP) or allow it to pass (called ACCEPT). There are three ways in which a packet filter can be configured, once the set of filtering rules has been defined. In the first method, the filter accepts only those packets that it is certain are safe, dropping all others. This is the most secure mode, but it can cause inconvenience if legitimate packets are inadvertently dropped. In the



second method, the filter drops only the packets that it is certain are unsafe, accepting all others. This mode is the least secure, but it causes less inconvenience, particularly in casual Web browsing. In the third method, if the filter encounters a packet for which its rules do not provide instructions, that packet can be quarantined, or the user can be specifically queried concerning what should be done with it.

### **Security and Network Monitoring:**

Security and network monitoring applications require packet classification and filtration. Packet classification implies finding out in a set of filters the highest priority filter matching the packet. The classification task has become more complex and is based on at least five packet header fields: source and destination IP addresses and ports and protocol identifier. The type of fields values are typically prefixes for IP addresses, ranges for ports and exact values or wildcard for protocol identifier

### **Differentiated Services:**

The ability to provide differential services to users with widely varying requirements is becoming increasingly important and internet service providers would like to provide these differentiated services using the same shared network infrastructure, the mechanism that enables this differentiation in a connectionless network, is packet classification.

## **2.2 Traditional Methods of Packet Classification**

### **Port based classification:**

A port number is the logical address of each application or process that uses a network or the Internet to communicate. Port numbers range from 0-65535.

It is the most common method the traffic classification is based on associating a well-known port number with a given traffic type. This method needs access only to the header of the packets. It matches port numbers to applications where an application is associated with a well-defined port numbers. It is fast and simple to implement. No complex computations required.

### **Payload Based Classification (Deep Packet Inspection):**

A payload refers to the actual data that is to be transferred to the desired destination. It is appended with a header having parameters like source and destination addresses for transport.

This method looks into the headers as well as the payloads inside the packet. It performs scanning of the payload bit by bit to identify a predefined sequence stream of a certain network protocol. The stored sequences of the bit streams are compared with those of the payloads and classified accordingly.

To overcome issues with old methods, Statistical flow-based classification is used.

## 2.3 Statistical Classification

Statistical based classification techniques is referred to as a modern technique for packet classification done at transport or network layer. A statistical classification algorithm is built on the basis of cluster analysis. Traffic is taken from the datasets made up of thousands of flows for many different application protocols. The classic approach of training and testing shows that cluster analysis yields very good results with very little information. The investigated applications are characterized from a signature at the network layer that is used to recognize such applications even when the port number is not significant. It deals with only a certain statistical parameter and not with the headers or data of the packets. It uses parameters such as:

1. Distribution of flow duration – The incoming packet flow is separated according to the incoming time and is further distributed on the basis of number of flows of various applications.
2. Flow idle time - It is a measure of the time for which the flow remains idle and no incoming packets are received. For e.g. - Packets received till (P.R.T): 9.54 p.m. Packets received after (P.R.A): 9.57 p.m.  
Hence, flow idle time = P.R.A – P.R.T = 3 minutes (no packets received for 3 minutes as the flow was idle).
3. Packet inter-arrival time – It is the mean or average of the time lapse between the data packets arriving at a certain host.
4. Total packet length – It is the total length of a single packet including its header and data.
5. Number of packets – It is the number of incoming packets in a given period of time.

6. Minimum/maximum, average and standard deviation of packet length  
Standard deviation of the packet lengths is found in order to differentiate the packets.
7. Minimum/maximum, average and standard deviation of inter-arrival time - Standard deviation of the inter-arrival time is found in order to differentiate the packets.

**Advantages:**

1. The main limitation of deep packet inspection is that security breach on instances where data is encrypted is avoided as in case of statistical approach inspection of packet is not needed.
2. Statistical approach does not need information like port number so any tampering with the port number, which is the limitation of port-based classification, is avoided.
3. Statistical classification is in real time and hence is adaptable to changes that happen in real time, it can train itself to adapt to the changes that happen over time.
4. Statistical classification attains high accuracy even though it is based on very limited information.
5. The statistical approach is faster than the traditional approach as it is based on the flow characteristics and does not require information like the source and destination IP etc. from the header.

These parameters are different for different applications. The traffic can be real-time, or can be obtained from the datasets such as NSL-KDD, DARPA, CAIDA, etc. The incoming packets having similar parameters are grouped into clusters, where one particular cluster belongs to one particular application. The formation of these clusters is done using machine learning (supervised learning).

We have used the parameters packet inter-arrival time, source IP address, and destination IP address for the implementation of our system.

**CHAPTER 3**

**NETWORK INTRUSION DETECTION AND  
PREVENTION SYSTEM**

### **3. Network Intrusion Detection and Prevention Systems**

Intrusion Detection system is a software which helps us to protect our system from other system when other person tries to access our system through network. It secures our system resources without giving access to other system. It is divided into two types. They are host-based intrusion detection system and network-based intrusion detection system. Then only we can secure our system. Predictive modeling is used to predict the output based on historical data. Classification is used to predict the output by historical data. Network intrusion detection system (NIDS) observes network traffic for identifying malicious packets. Core to NIDS function is packet classification component. It is in charge of scanning network packet header. An improved packet classification component bears direct result for NIDS performance.

#### **NIDS:**

Network intrusion detection systems are placed at a strategic point or points within the network to monitor traffic to and from all devices on the network. It performs an analysis of passing traffic on the entire subnet and matches the traffic that is passed on the subnets to the library of known attacks. Once an attack is identified, or abnormal behavior is sensed, the alert can be sent to the administrator. An example of an NIDS would be installing it on the subnet where firewalls are located in order to see if someone is trying to break into the firewall.

## **Detection**

The IPS has a number of detection methods for finding exploits, but signature-based detection and statistical anomaly-based detection are the two dominant mechanisms.

### **Signature-based detection:**

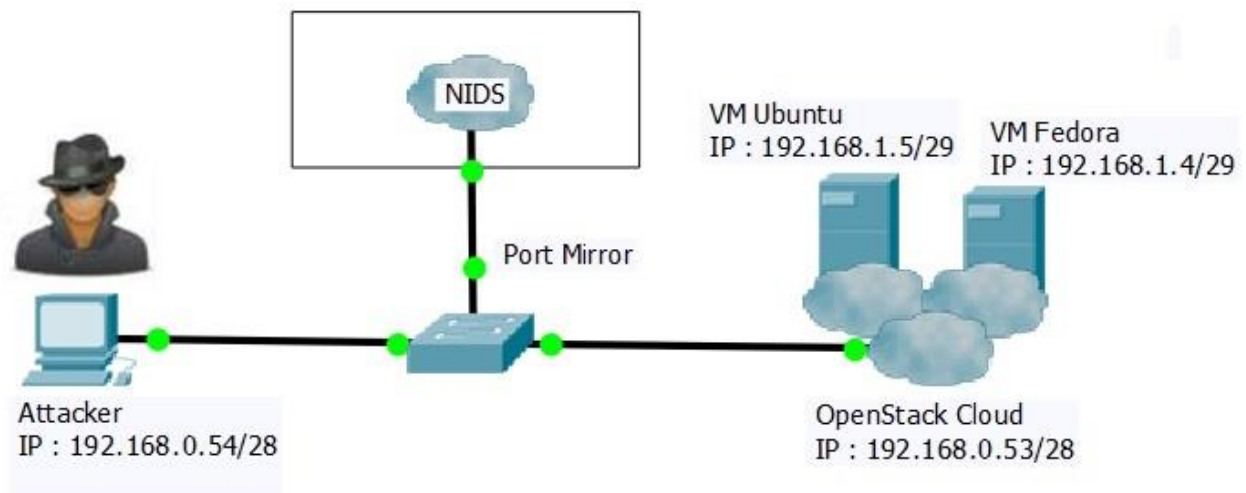
It is based on a dictionary of uniquely identifiable patterns (or signatures) in the code of each exploit. As an exploit is discovered, its signature is recorded and stored in a continuously growing dictionary of signatures. Signature detection for IPS breaks down into two types:

1. 1. Exploit-facing signatures identify individual exploits by triggering on the unique patterns of a particular exploit attempt. The IPS can identify specific exploits by finding a match with an exploit-facing signature in the traffic stream
2. 2. Vulnerability-facing signatures are broader signatures that target the underlying vulnerability in the system that is being targeted. These signatures allow networks to be protected from variants of an exploit that may not have been directly observed in the wild, but also raise the risk of false positives.

### **Statistical anomaly detection:**

It takes samples of network traffic at random and compares them to a pre-calculated baseline performance level. When the sample of network traffic activity is outside the parameters of baseline performance, the IPS takes action to handle the situation.





**Figure 2: A network Intrusion Detection System**

### **IDS vs FIREWALL:**

Though they both relate to network security, an IDS differs from a firewall in that a firewall looks outwardly for intrusions in order to stop them from happening. Firewalls limit access between networks to prevent intrusion and do not signal an attack from inside the network. An IDS evaluates a suspected intrusion once it has taken place and signals an alarm. An IDS also watches for attacks that originate from within a system. This is traditionally achieved by examining network communications, identifying heuristics and patterns (often known as signatures) of common computer attacks, and taking action to alert operators. A system that terminates connections is called an intrusion prevention system, and is another form of an application layer firewall.

## **NIPS:**

An **Intrusion Prevention System (IPS)** is a network security/threat prevention technology that examines network traffic flows to detect and prevent vulnerability exploits. Vulnerability exploits usually come in the form of malicious inputs to a target application or service that attackers use to interrupt and gain control of an application or machine. Following a successful exploit, the attacker can disable the target application (resulting in a denial-of-service state), or can potentially access to all the rights and permissions available to the compromised application.

### **Prevention**

The IPS often sits directly behind the firewall and provides a complementary layer of analysis that negatively selects for dangerous content. Unlike its predecessor the Intrusion Detection System (IDS)—which is a passive system that scans traffic and reports back on threats—the IPS is placed inline (in the direct communication path between source and destination), actively analyzing and taking automated actions on all traffic flows that enter the network. Specifically, these actions include:

- Sending an alarm to the administrator (as would be seen in an IDS)
- Dropping the malicious packets
- Blocking traffic from the source address
- Resetting the connection

As an inline security component, the IPS must work efficiently to avoid degrading network performance. It must also work fast because exploits can happen in near real-time. The IPS must also detect and respond accurately, so as to eliminate threats and false positives (legitimate packets misread as threats).

# **CHAPTER 4**

## **DISTRIBUTED DENIAL OF SERVICE ATTACKS**

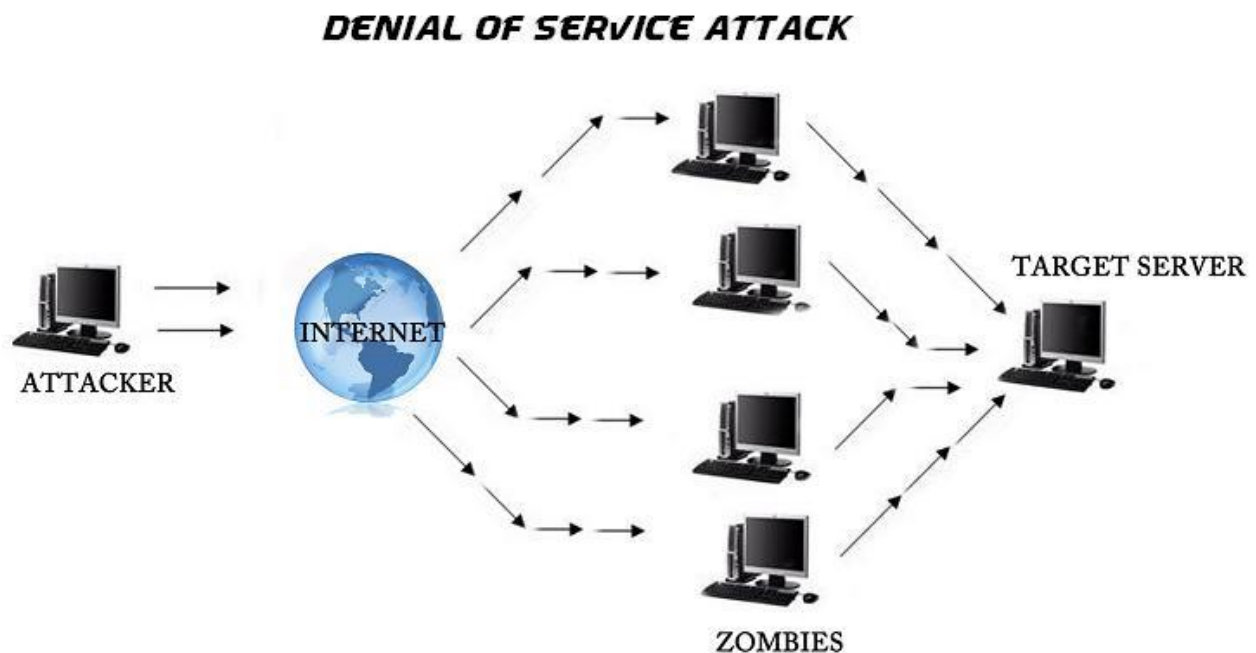
## 4. Distributed Denial of Services Attacks

### **DOS ATTACKS:**

A “denial of service” or DOS attack is used to tie up a website’s resources so that users who need to access the site cannot do so. Many major companies, have been the focus of DOS attacks in recent years. Because a DOS attack can be easily engineered from nearly any location, finding those responsible can be next to impossible. Unlike a virus or malware, a DOS attack doesn’t depend on a special program to run. Instead, it takes advantage of a natural vulnerability in the computer networks communicate. In a DOS attack, a computer is rigged to send not just one “introduction” to a server, but hundreds or sometimes thousands. The server—which cannot tell that the “introductions” are fake—sends back its usual response, waiting up to a minute in each case to hear a reply. When it gets no reply, the server shuts down the connection, and the computer executing the attack repeats, sending a new batch of fake requests.

A Denial-of-Service (DoS) attack is an attack meant to shut down a machine or network, making it inaccessible to its intended users. DoS attacks accomplish this by flooding the target with traffic or sending it information that triggers a crash. In both instances, the DoS attack deprives legitimate users (i.e. employees, members, or account holders) of the service or resource they expected.

Victims of DoS attacks often target web servers of high-profile organizations such as banking, commerce, and media companies, or government and trade organizations. Though DoS attacks do not typically result in the theft or loss of significant information or other assets, they can cost the victim a great deal of time and money to handle.



**Figure 3: A DOS Attack scenario**

### **Variations on DOS Attacks:**

One well-known variation on the DOS attack is a “DDOS” attack, or a “distributed denial of service” attack. A DDOS attack follows the same basic pattern as a standard DOS attack. Instead of using one computer that generates many fake user requests, a DDOS attack commandeers a fleet of computers to send requests. Because each computer has its own IP address, the server is further deceived into believing each request comes from a different “real” source. This makes DDOS attacks more likely to circumvent filters that are only designed to protect against multiple requests from the same source. Denial-of-service (DoS) attacks typically flood servers, systems or networks with traffic to overwhelm the victim resources and make it difficult or impossible for legitimate users to use them. While an attack that crashes a server can often be dealt with successfully by simply rebooting the system, flooding attacks can be more difficult to recover from. Following may indicate such an attack:

- Degradation in network performance, especially when attempting to open files stored on the network or accessing websites;
- Inability to reach a website;
- Difficulty in accessing any website; and
- A higher than usual volume of spam email.
- Many high-profile DoS attacks are distributed attacks, meaning the attack traffic is directed from multiple attack systems. While DoS attacks originating from a single source can be easier to mitigate because defenders can block network traffic from the offending source, attacks directed from multiple attacking systems are far more difficult to detect and defend against because it can be difficult to differentiate legitimate traffic from malicious traffic and filter malicious packets when they are sent from all over the internet.

### **Types of DDOS Attacks:**

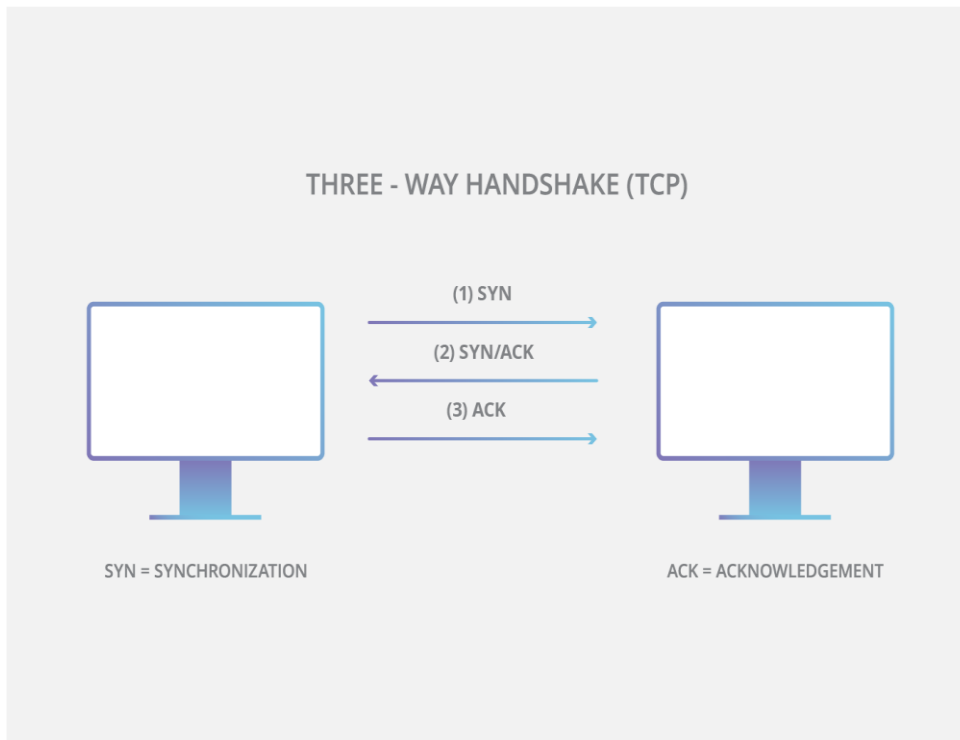
There are two general methods of DoS attacks: flooding services or crashing services. Flood attacks occur when the system receives too much traffic for the server to buffer, causing them to slow down and eventually stop. Popular flood attacks include:

- **Buffer overflow attacks** – the most common DoS attack. The concept is to send more traffic to a network address than the programmers have built the system to handle. It includes the attacks listed below, in addition to others that are designed to exploit bugs specific to certain applications or networks
- **ICMP flood** – leverages misconfigured network devices by sending spoofed packets that ping every computer on the targeted network, instead of just one

specific machine. The network is then triggered to amplify the traffic. This attack is also known as the smurf attack or ping of death.

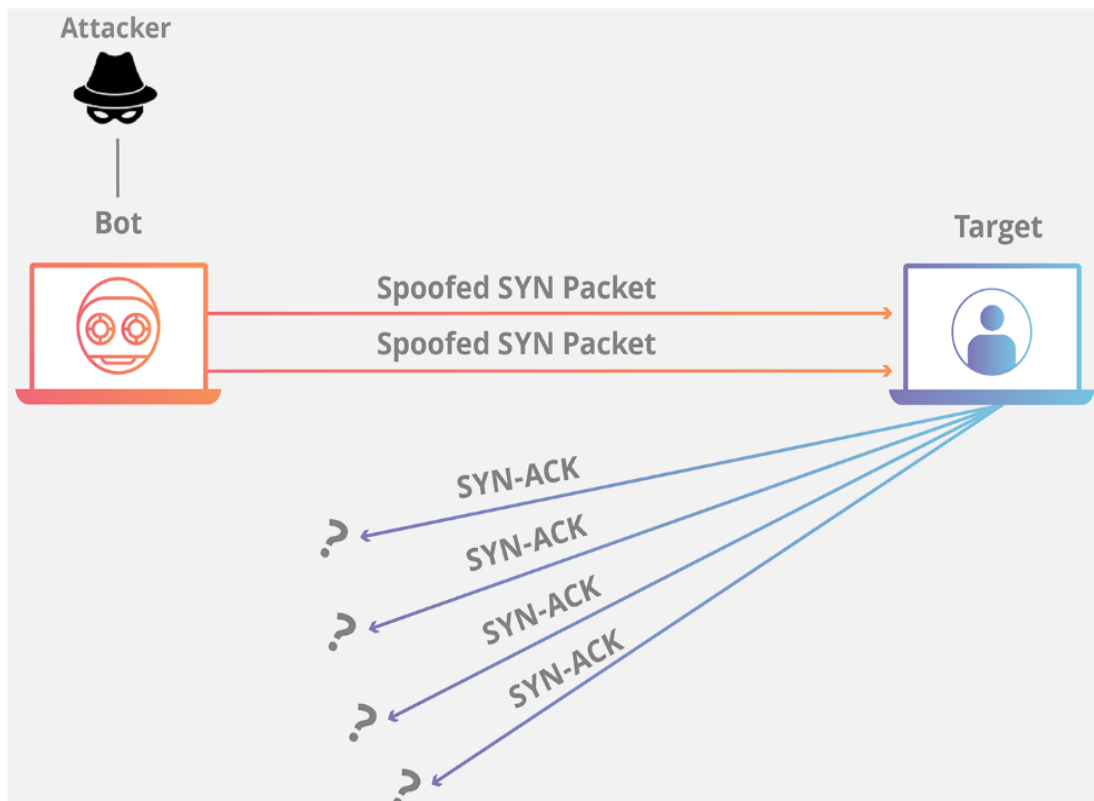
- **SYN flood** – sends a request to connect to a server, but never completes the handshakes. It continues until all open ports are saturated with requests and none are available for legitimate users to connect to.
- **UDP Flood** -UDP flood, by definition, is any DDoS attack that floods a target with User Datagram Protocol (UDP) packets. The goal of the attack is to flood random ports on a remote host. This causes the host to repeatedly check for the application listening at that port, and (when no application is found) reply with an ICMP ‘Destination Unreachable’ packet. This process saps host resources, which can ultimately lead to inaccessibility.
- **Ping of Death**- A ping of death ("POD") attack involves the attacker sending multiple malformed or malicious pings to a computer. The maximum packet length of an IP packet (including header) is 65,535 bytes. In this case, a large IP packet is split across multiple IP packets (known as fragments), and the recipient host reassembles the IP fragments into the complete packet. In a Ping of Death scenario, following malicious manipulation of fragment content, the recipient ends up with an IP packet which is larger than 65,535 bytes when reassembled. This can overflow memory buffers allocated for the packet, causing denial of service for legitimate packets.
- **HTTP Flood**- In an HTTP flood DDoS attack, the attacker exploits seemingly-legitimate HTTP GET or POST requests to attack a web server or application. HTTP floods do not use malformed packets, spoofing or reflection techniques, and require less bandwidth than other attacks to bring down the targeted site or server. The attack is most effective when it forces

the server or application to allocate the maximum resources possible in response to each single request.



**Figure 4: A normal TCP connection**





**Figure 5: A TCP-SYN Flooding**

# **CHAPTER 5**

## **DESIGN SPECIFICATIONS**

## 5. Design Specifications

Following are the specifications of the designed Network Intrusion Detection and Prevention system.

**Statistical Parameters:** Number of packets for a certain duration of flow.

**Other Parameters:** Packet length, Source IP and Destination IP address

**Type of the attack:** TCP-SYN Flooding

**IP version:** IPv4 addresses.

# **CHAPTER 6**

## **THE PROPOSED SYSTEM**

## 6. The Proposed System

### 6.1 Source-Destination IP pair and number of packets approach

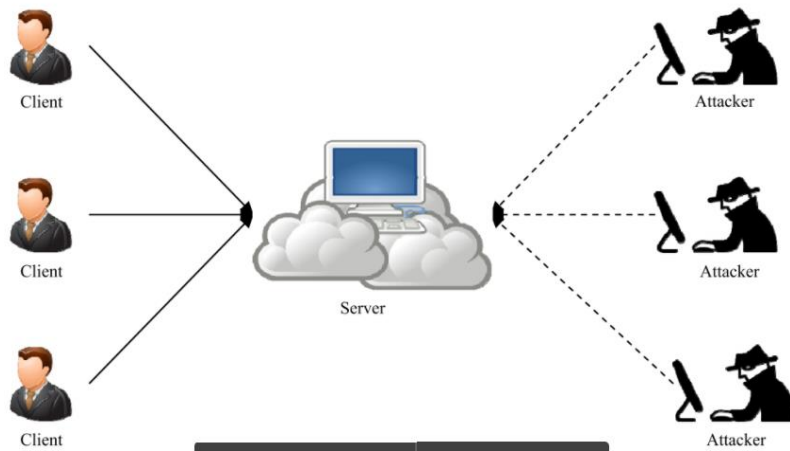


Figure 6: DDOS scenario

In the pairing-based approach, the source IP Addresses and destination IP Addresses between which connection is requested are paired. The number of transfers/connections between this pair are found out, also called as Pair count. A packet is an attacking one if the source requests connections to the same destination more frequently than an assumed threshold. i.e. if the pair count is too large between a specific IP address pair, it seems abnormal. As a part of DDOS attacks, TCP flooding involves many connection requests or transfers between a pair, hence this pair count approach correctly identifies a DDOS attack. The threshold can be manually adjusted by the system administrator to cater for the varying requirements of a network. This threshold varies according to different parameters like network, number of users, area of network, protocol, etc. and is hence set manually.

Example-

Source IP	Destination IP	Transfer
192.168.2.3	104.11.6.7	TCP
192.168.2.3	104.11.6.7	TCP
190.68.1.4	104.11.6.7	TCP
192.168.2.3	121.5.63.1	TCP
192.168.2.3	104.11.6.7	TCP
192.168.2.3	104.11.6.7	TCP
190.68.1.4	121.5.63.1	TCP

**Table 1: Source and Destination IPs**

Hence, the pairs found out in the above example based on connections are:

192.168.2.3---104.11.6.7

192.168.2.3---121.5.63.1

190.68.1.4---121.5.63.1

190.68.1.4---104.11.6.7

Now, the system computes pair counts (TCP connections and transfers) for the above pairs:

192.168.2.3---104.11.6.7                      4

192.168.2.3---121.5.63.1                      1

190.68.1.4---121.5.63.1                      1

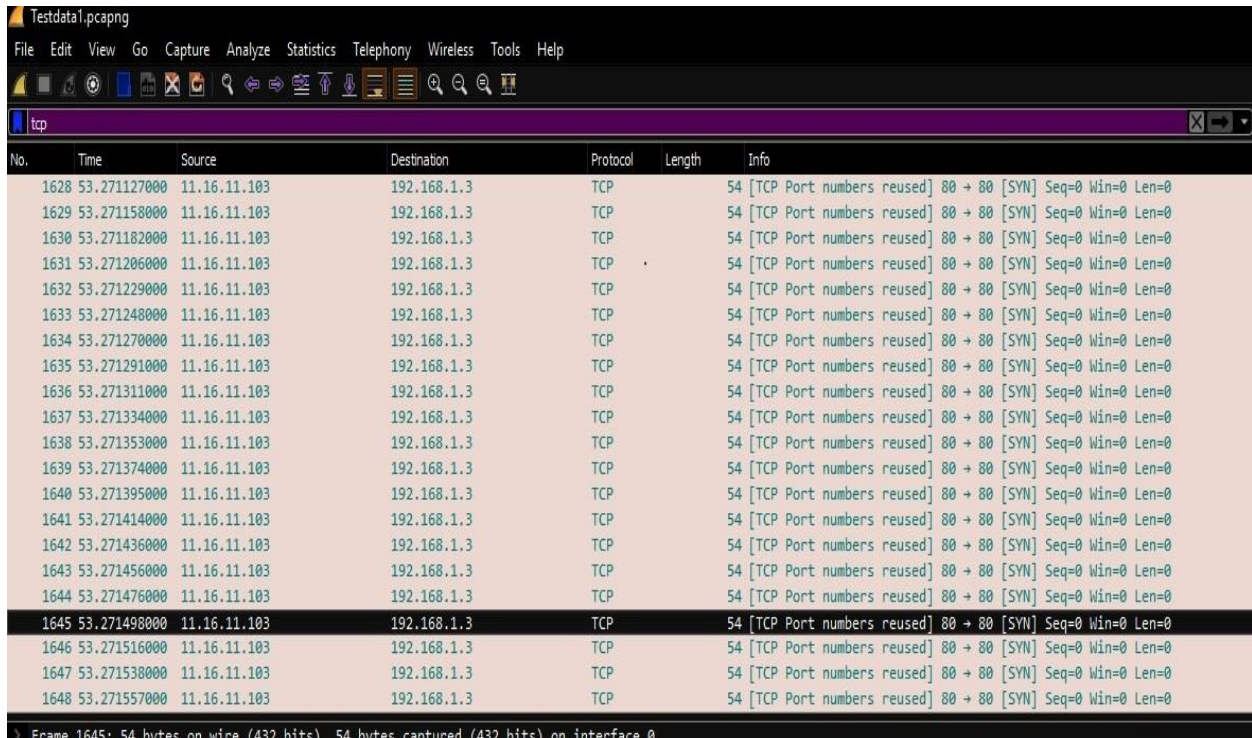
190.68.1.4---104.11.6.7                      1

Hence, for the pair 192.168.2.3---104.11.6.7, the pair count seems abnormally larger than the others which would identify this source IP address as an attacking one.

## 6.2 Inter-arrival Time-based approach

When a DoS or DDoS attack happens n number packets from same or different IP addresses reach the destination at same time instant. Therefore the count of number of packet per second exceeds the set threshold and can be categorized as attack. This threshold is set by the system operation manually in the preprocessing stage depending on type of attack, users and other network requirements. If this threshold is exceeded all the packets arriving at that time instant are declared as attack and dropped.

Consider the following packet capturing below,

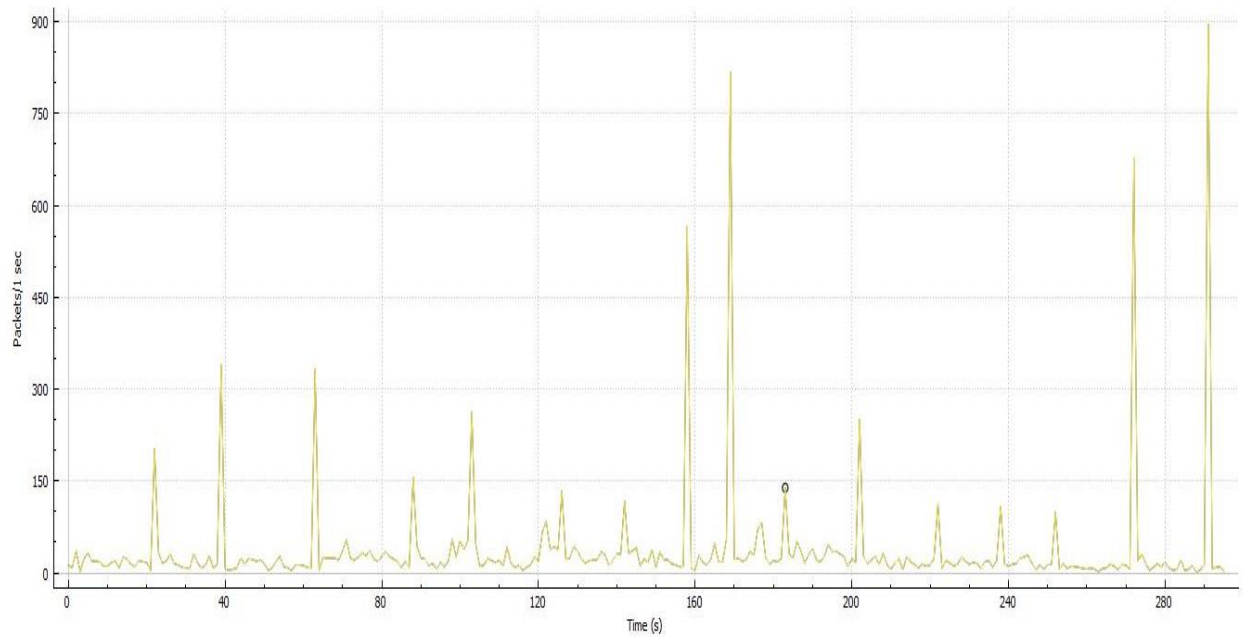


No.	Time	Source	Destination	Protocol	Length	Info
1628	53.271127000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1629	53.271158000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1630	53.271182000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1631	53.271206000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1632	53.271229000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1633	53.271248000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1634	53.271270000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1635	53.271291000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1636	53.271311000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1637	53.271334000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1638	53.271353000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1639	53.271374000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1640	53.271395000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1641	53.271414000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1642	53.271436000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1643	53.271456000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1644	53.271476000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1645	53.271498000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1646	53.271516000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1647	53.271538000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0
1648	53.271557000	11.16.11.103	192.168.1.3	TCP	54	[TCP Port numbers reused] 80 → 80 [SYN] Seq=0 Win=0 Len=0

Frame 1645: 54 bytes on wire (432 bits) 54 bytes captured (432 bits) on interface 0

Figure 7: TCP Flooding

As a TCP attack is executed on the destination 192.168.1.3 all the attack packets arrive the destination at same time instant.



**Figure 8: Bursts/sec**

By looking at the graph of time v/s packet per sec we can clearly identify the time instances of attacks, as the number of packets arriving the destination is evidently large.

The advantage of this system is that if the attack is a distributed DoS attack, in which the attack happens from multiple IP addresses rather than a few fixed addresses, the system does not fail as the source and destination IP are not required to predict the output.

One drawback of the system is that if normal packets arrive at the same time instant as attack packets, even these packets are classified as attack.



### 6.3 Block Diagram

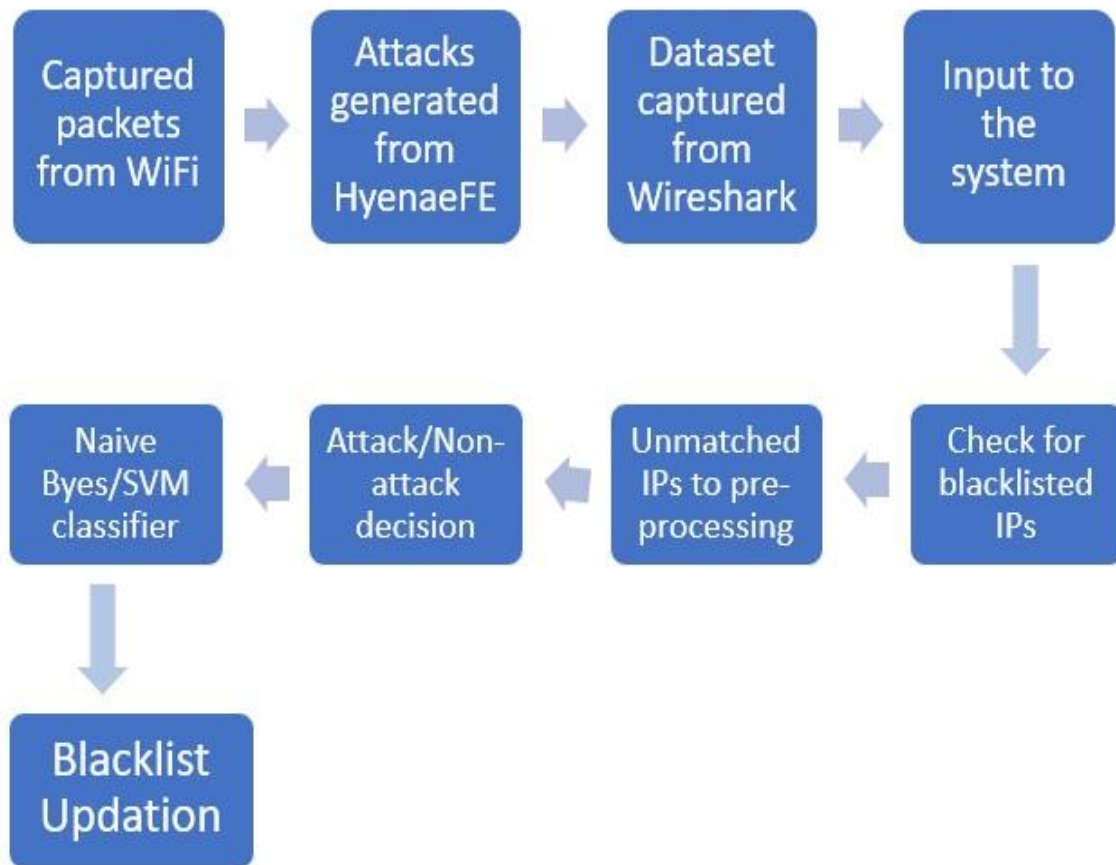


Figure 9: Block Diagram of the system

## 6.4 Explanation

The incoming packets from the desired network are given to the Network Intrusion Detection System. The system determines whether the incoming packets are malicious or safe and decided further whether to drop or forward the packets.

In the detection system the statistical parameter number of packets/flow is used to determine whether the incoming packet is a malicious one. For a certain duration of flow, a fixed number of packets are arrived from a certain source to a certain destination. During an attack, the attacker sends a large number of packets to the victim.

A threshold value is set for the number of packets per duration of a flow. A 5 min flow is taken into consideration. In this system number of packets from the source to destination is calculated in a fixed time interval. If the count exceeds the set threshold value of the count those packets are classified as attack packets. Their corresponding source IP Addresses are considered as unauthorized IPs and are stored in blacklist table.

The prevention system first scans the incoming packets and their source IP addresses. If any of the blacklisted IPs are found in the incoming data then those packets are automatically dropped and not forwarded to the target system. The remained packets are first given to the detection system to identify the malicious packets.

# **CHAPTER 7**

## **ALGORITHM FOR IMPLEMENTATION**

## 7. Algorithm for Implementation

The preprocessing algorithm is used to calculate the number of packets in a certain duration of the flow from the source to the target destination.

Following are the steps in the Preprocessing algorithm:

- 1) Load dataset
- 2) For I=1: n
- 3) Source=data (I, 1)
- 4) Destination= (I, 2)
- 5) For J=1: n
- 6) Count=find (data (J, 1) ==Destination) & (data (J, 2) ==Source)
- 7) If Count>=K
- 8) NewData (I, 2) =data (I, 2)
- 9) NewData(I, 1) =1
- 10) Else
- 11) NewData (I, 1) =data (I, 1)
- 12) NewData (I, 2) =0
- 13) End

Here:

- n is the number of packets
- Destination is the destination IP address
- Source is the source IP address
- Count is the number of packets from the same source to the same destination within 300 seconds
- K is the threshold for a packet to be considered an attacking

packet

- 1: indicates abnormal packets (blacklist array)
- 0: indicates normal packets

NewData is the new entries with the Source IP Destination IP the count and the Label Attack which has values 1 or 0

This preprocessed data is given as the training set for the classifiers.

The Naïve Bayes Classifier and SVM classifier are used in the detection system.

**Detection phase:** During the detection phase, the detection sub-system collects the incoming packets within a time frame, for example 300 seconds. The collected packets are subjected to a blacklist check to test whether their sources are blacklisted as attackers of the cloud system. If the packet source is listed in the attacker blacklist, the detection system will send the packets directly to the prevention sub-system without further processing. If the packet source is not blacklisted, the incoming packet will be passed to the classifier to decide whether the packets are normal (originating from a client) or abnormal (originating from an attacker). A packet is an attacking one if the source requests connections to the same destination more frequently than an assumed threshold. The threshold can be manually adjusted by the system administrator to cater for the varying requirements of a network. If a packet is normal, the detection system will send it to its destination (the cloud service provider). Otherwise, the detection sub-system will send the packet to the prevention sub-system.

**Prevention Phase:** When the packets reach the prevention system, they are attacking packets by the detection sub-system. The prevention sub-system first alerts the system administrator of the attacks. Then, the prevention sub-system will add the attacking source address to the attacker blacklist used by the detection sub-system, if it is not already on the list. Finally, the attacking packet will be dropped. The overall architecture of the system is shown in Figure. Pre-processing algorithm is used to determine whether these packets are normal or abnormal by counting the number of requests for a connection from an IP address and checking whether it exceeds a predefined threshold within a certain time frame. This algorithm is applied to the training data used for each classifier. As a result, each classifier used will predict the behavior of the attackers according to the pre-processing algorithm.

# **CHAPTER 8**

## **MACHINE LEARNING**

## 8. Machine Learning

It is defined as a field of study that gives computers the ability to learn without being explicitly programmed. A machine or a software learns from an experience with respect to some sets of tasks and performances such that its performance measure improves more and more with that experience.

Machine learning is broadly classified into two types namely Supervised Learning and Unsupervised learning.

### **Supervised Learning:**

In Supervised learning a dataset is available and the correct output is already known. The relationship between the input and output is already known. The dataset is available is already available with the expected or correct set of outputs. The machine is trained accordingly and then the supervised learning algorithms are implemented on new set of features to get the expected results.

Supervised Learning is of two types:

Regression: In regression we try to predict results within a continuous output.

Classification: In classification we try to predict the results into discrete classes.

- Algorithms used in supervised learning are
- Decision trees
- Support vector machine (SVM)
- k-Nearest Neighbors
- Naive Bayes
- Random forest

In this project we are using Support Vector Machine and Naïve Bayes algorithms which are classification supervised learning algorithms.



## **Unsupervised Learning:**

Unsupervised Machine learning is a type in which the system is trained from the datasets without label fields. The input output relationship is not known. In this type of learning the machine or a system analyzes the given data and tries to detect the similarities or patterns and predicts the output. This is used in application where the effect of variables is not known. In unsupervised learning there is no feedback based on the prediction results.

Clustering is a type of unsupervised learning task wherein the machine automatically divides the data into groups of clusters of similar properties. The data is organized into high intra-class similarity and low intra-class similarity.

Algorithms used in unsupervised machine learning are:

- Hierarchical Clustering
- K-means Clustering
- Apriori Association Analysis

## 8.1. Support Vector Machine Classifier

Support Vector Machine is a supervised type of Machine learning. It is basically a classification algorithm. The goal of a support vector machine is to find the optimal separating hyperplane which maximizes the margin of the training data. It is generally a binary classification algorithm which categorizes the data into two distinct classes and is usually depicted in 2D space. This algorithm finds out the best suitable hyperplane. A hyperplane is a generalization of a plane.

- In one dimension, a hyperplane is called a point
- In two dimensions, it is a line
- In three dimensions, it is a plane
- In more dimensions it is called a hyperplane.

Optimal separating hyperplane means to find out a line that will have maximum perpendicular distance from the feature points on either of its sides. These feature points which are considered as the references are called as the "Support Vectors." A linear equation is computed for the desired vectors. To classify the features in distinct classes a "Kernel Trick" is used. Kernel trick is basically used to find out the similarity between the objects. Different types of Kernel tricks are Linear Kernels, Polynomial Kernel, Radial Basis Function kernel.

The algorithm has following steps:

1. Generate a dataset of certain known parameters.
2. Data preprocessing that is removal of the redundant or empty features.
3. Mapping the features on a two-dimensional space.
4. Using the appropriate kernel trick and class factor compute the maximum margin and find out the best hyperplane.
5. Train the dataset accordingly using the vector equations.

6. From the testing dataset classify the features into two distinct classes using SVM

**Support Vector Implementation in our system:**

In our project we have used the algorithm to detect where a burst of packets is attack or non-attack type of burst.

The attacks are determined by assigning 1 and non-attacks are determined by assigning 0.

**The features passed to SVM are:**

Count- number of packets captured for a particular source IP - destination IP pair or number of packets captured in a 1 second time

Classification- whether a packet is attack or non-attack

**Working:**

Support Vector Machine classifies data by selecting the accurate hyper-plane.

Euclidean distance from this hyper plane to the data point is maximum.

**Training:**

SVM divides the data into two clusters: Attack and non-attack

**Testing:**

Data with count is passed to the Algorithm.

The algorithm then calculates distances from that point to nearest counts from both clusters.

The cluster with minimum distance is selected.

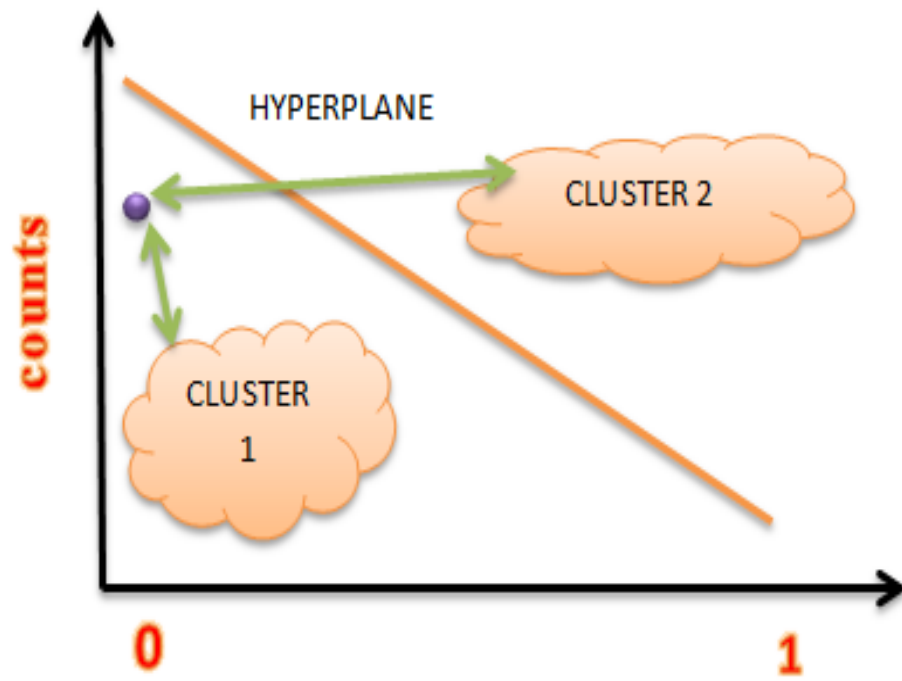


Figure 10: SVM in our system

## 8.2 Naïve Bayes Classifier

The Naïve Bayes algorithm is called “Naïve” because it makes the assumption that the occurrence of a certain feature is independent of the occurrence of other features.

Bayes: It refers to statistician and philosopher Thomas Bayes and theorem named after him, “Bayes’ theorem” which is the base of the Naïve Bayes algorithm.

Theorem: Bayes’ Theorem states as Probability of B given A is equal to the probability of the event A and B multiplied by the probability of A upon probability of B

$$P(A|B) = P(B|A) P(A) / P(B)$$

$P(A|B)$ : probability (conditional probability) of occurrence of event A given the event B is true.

$P(A)$ : probability of occurrence of A

$P(B)$ : probability of occurrence of B

$P(B|A)$ : probability of occurrence of event B given event A is true

### **Advantages:**

- It’s relatively simple to understand and build
- It’s easily trained, even with a small dataset
- It’s fast
- It’s not sensitive to irrelevant feature

### **Disadvantages:**

- It assumes every feature is independent, which isn’t always the case

In our system, the Naïve Bayes classifier calculates the probability of attack given the count of number of packets. It calculates the probability using individual probabilities such as probability of count greater than threshold, probability that the packet is an attack and probability of count given an attack packet.

The training data is a 300sec captured real-time data which has the training feature as the count of number of packets and the training label as the Attack or Non-Attack Packet.

### **Naïve Bayes Implementation in our system:**

In our project we have used the algorithm to detect where a burst of packets is attack or non-attack type of burst.

The attacks are determined by assigning 1 and non-attacks are determined by assigning 0.

### **The features passed to Naïve Bayes are:**

Count- number of packets captured for a particular source IP - destination IP pair or number of packets captured in a 1 second time

Classification- whether a packet is attack or non-attack

### **Training:**

Both the features are passed to Naïve Bayes for training.

### **Testing:**

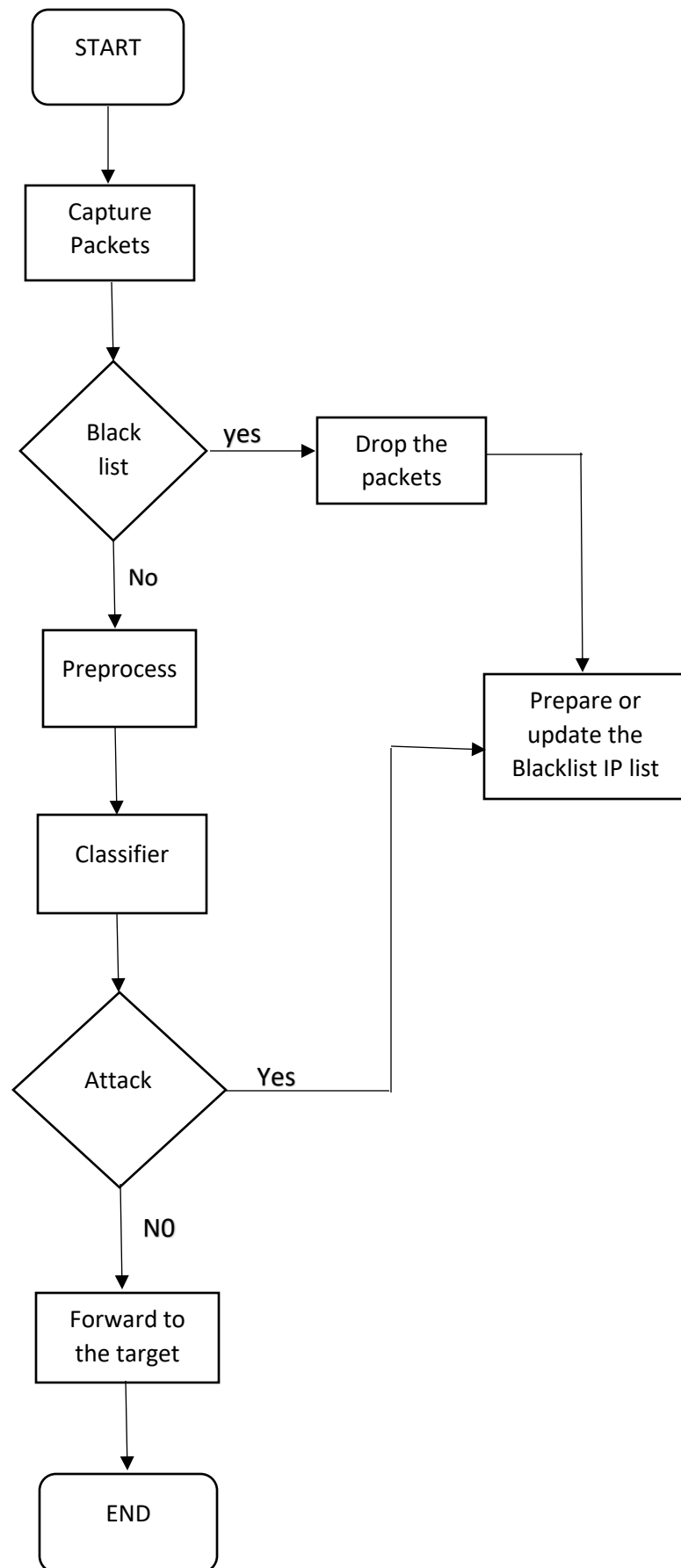
Naïve Bayes is a probabilistic model

In the testing phase the count is passed to Naïve Bayes and based on the count it classifies it as attack or non-attack.

This is done by calculating the probabilities of a count being attack or non-attack based on the training data.

# **CHAPTER 9**

## **FLOWCHART OF THE ALGORITHM**





# **CHAPTER 10**

## **PRACTICAL IMPLEMENTATION**

## 10. Practical Implementation

In order to test the system, a dataset is created and captured using the Wireshark software. Primarily, normal packets are captured from the WiFi and captured on Wireshark. These are the packets involved in browsing and internet. Different websites generate packets with different parameters such as length, timestamp, burst, etc.

Then, a software called HyenaeFe is used to generate attacks (these are fake attacks only for testing purposes), but they have similar parameters like real-time attacks. These packets are also captured in Wireshark.

Now, Wireshark contains all possible packets from WiFi and Hyenae including all the protocols. Here, a TCP filter is applied to the capturing, so that only TCP packets are available and SYN flooding can be observed. Data capturing is done for 5 minutes(300 seconds) and then stopped.

The input to the system is this dataset created using Wireshark. This dataset (normal packets + attacked packets) is converted into a CSV (comma separated values) and stored for further processing. Later, source-destination IPs are paired and the pair count is found. Similar is done for the time approach also, and its time count is computed.

The source IP addresses from the dataset are checked with the blacklisted IPs (list created for prevention/IPS). If any of the IP matches with the blacklisted one, then it is dropped right away. This step optimizes the system by avoiding an already detected abnormal source IP to go through the entire processing and to be re-detected as an attacked packets' IP address. It saves time and reduces complexity.

If any of the dataset's IPs do not match the blacklist of IPs, then, the remaining dataset is given to the pre-processing algorithm. Based on the fixed threshold value,

the algorithm determines which packets are attacked and which are the normal ones. This algorithm labels the packets as 0 and 1 or to separate them into groups.

The classifier Naive Bayes/ SVM's work starts here. The classifier is fed with the decision of attack/non-attack from the pre-processing algorithm and also the pair count as computed previously. The AI classifier trains itself on the given dataset and the two features. Hence, later, when any new dataset is given to the classifier, it uses the knowledge from training, and determines the classification accordingly. Thus, any new dataset (with similar functions from training) can be inputted as testing to these ML classifiers. The TP, TN, FP, FN and accuracy, sensitivity, etc. are calculated over here.

Now, the old blacklist is again updated according to the results from the classifiers, hence the source IPs to be blocked (as a part of the prevention) are added to this list. This shall optimize any further processing for intrusion detection.

Thus, an IDS and IPS is built which uses statistical classification and is a modern efficient way for packet classification.

## 10.1 Software Used

### **Wireshark:**

Wireshark is a free and open source packet analyzer. It is used for network troubleshooting, software and communications protocol development, and education. Originally named Ethereal, the project was renamed Wireshark in May 2006 due to trademark issues. Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows. There is also a terminal-based (non-GUI) version called TShark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of the GNU General Public License. In our project we have used wireshark to capture TCP packets and create our dataset.

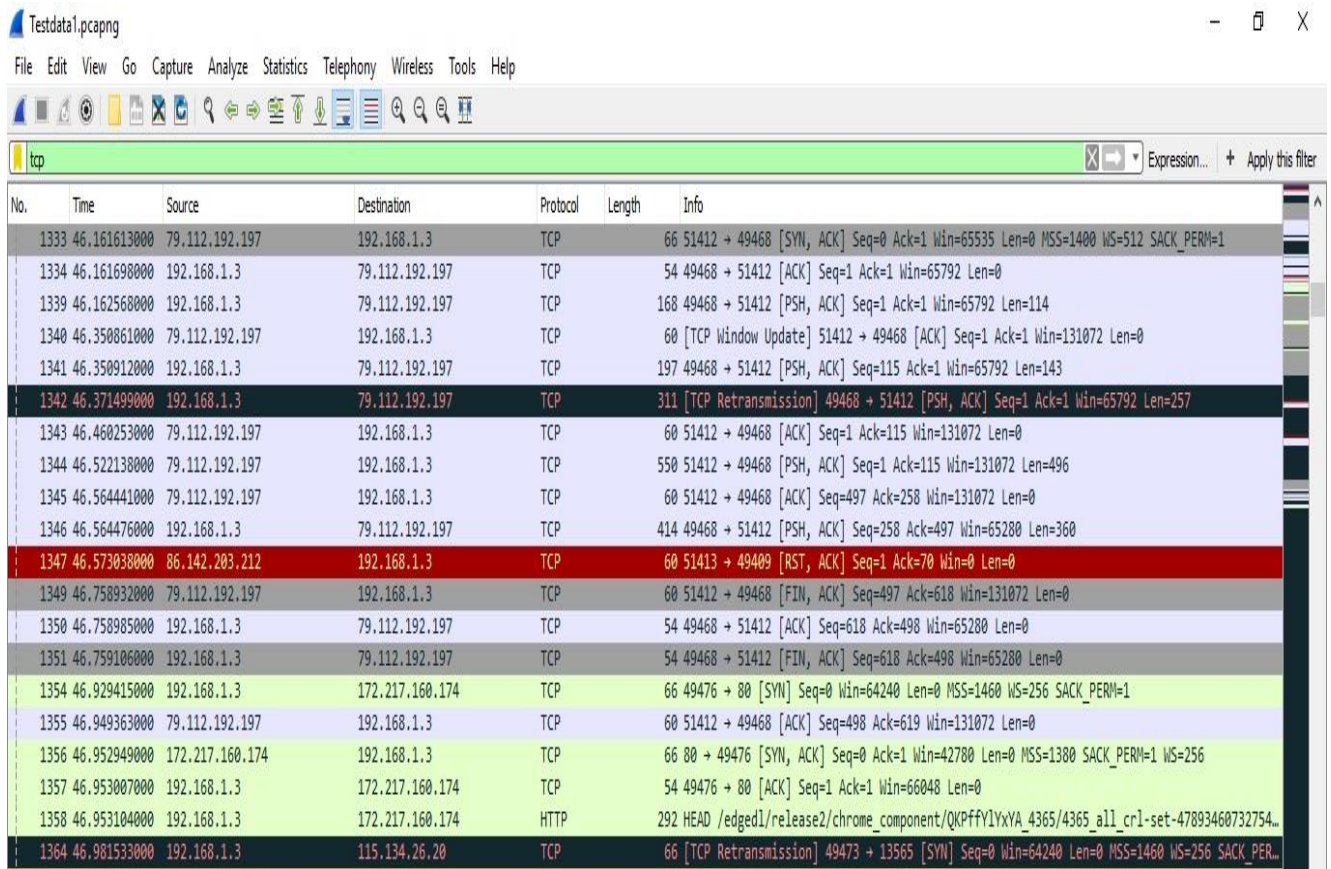
### **Applications:**

- Network administrators use it to troubleshoot network problems
- Network security engineers use it to examine security problems
- QA engineers use it to verify network applications
- Developers use it to debug protocol implementations
- People use it to learn network protocol internals

### **Features:**

- Available for UNIX and Windows.
- Capture live packet data from a network interface.
- Open files containing packet data captured with tcpdump/WinDump, Wireshark, and a number of other packet capture programs.
- Import packets from text files containing hex dumps of packet data.

- Display packets with very detailed protocol information.
- Save packet data captured.
- Export some or all packets in a number of capture file formats.
- packets on many criteria.
- Search for packets on many criteria.
- Colorize packet display based on filters.
- Create various statistics.



No.	Time	Source	Destination	Protocol	Length	Info
1333	46.161613000	79.112.192.197	192.168.1.3	TCP	66	51412 → 49468 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 WS=512 SACK_PERM=1
1334	46.161698000	192.168.1.3	79.112.192.197	TCP	54	49468 → 51412 [ACK] Seq=1 Ack=1 Win=65792 Len=0
1339	46.162568000	192.168.1.3	79.112.192.197	TCP	168	49468 → 51412 [PSH, ACK] Seq=1 Ack=1 Win=65792 Len=114
1340	46.350861000	79.112.192.197	192.168.1.3	TCP	60	[TCP Window Update] 51412 → 49468 [ACK] Seq=1 Ack=1 Win=131072 Len=0
1341	46.350912000	192.168.1.3	79.112.192.197	TCP	197	49468 → 51412 [PSH, ACK] Seq=115 Ack=1 Win=65792 Len=143
1342	46.371499000	192.168.1.3	79.112.192.197	TCP	311	[TCP Retransmission] 49468 → 51412 [PSH, ACK] Seq=1 Ack=1 Win=65792 Len=257
1343	46.460253000	79.112.192.197	192.168.1.3	TCP	60	51412 → 49468 [ACK] Seq=1 Ack=115 Win=131072 Len=0
1344	46.522138000	79.112.192.197	192.168.1.3	TCP	550	51412 → 49468 [PSH, ACK] Seq=1 Ack=115 Win=131072 Len=496
1345	46.564441000	79.112.192.197	192.168.1.3	TCP	60	51412 → 49468 [ACK] Seq=497 Ack=258 Win=131072 Len=0
1346	46.564476000	192.168.1.3	79.112.192.197	TCP	414	49468 → 51412 [PSH, ACK] Seq=258 Ack=497 Win=65280 Len=360
1347	46.573038000	86.142.203.212	192.168.1.3	TCP	60	51413 → 49409 [RST, ACK] Seq=1 Ack=70 Win=0 Len=0
1349	46.758932000	79.112.192.197	192.168.1.3	TCP	60	51412 → 49468 [FIN, ACK] Seq=497 Ack=618 Win=131072 Len=0
1350	46.758985000	192.168.1.3	79.112.192.197	TCP	54	49468 → 51412 [ACK] Seq=618 Ack=498 Win=65280 Len=0
1351	46.759106000	192.168.1.3	79.112.192.197	TCP	54	49468 → 51412 [FIN, ACK] Seq=618 Ack=498 Win=65280 Len=0
1354	46.929415000	192.168.1.3	172.217.160.174	TCP	66	49476 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
1355	46.949363000	79.112.192.197	192.168.1.3	TCP	60	51412 → 49468 [ACK] Seq=498 Ack=619 Win=131072 Len=0
1356	46.952949000	172.217.160.174	192.168.1.3	TCP	66	80 → 49476 [SYN, ACK] Seq=0 Ack=1 Win=42780 Len=0 MSS=1380 SACK_PERM=1 WS=256
1357	46.953007000	192.168.1.3	172.217.160.174	TCP	54	49476 → 80 [ACK] Seq=1 Ack=1 Win=66048 Len=0
1358	46.953104000	192.168.1.3	172.217.160.174	HTTP	292	HEAD /edged1/release2/chrome_component/QPffYLYxYA_4365/4365_all_crl-set-47893460732754...
1364	46.981533000	192.168.1.3	115.134.26.20	TCP	66	[TCP Retransmission] 49473 → 13565 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1

Figure 11: Wireshark window

## **HyenaeFE:**

Hyenae is a highly flexible platform independent network packet generator. It allows you to reproduce several MITM, DoS and DDoS attack scenarios. It comes with a clusterable remote daemon and an interactive attack assistant. In our project we have used Hyenae to generate TCP flooding attacks.

### **Features:**

- ARP-Request flooding
- ARP-Cache poisoning
- PPPoE session initiation flooding
- Blind PPPoE session termination
- ICMP-Echo flooding
- ICMP-Smurf attack
- ICMP based TCP-Connection reset
- TCP-SYN flooding
- TCP-Land attack
- Blind TCP-Connection reset
- UDP flooding
- DNS-Query flooding
- DHCP-Discover flooding
- DHCP starvation attack
- DHCP-Release forcing
- Cisco HSRP active router hijacking
- Pattern based packet address configuration
- Intelligent address and address protocol detection

- Smart wildcard-based randomization
- Daemon for setting up remote attack networks
- HyenaeFE QT-Frontend support

HyenaeFE

Operation Mode  
 Attack from local machine

Network Interface  
 Microsoft

Network Protocol  
 IP-Version: IPv4  
 Packet Type: TCP

Send Parameters  
 Fixed packet limit: 116 - 1000  
 No send delay: 1000 - 3000  
 No send duration: 10000 - 15000

TCP Packets  
 Source Pattern: %-112.12.3.6@80  
 Destination Pattern: 2C:33:7A:FA:EB:8F-192.168.1.3@80  
 TCP Flags: ☐ FIN ☒ SYN ☐ RST  
☐ PSH ☐ ACK  
 TTL (Time To Live): 128  
 Acknowledgement No.: 0  
 Window Size: 0  
 Sequence No. Offset: 0  
 Sequence No. Incr. Steps: 1

Packet Payload  
 No payload

Command Line Usage  

```
hyenae -I 1 -A 4 -a tcp -s %-112.12.3.6@80 -d 2C:33:7A:FA:EB:8F-192.168.1.3@80 -f S -t 128 -k 0 -w 0 -q 0 -Q 1 -c 116
```

About Execute

**Figure 12: HyenaeFE Packet Generator**

## **Spyder:**

Spyder (formerly Pydee) is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates NumPy, SciPy, Matplotlib and IPython, as well as another open source software. It is released under the MIT license Spyder is extensible with plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and Rope. It is available cross-platform through Anaconda, on Windows with WinPython] and Python (x,y), on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu Spyder makes use of Qt either through the binding PyQt or PySide. This flexibility is reached through a small abstraction layer called QtPy.

### **Features:**

- Editor with syntax highlighting and introspection for code completion
- support for multiple Python consoles (including IPython)
- the ability to explore and edit variables from a GUI



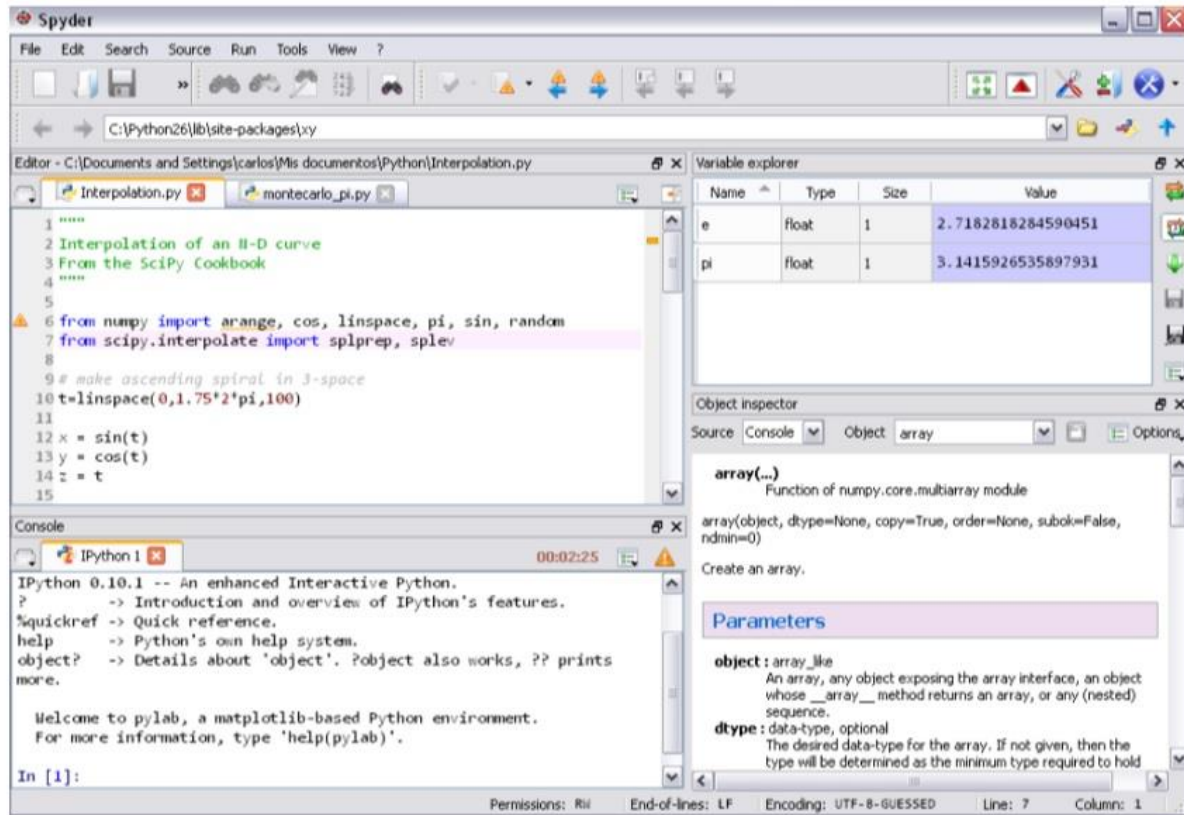


Figure 13: Spyder window for Python

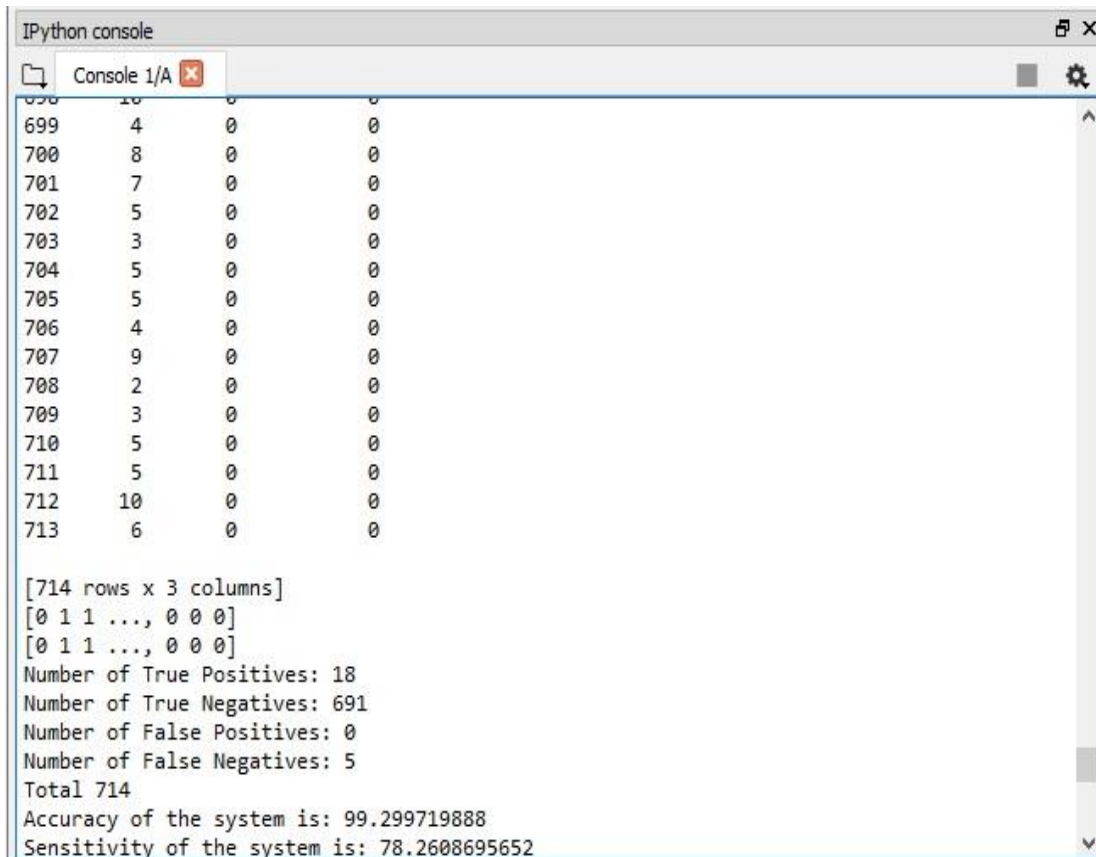
# **CHAPTER 11**

## **TESTING**

# 11. Testing

We have used three test cases for testing our designed system.

1. Wireshark is used for capturing the data for 5 mins time interval. It contains some normal data and some malicious TCP-SYN packets generated by HyeaneFE software. This data is given as the training data for the classifier. Accordingly a list of Blacklisted IPs is created.



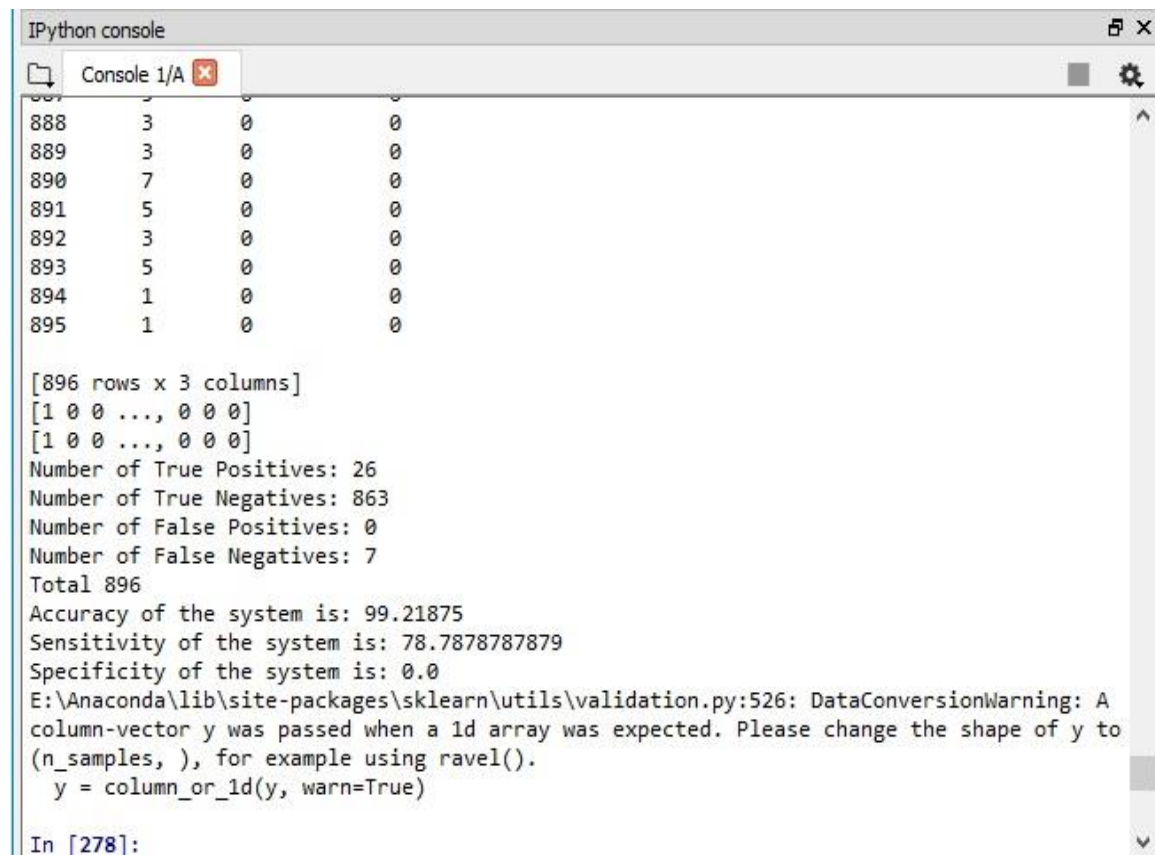
```
IPython console
Console 1/A
699  4  0  0
700  8  0  0
701  7  0  0
702  5  0  0
703  3  0  0
704  5  0  0
705  5  0  0
706  4  0  0
707  9  0  0
708  2  0  0
709  3  0  0
710  5  0  0
711  5  0  0
712 10  0  0
713  6  0  0

[714 rows x 3 columns]
[0 1 1 ..., 0 0 0]
[0 1 1 ..., 0 0 0]
Number of True Positives: 18
Number of True Negatives: 691
Number of False Positives: 0
Number of False Negatives: 5
Total 714
Accuracy of the system is: 99.299719888
Sensitivity of the system is: 78.2608695652
```

Figure 14: Test Case 1

2. In second test case, data was captured using Wireshark having additional malicious TCP-SYN packets for a period of 10 mins. This data was given as a new testing data for the classifier. As there are no already existing IPs in the blacklist the data goes further for classification and the newly identified

attack packets and their corresponding source IPs are added in the existing blacklist.



```
IPython console
Console 1/A
888      3      0      0
889      3      0      0
890      7      0      0
891      5      0      0
892      3      0      0
893      5      0      0
894      1      0      0
895      1      0      0

[896 rows x 3 columns]
[1 0 0 ..., 0 0 0]
[1 0 0 ..., 0 0 0]
Number of True Positives: 26
Number of True Negatives: 863
Number of False Positives: 0
Number of False Negatives: 7
Total 896
Accuracy of the system is: 99.21875
Sensitivity of the system is: 78.7878787879
Specificity of the system is: 0.0
E:\Anaconda\lib\site-packages\sklearn\utils\validation.py:526: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)

In [278]:
```

**Figure 15: Test Case 2**

3. In the third test case, data was captured for 15 mins using Wireshark. It contains packets from the source IPs which are already existing in the blacklist and some another source IPs which are sending malicious packets. The IPs are checked and those present in the blacklist are dropped and not sent to the target destination. The rest of the packets are forwarded to the classifier to detect the attacks and again the blacklist is updated.

```
IPython console
Console 1/A x
863 3 0 0
864 3 0 0
865 7 0 0
866 5 0 0
867 3 0 0
868 5 0 0
869 1 0 0
870 1 0 0

[871 rows x 3 columns]
[0 0 0 ..., 0 0 0]
[0 0 0 ..., 0 0 0]
Number of True Positives: 2
Number of True Negatives: 863
Number of False Positives: 0
Number of False Negatives: 6
Total 871
Accuracy of the system is: 99.3111366246
Sensitivity of the system is: 25.0
Specificity of the system is: 0.0
E:\Anaconda\lib\site-packages\sklearn\utils\validation.py:526: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)

In [279]:
```

Figure 16: Test case 3

# **CHAPTER 12**

## **RESULTS AND CONCLUSION**

## 12. Results and Conclusion

Approach	Naïve Bayes	SVM
Number of packets from a particular source to destination in a given duration.	<b><u>ACCURACY: 99.29%</u></b> TP: 18 TN: 691 FP: 0 FN: 5	<b><u>Accuracy: 99.66%</u></b> TP: 23 TN: 870 FP: 0 FN: 3
Number of packets per second in a given duration.	<b><u>ACCURACY: 98.93%</u></b> TP: 17 TN: 275 FP: 0 FN: 3	<b><u>ACCURACY: 99.32%</u></b> TP: 16 TN: 278 FP: 1 FN: 0

Table No. 2: Results of the designed system

Here,

Accuracy: The accuracy represents the rate of correctly identified results over the entire data used.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} * 100$$

True positives (TP): correctly identified abnormal packets in this research.

False positives (FP): incorrectly identified abnormal packets.

True negatives (TN): correctly identified normal packets.

False negatives (FN): incorrectly identified normal packets.

Hence, according to the analysis,

Source-Destination IP Address Pairing based approach along with the SVM classifier works best for NIDS and NIPS and yields maximum accuracy.

The response time of the system is around 0.715 seconds. The developed system is fast and efficient giving accuracy upto 99%. This can be implemented for large network systems against the DDOS attacks.

In a cloud system this system can be implemented as a preventive measure



# **CHAPTER 13**

## **FUTURE SCOPE**

## 13. Future Scope

The above system is implemented for IPv4 addresses. This system can be developed for a range of IPv6 addresses as well.

The inter-arrival time-based approach can be used for calculating the bursts in a certain duration of the window frame. A prevention system can be developed for the same.

Currently, the system is developed for only one type of DDOS attacks. The system can be efficiently designed for multiple DDOS attacks like Ping-Death, UDP Flooding, HTTP Flooding. Accordingly, the value of the threshold can be automatically updated depending upon the value of threshold for different types of attack.

# **CHAPTER 14**

## **REFERENCES**

## References

- [1] Afreen Jaham , Prof. M. Afshar Alam, “Intrusion Detection Systems based on Artificial Intelligence”, Volume 8 , No. 5 May-June 2017
- [2] Ciprian Dobre, “Internet Traffic Classification based on Flow Statistical Properties with Machine Learnin”, Int.J.Network Mgmt 2015; 00:1-14
- [3] Mahbod Tavallae, “A Detailed Analysis of the KDD Cup99 Dataset”, CISDA 2009
- [4] Adriana-Cristina Enache1, Valentin Sgârciu, “Anomaly Intrusions Detection Based On Support Vector Machines with Bat Algorithm”, 18th International Conference on System Theory, Control and Computing, Sinaia, Romania, October 17-19, 2014
- [5] Miroslav Stampar , “Artificial Intelligence in Network Intrusion Detection.”
- [6] Preeti Aggarwal , Sudhir kumar Sharma, “Analysis of KDD Dataset Attributes Class wise for Intrusion Detection Systems”, Procedia Computer Science 57 (2015) 842-851
- [7] A. Girma, K. Abayomi, and M. Garuba, “The design, data flow architecture, and methodologies for a newly researched comprehensive hybrid model for the detection of DDoS attacks on cloud computing environment,” in Proc. Inf. Technol. Generat. 13th Int. Conf. Inf. Technol., 2016, pp. 377–387.
- [8] B. Cashell, W. D. Jackson, M. Jickling, and B. Webel, The Economic Impact of Cyber-Attacks, document CRS RL32331, Congressional Research Service Documents, Washington, DC, USA, 2004.

[9] Q. Yan, F. R. Yu, Q. Gong, and J. Li, “Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges,” *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 602–622, 1st Quart., 2016.

[10] K. Lee, J. Kim, K. H. Kwon, Y. Han, and S. Kim, “DDoS attack detection method using cluster analysis,” *Expert Syst. Appl.*, vol. 34, no. 3, pp. 1659–1665, 2008