

# **THEORETICAL ANALYSIS OF P2P BOTNET DETECTION IN DDOS ATTACKS USING MACHINE LEARNING TECHNIQUES**

**A PROJECT REPORT**

*Submitted by*

**Bhushan Jadhav** (20MEI10035)  
**Janhavi Kulkarni** (20MEI10053)  
**Abhishek Sanjeev** (20MEI10065)  
**S Gurusaran** (20MEI10066)

*in partial fulfillment for the award of the degree  
of*

**INTEGRATED MASTERS OF TECHNOLOGY**

*in*

**COMPUTER SCIENCE AND ENGINEERING**



**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**VIT BHOPAL UNIVERSITY**

**KOTRIKALAN, SEHORE  
MADHYA PRADESH - 466114**

**DECEMBER 2021**

**VIT BHOPAL UNIVERSITY, KOTRIKALAN, SEHORE  
MADHYA PRADESH – 466114**

**BONAFIDE CERTIFICATE**

Certified that this project report titled “**THEORETICAL ANALYSIS OF P2P  
BOTNET DETECTION IN DDOS ATTACKS USING MACHINE LEARNING  
TECHNIQUES**” is the Bonafide work of Mr. BHUSHAN JADHAV (20MEI10035),  
Ms. JANHAVI KULKARNI (20MEI10053), Mr. ABHISHEK SANJEEV  
(20MEI10065) & Mr. S GURUSARAN (20MEI10066) who carried out the project  
work under my supervision. Certified further that to the best of my knowledge the  
work reported at this time does not form part of any other project/research work based  
on which a degree or award was conferred on an earlier occasion on this or any other  
candidate.

**PROGRAM CHAIR**

Dr. Azath H  
School of Computer Science and Engineering  
VIT BHOPAL UNIVERSITY

**PROJECT GUIDE**

Dr. R Ganeshan  
School of Computer Science and Engineering  
VIT BHOPAL UNIVERSITY

The Project Exhibition I Examination is held on \_\_\_\_\_

## **ACKNOWLEDGEMENT**

Primarily, we would like to thank the Lord Almighty for his presence and immense blessings throughout the project work.

We wish to express our heartfelt gratitude to Dr. Azath Hussain, Head of the Department, School of Computer Science & Engineering for much of his valuable support encouragement in carrying out this work.

We would like to thank our internal guide Dr. R Ganeshan, for continually guiding and actively taking part in our project, giving valuable suggestions to complete the project work.

We would like to thank all the technical and teaching staff of the School of Computer Science & Engineering, who extended directly or indirectly all support.

Last, but not least, we are deeply indebted to our parents who have been the greatest support while we worked day and night for the project to make it a success.

## **LIST OF ABBREVIATIONS**

**BFS - Breadth-first search**  
**BT - Booby trap**  
**C2 - Command-and-control server**  
**DDoS - Distributed denial of service**  
**DFS - Depth-first search**  
**DGA - Domain generation algorithm**  
**DHCP - Dynamic Host Configuration Protocol**  
**DHT - Distributed hash table**  
**DNS - Domain Name System DSI Distributed sensor injection**  
**HTTP - Hypertext Transfer Protocol**  
**HTTPS - Hypertext Transfer Protocol Secure**  
**IDS - Intrusion detection system**  
**IP - Internet Protocol**  
**IPS - Intrusion prevention system**  
**IRC - Internet Relay Chat**  
**ISP - Internet service provider**  
**LCC - Local clustering coefficient**  
**LICA - Less Invasive Crawling Algorithm**  
**MM - Membership maintenance**  
**NAT - Network address translation**  
**NL - Neighbor list**  
**OSN - Online social network**  
**P2P - Peer-to-peer**  
**SCC - Strongly connected component**  
**UDP - User Datagram Protocol**  
**URL - Uniform Resource Locator**

## LIST OF FIGURES

<b>FIGURE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
<b>1</b>	<b>Schematic overview of a botnet</b>	<b>2</b>
<b>2</b>	<b>Comparison of botnet architectures</b>	<b>5</b>
<b>3</b>	<b>System framework</b>	<b>14</b>
<b>4</b>	<b>Decision tree classifier</b>	<b>20</b>
<b>5</b>	<b>Random Forest Classifier</b>	<b>23</b>
<b>6</b>	<b>System architecture of botnet</b>	<b>25</b>

## LIST OF TABLES

<b>TABLE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
<b>1</b>	<b>p2p host detection dataset</b>	<b>13</b>
<b>2</b>	<b>p2p botnet detection dataset</b>	<b>14</b>
<b>3</b>	<b>Selected network traffic features for P2P host detection</b>	<b>18</b>
<b>4</b>	<b>Selected network traffic features for P2P botnet detection</b>	<b>22</b>
<b>5</b>	<b>Analysis of research papers</b>	<b>30</b>

## **ABSTRACT**

Botnet is a network of infected hosts (bots) that works independently under the control of a Bot master (Botherder), which issues commands to bots using command and control (C&C) servers. P2P botnets provide central frameworks for different cyber-crimes including DDoS (Distributed Denial of Service), phishing, password sniffing, etc.

In this work, we analyze PeerClear, a novel approach for detecting P2P botnets using network traffic tracing. It uses a two-step process for identifying P2P bots. First, it identifies all the hosts which are involved in the P2P activity and then from these identified P2P clients, detect hosts who are likely to be engaged in P2P bot activity. For P2P host detection, the system leverages various properties seen in P2P hosts like failed connection attempts, non-DNS connection attempts, etc. For detecting stealthy P2P bots from P2P clients, it uses various features distinctive to bots such as inter-arrival time of packets, duration of the conversation, etc. Our evaluation shows that PeerClear is able to achieve high detection rates of more than 99.6% and low false positive rates of less than 0.28%.

# TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	List of Abbreviations	iii
	List of Figures	iv
	List of Tables	v
	Abstract	vi
1	<b>CHAPTER 1: PROJECT DESCRIPTION AND OUTLINE</b>  1.1 Introduction 1.2 Motivation for the work 1.3 Introduction to the project 1.3.1 Botnet 1.3.2 Overview of botnet 1.3.3 Lifecycle of botnet 1.3.4 P2P network 1.3.5 Types of botnets 1.3.6 Difficulty in finding p2p botnets in real time 1.3.7 How does a PC gets infected and becomes bot? 1.3.8 DDoS attacks and its execution 1.4 Problem statement 1.5 Objective of the work 1.6 Organization of the project 1.7 Summary	1



2	<b>CHAPTER 2: RELATED WORK INVESTIGATION</b>  2.1 Introduction 2.2 Core area of the project 2.3 Existing Approaches 2.3.1 Network signature-based botnet detection 2.3.2 Network behaviour-based botnet detection 2.4 Issues observed in existing approaches 2.5 Summary	9
3	<b>CHAPTER 3: REQUIREMENT ARTEFACTS</b>  3.1 Introduction 3.2 Project requirements 3.2.1 Data requirement 3.2.2 Functions requirement 3.3 Summary	12
4	<b>CHAPTER 4: DESIGN METHODOLOGY AND ITS NOVELTY</b>  4.1 Methodology and goals 4.2 Functional modules design and analysis 4.3 System architecture of botnet 4.4 Summary	16

5	<b>CHAPTER 5: THEORETICAL ANALYSIS OF BOTNET DETECTION TECHNIQUES</b>  5.1 Outline 5.2 Analysis 5.2.1 On the basis of approaches 5.2.2 On the basis of datasets and accuracy 5.3 Advantages and disadvantages 5.4 Summary	27
6	<b>CHAPTER 6: PROJECT OUTCOME AND APPLICABILITY</b>  6.1 Outline 6.2 key implementations outline of the system 6.3 Significant project outcomes 6.4 Project applicability on Real-world applications 6.4 Inference	31
7	<b>CHAPTER 7: CONCLUSIONS AND FUTURE IMPROVEMENTS</b>  7.1 Outline 7.2 Future enhancements 7.3 Inference	34
8	References	

**(THIS PAGE IS LEFT BLANK)**

# **CHAPTER 1: PROJECT DESCRIPTION AND OUTLINE**

## **1.1 Introduction**

This chapter gives us a brief overview about botnets, its several types, P2P network and a brief insight about DDoS attack and its execution.

## **1.2 Motivation for the work**

Since 2020 when we started our semesters, it has been the covid situation which has trembled the economy and has played with the lives of people. All though the interaction with our colleagues was not possible and we have been in the online classes for a year now.

In this period especially in year 2020 and 2021 the arousal of cyber-attacks had increased, as many people we expel from their jobs and people were very carry to get money, therefore to run their life and take care of their family as the phrase goes “An idle mind is the devil's workshop”

## **1.3 Introduction to the project**

### **1.3.1 Botnet**

Botnet derived from the word "Robot" and "Network", is a network of infected hosts or zombies that run automatically and autonomously under the control of an individual or organization referred to as bot master or botherder. To communicate with bots, bot masters use special types of channels referred to as Command and Control (C&C) channels. Some of the prominent malicious attacks that can be ascribed to botnet include DDoS (Distributed Denial of Service), phishing, spam, password stealing, identity theft, ransomware, etc.

### 1.3.2 Overview of botnet

As we discussed, bot masters communicate with the bots using C&C channels. Fig 1.1 shows the schematic overview of the botnet. For simple botnets, having a smaller attack domain, bot master may be directly connected to the bots. However, for bots having larger attack domain such as Zeus, Waledac, etc., bot master is connected to the bots through intermediate hosts known as command-and-control servers (C&C servers). Bot masters switch their C&C servers to prevent identification of bots. This is the reason most of the researchers are interested in tracking C&C servers for finding structures of the botnet. These intermediate computers complicate the process of tracing back bot masters from detected bots.

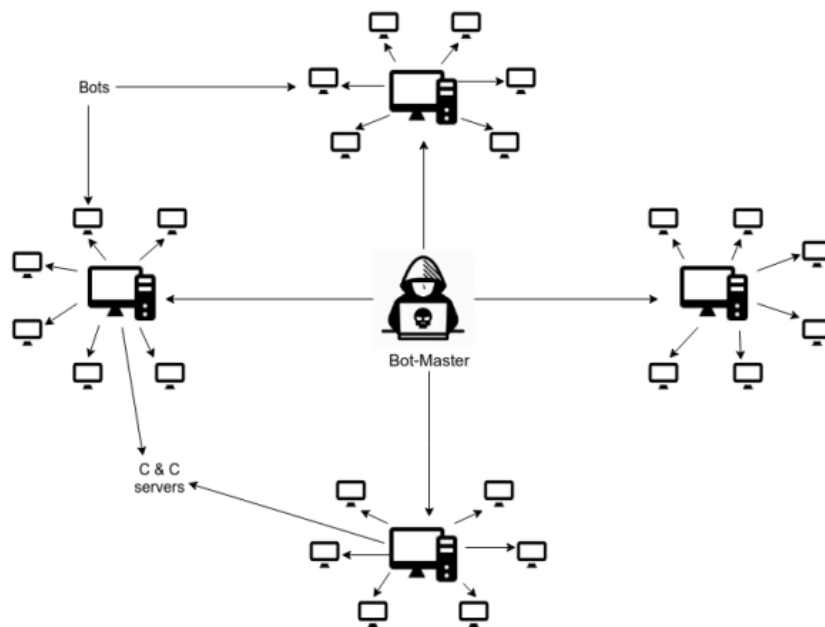


Fig.1: Schematic overview of a botnet.

### 1.3.3 Lifecycle of botnet

Although we investigate the use of a botnet life-cycle as a tool for analyzing this problem here, this concept has already been offered in the literature. In fact, the concept of a botnet life-cycle has already been acknowledged in certain articles (Fiely et al., 2009) (Liu et al., 2009). These studies, on the other hand, just depict part of the processes needed in the typical operation of a botnet, and as a result, there is no consistency in the stages that make up the life-cycle, nor in the interactions between these stages. To put it another way, to our knowledge, there is currently no in-depth research on what these stages are, how they should be classified, or how they differ from one another.

The life-cycle proposed here is a series of stages in a linear order. As a result, the end of the life-cycle, i.e., the attack success, is only reached if the earlier stages have been completed successfully. There are six steps to this life-cycle: conceptualization, recruiting, interaction, marketing, attack execution, and attack success. In addition to the stages of the botnet life-cycle, there are several complementary mechanisms. These measures are usually designed to keep the botnet hidden from security officers.

It is worth noting that each step of the life-cycle just signifies the start of a given process' execution. Although a botnet achieves the attack success stage, new bots will continue to be recruited (recruitment stage) and run at the same time (interaction stage). As a result, every process contained in that stage in our model should be clear that it could be re-executed later.

Once the botnet life-cycle has been created, it is vital to note that any effort to stop a botnet is truly focused on a specific procedure in one of the stages mentioned. Thus, a hypothetical efficient measure that stops any of the cited stages from being implemented is sufficient to prevent botnet success, e.g., if a measure prevents bot infection, recruiting enough soldiers to carry out the attack would be impossible. In conclusion, we argue that halting the execution of only one stage of the botnet's life-cycle makes the botnet ineffective.

### 1.3.4 P2P network

A P2P network is a collection of heterogeneous distributed resources which are connected by a network.

*A distributed network architecture may be called a P2P network if the participants share a part of their resources. The sharing is important to avail of services offered by the network. They are accessible by other peers directly, without passing intermediary entities. The participants of such a network thus resource providers as well as resource requesters.*

The most distinctive difference between client-server networking and P2P networking is the existence of servants (server client). A host that may act both as a server and a client at the same time is called a servant. P2P networks implement some form of virtual overlay network on top of the physical network topology, where the nodes in the overlay form a subset of the nodes in the physical network. Data is still exchanged directly over the underlying TCP/IP network, but at the application level.

Layer peers can communicate with each other directly, via the logical overlay links. To distinguish P2P networks with central entity from those without, P2P networking is split into two subdomains:

- **Pure Peer-to-Peer network:** A distributed network architecture is classified as a Pure P2P network, if it is a P2P network, and additionally, any arbitrarily chosen node can be removed from the network without having any network suffering or any loss of the network service.

- **Hybrid Peer-to-Peer network:** A distributed network architecture is classified as Hybrid P2P network, if it is a P2P network, and additionally, a central entity is necessary to supply parts of the offered network services.

### 1.3.5 Types of botnets

A botnet can be classified according to three architectures as described in the following;

#### Centralized botnets:

Traditional botnets have been seen to use centralized C2s. C2s are often implemented on self-hosted Internet Relay Chat (IRC) servers or hacked web servers. Figure 1.1a depicts the network topology of such botnets. Bots in centralized botnets check C2s on a regular basis for newer updates from bot masters and implement them as soon as they become available. Although centralized C2s are simple to construct and have low communication latency, the architecture supplies a single point of failure, for example, botnet takedown attempts. The decommissioning of the C2 server prevents the entire botnet from retrieving commands or communicating with the bot masters. Furthermore, simply using the server's connection records, defenders may easily count all affected computers.

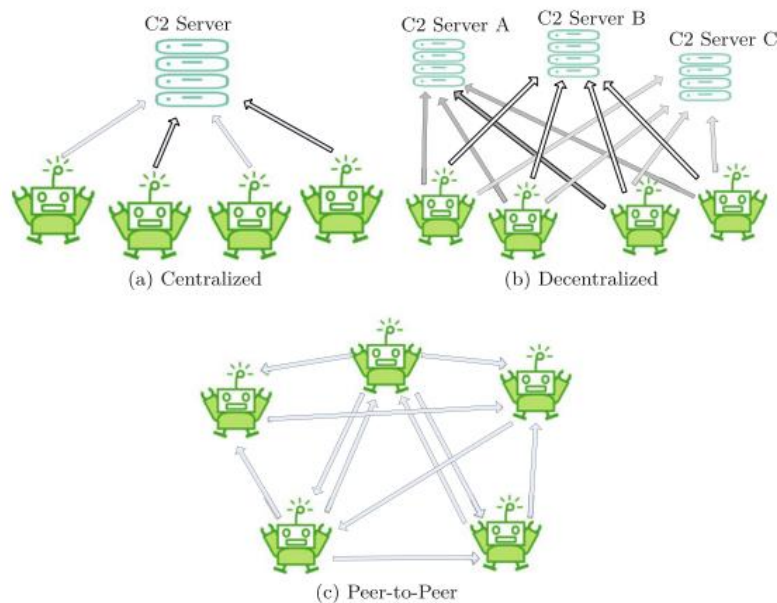


Fig. 2: Comparison of botnet architectures.



### Decentralized botnets:

The weakness of centralized botnets prompted the next generation of botnets to adopt a decentralized architecture that keeps as much of the simplicity and efficiency of a centralized architecture as possible while improving resilience against botnet takedowns by adding redundant C2s, as shown in Fig. 1.1b. Bot masters were seen experimenting with various techniques for implementing the redundant C2 feature. Most bots have many hard-coded C2 addresses that are visited in order if an existing C2 cannot be reached. More advanced solutions made use of the Domain Generation Algorithm (DGA) or fast-fluxing processes. A DGA is a method that produces time-sensitive domain names for C2s across several bots using a common seed, such as the date of the day. In this manner, the bot master can register many domain names that will be contacted by the bots in the future.

Even if some of the C2s are taken down, the bot master can still communicate via future domain names. However, because the DGA is often hard-coded in the bot's binary, reverse-engineering can be used to discover future domain names that will be used in the bots. As a result, defenders can discover and register these domain names to hijack the botnet from the bot masters. Another form of the redundancy approach used in C2s is fast-fluxing Domain Name System (DNS) networks. The IP addresses of many C2s are rapidly cycled in such networks using DNS records for a domain name. In contrast to DGA-based techniques, defenders cannot forecast which IP address will be used to point bots to a C2 in the future if the bot masters control the fluctuation of the IPs. Furthermore, a more advanced variation of fast-fluxing.

### **1.3.6 Difficulty of finding P2P botnets in real time**

For starters, unlike IRC and DNS botnets, botnets do not have a C&C server. Although just one bot is linked to the bot master's server. After that, the attacker's message is given one by one.

As with Internet Relay Chat, if the C&C server goes down, the entire bot network goes down with it. Second, communication takes place amongst the bots in the network. Because each bot has a Membership management mechanism in place to ensure that there is no overlap in list of Neighbors.

Unstructured botnets are more harmful than structured botnets because if we can find the Hash table and find each botnet with its UID, we can take them out one by one. However, with added cases, this is not the case. T will supply an explanation. As a result, in a botnet, every functioning node is referred to as a 'Super peer.'

### **1.3.7 How does a PC get infected and become a bot?**

Computers are most recruited into botnets through malware sent via spam email campaigns – such as those sent out by the spamming botnets. messages can be sent via social media networks and chat apps, which also direct users to malicious websites where malware is downloaded.

Botnets can be created through several different methods. in the case of IOT devices, attackers often take advantage of weak passwords and default credentials that have not been changed.

Since IOT devices are less likely to be updated automatically with the latest software and firmware, it is easier to exploit flaws to gain access to the devices. IOT devices also rarely have antivirus controls, making infection easier and detection of malware much harder.

### **1.3.8 DDoS attacks and its execution**

DDoS is a coordinated attack that uses many hacked hosts to start an attack.

Initially, the attacker seeks out vulnerabilities in one or more networks to install malware programs on several workstations and control them from afar. The attacker then uses these compromised hosts to deliver attack packets to the target machine(s), which are usually outside the original network of infected hosts, without the compromised hosts' knowledge.

The amount of damage done to the victim network is proportional to the intensity of the attack packets and the number of hosts used to launch the attack. A network or a Web server could be affected in a matter of minutes if an attacker can exploit a considerable number of compromised hosts.

## **1.4 Problem statement**

Change is the only constant character which stays everywhere, be it technological trend or change in living your life. Further we are living in a state of transition from paper-work to Digitalization of everything. Threats of distributed denial of service (DDoS) attacks have been increasing day-by-day due to rapid development of computer networks and associated infrastructure, and millions of software applications, large and small, addressing all varieties of tasks. Botnets pose a major threat to network security as they are widely used for many Internet crimes such as DDoS attacks, identity theft, email spamming, and click fraud. Botnet based DDoS attacks are catastrophic to the victim network as they can exhaust both network bandwidth and resources of the victim machine

## **1.5 Objective of work**

We, (Group-14) will be working on the theoretical explanation and analysis of p2p botnets detection techniques which are currently in use. Further, the main aim is to study, analyze and use the machine learning based approaches for conducting correct, real time and efficient p2p botnet detection and explain it with presentations.

## **1.6 Organization of the project**

This Chapter (1) gives a brief introduction to botnets and why botnet detection is important. Chapter 2 gives earlier research done in the field and related work investigation.

Project requirements and proposed approach for P2P botnet detection are discussed in detail and are mentioned in chapter 3 and 4, respectively. Theoretical analysis and comparison is done 7 leading research papers and are explained in chapter 5 while project outcome and applicability are given in Chapter 6. Finally, in the last Chapter (7), conclusions of the thesis are presented and several probable future works are suggested.

## **1.7 Summary**

In this entire chapter, we went over our title, why we chose botnets, and motivation as we succeed in understanding the concepts, we can help the society and organizations run better and minimize any kind of malpractice. Hence, we explained what, when, and how botnets can be and are a threat to an organization. We also discussed DDOS operations that use botnets as their playthings. Furthermore, we tried to describe the concept of our issue statement and the goals we hope to achieve through this project.

# **CHAPTER 2: RELATED WORK INVESTIGATION**

## **2.1 Introduction**

This section includes the existing methods available to detect botnets, which are available in live traffic. This section also includes the core area of project, merits and demerits of the existing method and will conclude with issues seen in those methods.

## **2.2 Core area of the project**

The concept of detecting botnets lies in the domain of cyber security specifically, computer networking. Fundamental concepts of computer networking like ‘peer-to-peer’ networking, packet sniffing was used while working on this project. Since the project is based on theoretical analysis, we have referred multiple research papers

## **2.3 Existing Approaches**

### **2.3.1 Network signature-based botnet detection**

A network signature is a distinct pattern of network traffic, such as connection attempts from a pre-defined IP address. The network signatures of several known botnets are retrieved using this method, and the extracted network signatures are validated against the known botnet signatures during the testing phase. Initially, botnets like Nugache used port 8 as a signature, however this strategy no longer works because today's botnets are stealthier and have complicated communication patterns. Despite this, signature-based methods are simpler to implement, though they cannot detect new botnets unless the signature base is constantly updated.

Signature-based approaches frequently fail to function on encrypted traffic. A signature-based Botnet detection technique uses the signatures of current Botnets for its detection. This method has several advantages, such as exceptionally low false alarm rate, immediate detection, easier to implement and there is better information about the type of detected attack. Signature based detection methods are only able to detect well known botnets. Thus, unknown Botnets cannot be detected by this method. Anomaly-based detection techniques are introduced to overcome this drawback.

### **2.3.2 Network behavior-based botnet detection**

The distinctions between the network behavior of normal traffic and that of botnet traffic are examined in this technique. Because they are not limited to unencrypted botnet traffic, these approaches are more general and efficient than network signature-based botnet identification. A network behavior-based technique can also discover unknown botnets if they exhibit similar behavior to previously known botnets.

The increased false positive rate is an inherent drawback of these techniques. Network behavior might include characteristics such as packet count, bandwidth use, packet timings, and so on. In this strategy, traffic that is unlikely to be part of a botnet is deleted first. The remaining traffic is then used to build a classifier based on the network activity features.

## **2.4 Issues observed in existing approaches**

The pre-computed hashes of existing malware binaries are detected by signature-based botnet detection systems. It can be extremely powerful in monitoring inbound network traffic, and it can also process a high volume of network traffic efficiently. Because of its low false-positive rate, it is the preferred method for some industries. A signature-based botnet detection can be used as an agent, working on a gateway or an end-host to further examine binaries in transfer on-the-fly.

Despite the fact that signature-based botnet detection relies on its database of known threats, which is in itself a clear advantage. Signature-based botnet detection need frequent updates, by which they can be easily violated or modified by polymorphous attacks, zero-day attacks, and similar approaches. Thus, signature-based IDSs are not sufficient to detect botnets. Network-based anomaly detection approach corresponds to the problem of discovering irregular patterns in network traffic that are not expected behavior. It analyzes and collects network traces, including flow statistics and network packets.

## **2.5 Summary**

Since the project relies heavily on computer networks, it was best to brief up network-based botnet detection techniques which already exist. Network signature-based botnet detection and network behavior-based botnet detection techniques were discussed in detail along with few pros and cons, respectively.

## **CHAPTER 3: Requirement artefacts**

### **3. Requirement artefacts**

#### **3.1 Introduction**

This section includes few specific requirements which were needed to analyze detection techniques. The list of requirements also includes datasets which were used for analysis.

#### **3.2 Project requirement**

Network analyzer tools such as Wireshark and Pyshark and Tshark were used for the packet filtering process to capture network packets from live traffic. To include Machine learning based techniques, we research and understood many modules of python like;

- datetime module
- csv module
- random module
- NumPy module
- h5py

##### **3.3.1 Data requirement**

This module includes the data collection phases for training models of the P2P host detection and the P2P botnet detection. We have obtained most of our data (network .pcap files). We have generated some of our data by running the P2P application and the Vinchuca botnet on lab systems and taking the dump of network traffic. Table.1 shows the dataset for P2P host detection and Table 2 shows the dataset for P2P botnet detection.

- **P2P Network Traffic:** Our P2P network traffic is from 11 distinct hosts which run five different P2P applications for several days. The P2P applications include Skype, eMule,  $\mu$ -Torrent, Frostwire, and Vuze. Out of 11 hosts, 9 were running Skype because of its high usage in the real world and other 2 were running other applications at different times. AutoIt [6] tool was used to write the scripts to automate user activities on the host. This includes around 122GB of data. Since  $\mu$ -Torrent is also highly used and above dataset does not contain many samples of the  $\mu$ -torrent application, we have generated 4.5 GB of  $\mu$ -torrent data on lab systems.

- **P2P Botnet Traffic:** We have obtained botnet data mainly from third parties for three different botnets Storm, Waledac, and Zeus. Waledac network traffic was collected before the botnet was taken down by Microsoft. We also ran one P2P botnet named Vinchuca (developed for research purposes) and collected some network traces. We have around 7GB of botnet traffic.

- **Non-P2P Traffic:** Non-P2P Traffic was obtained from the departmental network over five days. Network sniffing tool based on libpcap was used to capture the packets. For DNS packets payload was not truncated, but for other packets, the payload was truncated.

<b>time window</b>	<b>P2P samples</b>	<b>Non-P2P samples</b>
5 min	21387	37394
10 min	10695	18701
20 min	5062	9376

Table 1: P2P host detection dataset



<b>Application</b>	<b>size</b>	<b>Flows</b>	<b>Training</b>	<b>Evaluation</b>
Storm	5.1GB	660970	528776	132194
Zeus	109.8MB	17634	14107	3527
Waledac	1.1GB	43667	34934	8733
Vinchuca	622MB	1423	1138	285
P2P benign	122GB	815659	652527	163132

Table 2: P2P botnet detection dataset

### 3.3.2 Functions requirement

We use network traffic to achieve this without using any bot binary. The IP addresses are taken as input from the user and added to the P2P detection list. Flow-based approach is used to construct an 18-dimensional feature vector (Table 4) for every hour. In the P2P host detection phase, we set the time window to 10 minutes. A 14-dimensional feature vector (FP2P) is extracted for each host separately (Table 3).

If the classifier detects the host feature vector as non-P2P then it is rechecked in subsequent time windows. We use a time window of one hour to detect P2P botnet traffic because of the stealthy nature of such traffic. As soon as any host is identified as a P2B host, the detection phase of the system begins.

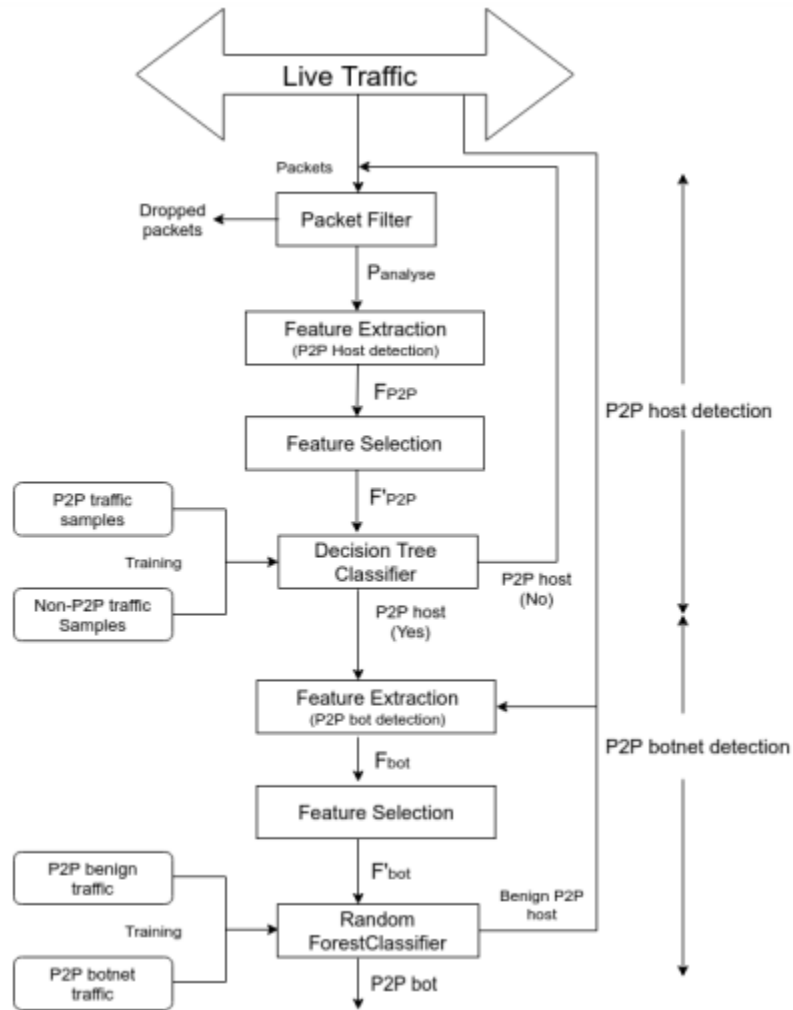


Fig. 3: System framework

The system works as follows;

- Creation of botnets that can be used to execute malicious attacks like DDoS and DNS spoofing.
- Packet filtering of network packets from live traffic using network analyzers.
- Extraction & selection of a few specific network traffic features for hosts.
- Applying Decision Tree classifier to detect p2p hosts.

- Extraction & selection of a few specific network traffic features for botnets.
- Applying Random Forest classifier to detect p2p botnets.

### **3.4 Summary**

This chapter focused on project requirements such as python modules like datetime module, csv module, random module, NumPy module, h5py and specific datasets for p2p network traffic, p2p botnet traffic and non-p2p network traffic. System framework was also explained in detail.

## **CHAPTER 4: DESIGN METHODOLOGY AND ITS NOVELTY**

### **4.1 Methodology and goals**

Our proposed approach works in two phases, the first phase is the P2P Host Detection phase i.e., identifying all the hosts which are involved in the P2P activity. (Host detection using decision tree approach) and the second one is the P2P Botnet Detection phase i.e., detecting P2P bots from identified P2P hosts. (Botnet detection using random forest classifier). The host-based approach is used in P2P host detection phase, in which we extract features that are distinctive to P2P hosts for each host under examination.

In P2P Botnet Detection phase, all P2P types are detected before identifying P2P bots. The stream of packets represents a set of IP packets exchanged between two nodes. It is uniquely identified by the five-tuple set that contains the following information: protocol, source IP address, destination IP address, source port number and destination port number. These packets are generated by various P2P network activities, such as continuation of communication between network members, peer discovery, content request and data transmission.

### 4.1.1 P2P Host Detection

The primary goal of this phase is to identify all hosts that are participating in peer-to-peer (P2P) activity. It consists of the following four modules: (a) Packet Filter, (b) Feature Extraction, (c) Feature selection and (d) Classifier.

- a. Packet Filtering - Packet filtering is a firewall mechanism that monitors outgoing and incoming packets and allows them to pass or halt based on the source and destination Internet Protocol (IP) addresses, protocols, and ports. Packet filtering rule sets are defined by network layer firewalls, which provide very efficient security measures.
- b. Feature extraction - Feature extraction is a broad phrase that refers to strategies for creating combinations of variables to get around these issues while still accurately representing the data. After flow extraction, feature extraction communication information is treated as a two-dimensional image in this study. The abscissa and ordinate in a transmission are the packet sizes and time intervals. The feature can better explain the botnet's flow characteristics, and machine learning can produce good detection results.

Feature	Description
$F_1$	Retransmitted packet count
$F_2$	Destination Diversity
$F_3$	Destination Diversity Ratio
$F_4$	Number of connection attempts made on distinct port
$F_5$	Number of distinct IPs contacted
$F_6$	Reset packets count
$F_7$	Out of order packets count
$F_8$	ICMP destination unreachable packets count
$F_9$	Number of packets sent and received
$F_{10}$	Bytes per packet in forward direction
$F_{11}$	Bytes per packet in backward direction
$F_{12}$	Average retransmitted packets per host count
$F_{13}$	Duplicate ack packets count
$F_{14}$	Total number of control packets (packet without data) sent and received

Table 3: Selected network traffic features for P2P host detection

- c. Feature selection: Feature selection is the process of reducing the number of input variables when developing a predictive model. The number of input variables should be reduced to lower the computational cost of modelling and, in some situations, to increase the model's performance. For the P2P host detection phase, we use 14 features based on P2P host behavior. We will look at a feature reduction strategy that we utilized to reduce the dimensionality of the feature vector in this module. Because we are employing a decision classifier, and the creation of a decision tree is based on information entropy, which is used to calculate information gain, the information gain algorithm is used as a measure of feature reduction. As a result, when using a Decision Tree Classifier, this feature reduction technique is appropriate for feature selection.

- d. Machine Learning-based Classification Techniques - Machine learning techniques are widely used in different research areas because of their outstanding performance. In machine learning, a classifier is an algorithm that automatically sorts or categorizes data into one or more "classes." To understand how given input variables, relate to the class, a classifier uses some training data. We need machine learning methods with a short classification time because we're building a model that operates on real traffic with a high packet rate.

A decision tree is a comprehensive classification method that can be used to explore data and uncover links between a large number of training instances. A Decision Tree is a supervised learning technique that may be used to solve both classification and regression problems, however, it is most commonly employed to solve classification issues. Internal nodes represent dataset attributes, branches represent decision rules, and each leaf node provides the conclusion in this tree-structured classifier.

The Decision Node and the Leaf Node are the two nodes of a Decision tree. Leaf nodes are the output of those decisions and do not contain any more branches, whereas Decision nodes are used to make any decision and have several branches. The decisions or tests are made based on the characteristics of the given dataset. The root node, which is the topmost node in a decision tree, corresponds to the best feature. Both numerical and categorical data can be handled by decision trees.

#### What are the benefits of using Decision Trees?

Machine learning uses a variety of methods, therefore picking the optimal approach for the given dataset and problem is the most important thing to remember while building a machine learning model. The following are two reasons to use the Decision Tree:

1. Decision Trees are designed to mirror human thinking abilities when making decisions, making them simple to comprehend.
2. Because the decision tree has a tree-like form, the rationale behind it is simple to comprehend.

The ID3 technique is used to construct the decision tree in this case. ID3 is the best algorithm for building a decision tree. ID3 (Iterative Dichotomizer-3) is a decision tree learning technique designed by Ross Quinlan that generates a decision tree from a dataset. ID3 is the precursor of the C4.5 algorithm, and it's typically used in machine learning and natural language processing. To construct a decision tree, ID3 employs a top-down greedy approach. In simple terms, the top-down approach means that we build the tree from the top-down, whereas the greedy approach means that we choose the best feature at the time to generate a node at each iteration. ID3 is often only used for classification tasks involving nominal features.

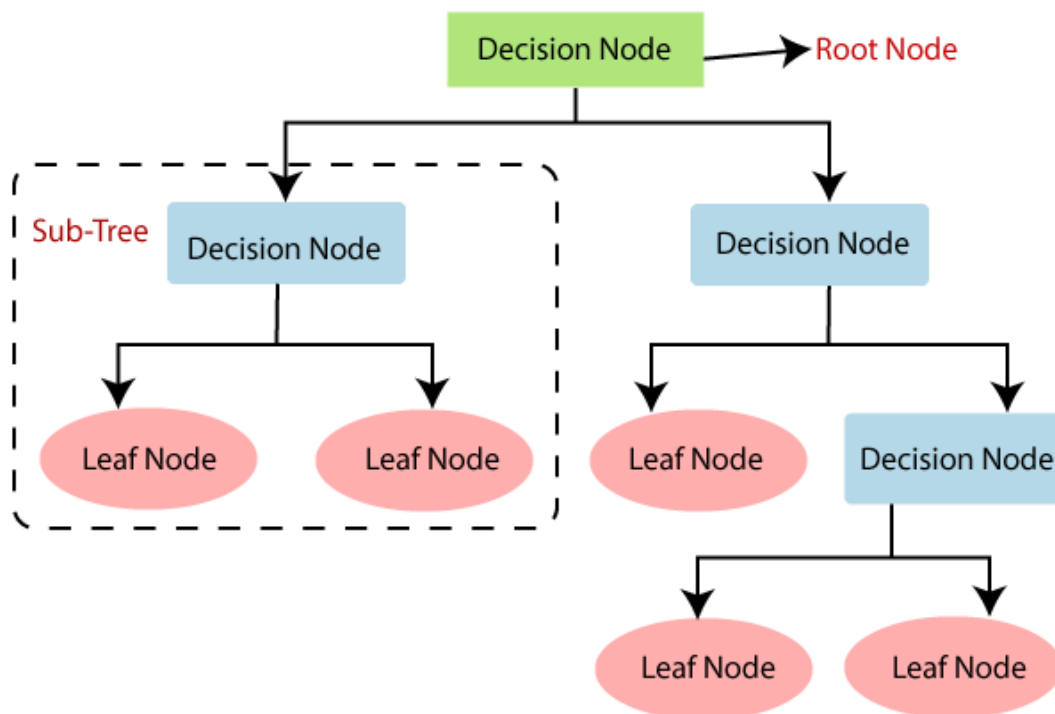


Fig 4: Decision Tree classifier

### **4.1.2 P2P botnet detection**

Following the identification of P2P hosts in the network, the P2P botnet detection step seeks to identify hosts that behave similarly to P2P bots among the identified P2P hosts. The three modules that make up this phase are as follows: Techniques for a) Feature Extraction, b) Feature Selection, and c) Classification

- e. Feature Extraction: The flow-based technique is used for the P2P botnet detection phase. A network flow is a collection of packets sent and received by two hosts. The five tuples source IP, source port, destination IP, destination port, protocol> uniquely identify a network flow. We considered both TCP and UDP flows since different P2P protocols employ different transport layer protocols to distribute files.



Feature	Description
$F_1$	Mean of the inter-arrival time between packets
$F_2$	Number of packets sent in flow
$F_3$	Number of packets received in flow
$F_4$	Number of bytes sent in flow
$F_5$	Number of bytes received in flow
$F_6$	Total data sent and received in flow including headers
$F_7$	Smallest packet in flow
$F_8$	Largest packet in flow
$F_9$	Maximum inter-arrival time between any two packets in flow
$F_{10}$	Minimum inter-arrival time between any two packets in flow
$F_{11}$	Total duration of flow
$F_{12}$	Packet frequency (flow duration/number of packets in flow)
$F_{13}$	Mean inter-time between packets sent in forward direction
$F_{14}$	Mean inter-time between packets sent in backward direction
$F_{15}$	Maximum inter-time between packets sent in forward direction
$F_{16}$	Minimum inter-time between packets sent in forward direction
$F_{17}$	Maximum inter-time between packets sent in backward direction
$F_{18}$	Minimum inter-time between packets sent in backward direction

Table 4: Selected network traffic features for P2P botnet detection

- f. **Feature Selection:** For P2P botnet detection, we extract 18 features based on host access patterns and flow size features. To minimize the dimensionality of the feature vector, we employed an information gain feature selection approach, similar to that used in the P2P host discovery phase. Because the Random Forest Classifier suits a variety of decision trees. The information entropy, which is used to quantify information gain, is utilized to build decision trees. As a result, this feature reduction strategy is suitable for Random Forest Classifier as well.

- g. Machine Learning-based Classification Techniques: The random forest classifier was used to detect P2P botnets. It is a well-known machine learning algorithm that uses the supervised learning method. In machine learning, it can be utilized for both classification and regression problems. It is based on ensemble learning, which is a method of integrating several classifiers to solve a complex problem and increase the model's performance. "Random Forest is a classifier that contains a number of decision trees on various subsets of a given dataset and takes the average to enhance the predicted accuracy of that dataset," according to the name. Instead of relying on a single decision tree, the random forest collects the predictions from each tree and predicts the final output based on the majority votes of predictions. The greater the number of trees in the forest, the more accurate it is and the problem of overfitting is avoided.

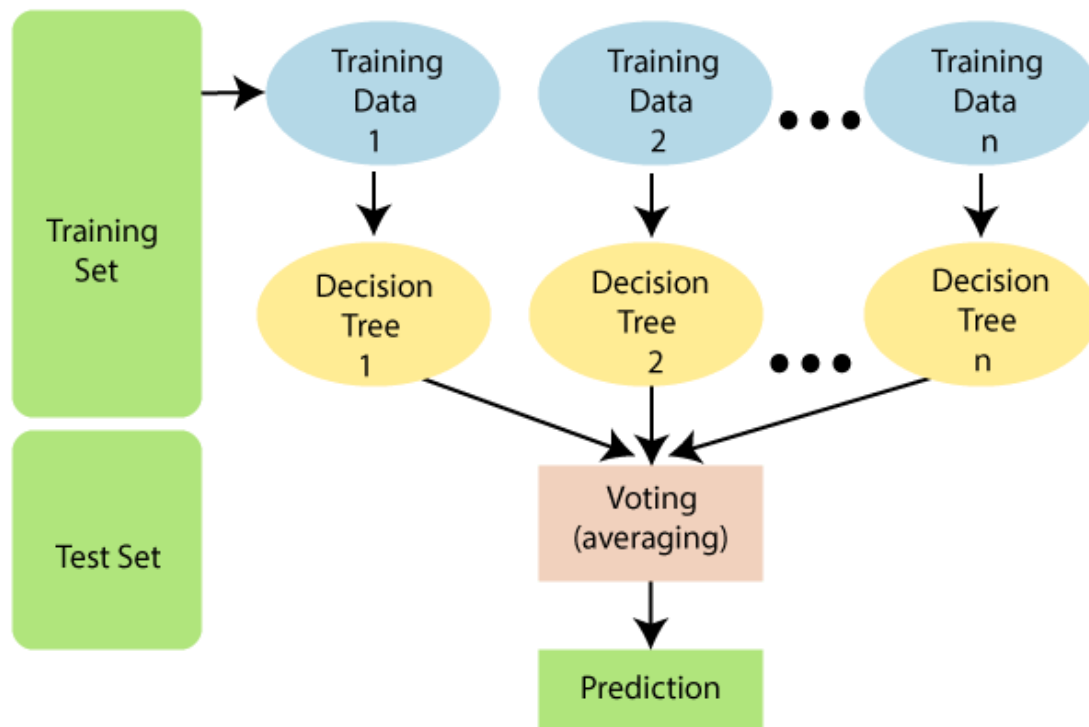


Fig 5: Random Forest Classifier

Why use Random Forest?

Below are a few reasons why we should use the Random Forest algorithm:

- When compared to other algorithms, it takes less time to train.
- It predicts output with high accuracy, and it runs quickly even with a huge dataset.
- When a large portion of the data is missing, it can still maintain accuracy.

## **4.2 Functional modules design and analysis**

### **4.2.1 Working of decision tree algorithm**

The procedure for determining the class of a given dataset in a decision tree starts at the root node of the tree. This algorithm checks the values of the root attribute with the values of the record (actual dataset) attribute and then follows the branch and jumps to the next node based on the comparison. The algorithm compares the attribute value with the other sub-nodes and moves on to the next node. It repeats the process until it reaches the tree's leaf node. The following algorithm can help you understand the entire process:

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using the Attribute Selection Measure (ASM).

Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3.

Continue this process until a stage is reached where you cannot further classify the nodes and call the final node as a leaf node.

### 4.2.2 Working of Random Forest algorithm

The random forest is formed in two phases: the first is to combine N decision trees to build the random forest, and the second is to make predictions for each tree created in the first phase.

The following steps can be used to demonstrate the working process:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

### 4.3 System architectural designs

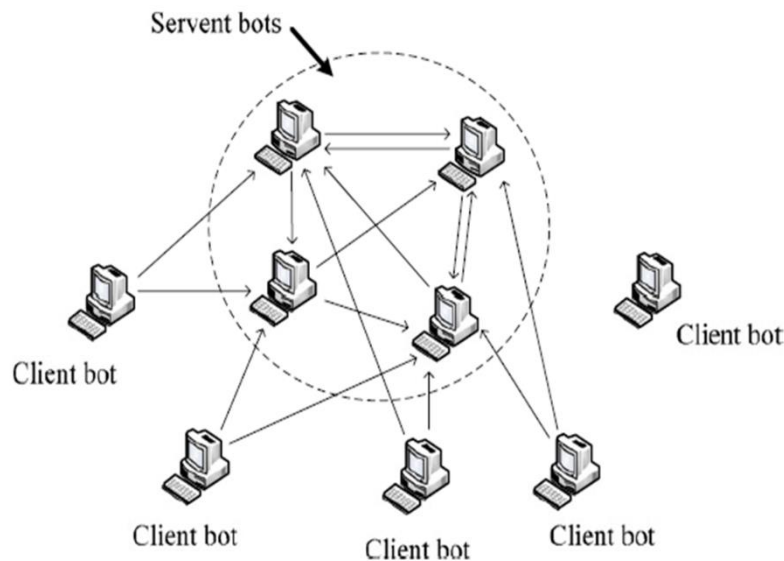


Fig. 6: System architecture of botnet

A client-server approach is used in the "conventional" botnet infrastructure, which involves a Control and Command server that has centralized control over the bots. Using a common communications protocol, such as IRC or HTTP, the C&C server sends automated commands throughout the botnet.

The bot master can use this method of communication to establish dedicated channels between the bots and the C&C, as well as subgroup communications across the bot army. Botnets featuring client-server architecture are easier to set up, boast a well-known infrastructure with many guides and models to learn from and allow the bot master to directly communicate with all bots in a simple two-way session. This design, on the other hand, is reliant on centralized C&C servers, making the botnet easier to take down once identified. Furthermore, the protocols used to establish two-way communication generate more traffic, making victim device exploitation simpler to identify.

A more recent approach to botnets eliminates the need for C&C servers entirely, instead relying on a decentralized peer-to-peer (P2P) architecture. In this model, each bot serves as both client and server. The bots can then transfer information between different devices in the network in this manner. While a botnet's C&C server must possess a list of all the bots in its network, in a P2P model, each peer only possesses a list of its neighboring peers. Using this architecture, botnet traffic is harder to distinguish from legitimate traffic, the bots are harder to find, and the networks are harder to take down as there is no centralized power in the network.

## **4.4 Summary**

In this chapter, we introduced our proposed approach which works in two phases i.e., P2P host detection phase and the P2P Botnet detection phase. Further, we explained P2P host detection and P2P botnet detection in detail. Also, the decision tree classifier and random forest approach were explained along with their working. The system architecture explains the traditional architecture (client-server-based approach) and decentralized peer to peer (P2P) architecture which is the most recent approach to botnets that eliminates the need for C&C servers entirely by relying on it.

# CHAPTER 5: THEORETICAL ANALYSIS OF BOTNET DETECTION TECHNIQUES

## 5.1 Outline

This section is addressing the important part of our report, here we have concisely worked upon the different detections and analysis on which our group will provide the best suitable option for using the Botnet detection technique.

In this report, we are mainly focusing on the **MACHINE LEARNING** techniques because we believe that **AI-and-ML** is a very strong era where we can use it to its full potential. With that, we are continuing to explore more on this topic.

## 5.2 Analysis

### 5.2.1 On the basis of approaches

**PeerClean** is a novel system that detects P2P botnets in real-time using only high-level features extracted from Command & Control network flow traffic. It reliably distinguishes P2P bot-infected hosts from legitimate P2P hosts by jointly considering flow-level traffic statistics and network connection patterns. Instead of working on individual connections or hosts, PeerClean clusters hosts with similar flow traffic statistics into groups. It then extracts the collective and dynamic connection patterns of each group by leveraging a novel dynamic group behavior analysis. PeerClean is able to achieve high detection rates with few false positives.

**PeerDigger** is also a novel real-time system capable of detecting stealthy P2P bots. It first detects all P2P hosts based on several basic properties of Flow records, and then distinguishes P2P bots from benign P2P hosts by analyzing their network behavior patterns. The experimental results demonstrate that this system is able to identify P2P bots with an average TPR of 98.07% and an average FPR of 1.5% within 4 minutes.

**PeerHunter** is based on the community behavior analysis method, which is capable of detecting botnets that communicate via a P2P structure. PeerHunter starts from a P2P hosts detection component. Then, it uses mutual contacts as the main feature to cluster bots into communities. Finally, it uses community behavior analysis to detect potential botnet communities and further identify bot candidates. Through extensive experiments with real and simulated network traces, PeerHunter can achieve a very high detection rate and low false positives.

**PeerRush** is a novel system for the identification of unwanted P2P traffic. Unlike most previous work, PeerRush goes beyond P2P traffic detection, and can accurately categorize the detected P2P traffic and attribute it to specific P2P applications, including malicious applications such as P2P botnets. PeerRush achieves these results without the need for deep packet inspection, and can accurately identify applications that use encrypted P2P traffic. We can detect all the considered types of P2P traffic with up to 99.5% true positives and 0.1% false positives.

**PeerShark** is a novel methodology to detect P2P botnet traffic and differentiate it from benign P2P traffic in a network. Instead of the traditional 5-tuple ‘flow-based’ detection approach, it uses a 2-tuple ‘conversation-based’ approach which is port-oblivious, protocol-oblivious and does not require Deep Packet Inspection. PeerShark could also classify different P2P applications with an accuracy of more than 95%.

**PeerViewer** is a system that automatically classifies P2P malware based on its P2P network behavior. It does not use system-level information, nor does it use flow content signatures during its processing of malware traffic. Indeed, PeerViewer classifies P2P flows for a specific malware into categories where they implement the same P2P signalling activity.

It further builds a footprint that characterizes the P2P network behavior of malware, using the different categories of signalling flows triggered by this malware. To the best of our knowledge, PeerViewer is the first to propose a fully behavioral approach that detects and classifies P2P malware into specific malware families. PeerViewer accurately classifies P2P malware, with a very low false positives rate.

**PeerClear** which is a novel approach for detecting P2P botnets using network traffic tracing. We use a two-step process for identifying P2P bots. First, it identifies all the hosts which are involved in the P2P activity and then from these identified P2P clients, detect hosts who are likely to be engaged in P2P bot activity. For P2P host detection, we leverage various properties seen in P2P hosts like failed connection attempts, non-DNS connection attempts, etc. For detecting stealthy P2P bots from P2P clients, we use various features distinctive to bots such as inter-arrival time of packets, duration of the conversation, etc. Our evaluation shows that PeerClear is able to achieve high detection rates of more than 99.6% and low false-positive rates of less than 0.28%

### **5.2.2 On the Basis of Dataset and accuracy**

The dataset here we have used is downloaded or taken from an organization called **CAIDA** which provides a dataset of network traffic packets for the researchers and University students. [CAIDA](https://www.caida.org/) (<https://www.caida.org/>)

In our project also, we have used the dataset of botnet currently but as per suggestions given by our reviewer of the project we shall try to continue to work on it in a real-time environment with help of our seniors and will develop this system more.



The details of the accuracy are given below in this diagram


Title	Authors	Approach	Dataset	Accuracy
PeerClean [36]	Qiben Yan et. al.	Dynamic Group Behaviour Analysis	73712250 botnet P2P & 401661350 benign P2P flows	98.8%
PeerDigger [16]	Jie He et. al	Flow-based approach	618K Non-P2P, 648K benign P2P & 4874K malicious P2P flows	98.96%
PeerHunter [39]	DI Zhuang & J. Morris Chang	Community Behaviour Analysis	16143828 botnet P2P & 27013777 benign P2P instances	100%
PeerRush [29]	Babak et. al.	Flow-based approach	574 P2P botnet instances & 2041 P2P benign instances	99.5%
PeerShark [25]	Pratik Narang et. al.	Conversation-based approach	50000 malicious P2P & 50000 benign P2P conversations	95%
PeerViewer [20]	Nizar Kheier & Xiao Han	Flow-based approach	158615 P2P botnet & 46782 benign P2P flows	97.6%
PeerClear		Integrated host-based & Flow-based approach	711149 P2P botnet & 815659 benign P2P flows	99.69%

Table 5- Analysis of research papers

### 5.3 Advantages and Disadvantages

The pros are based on the fact that how many datasets and what type of protocol we will be scanning and filtering in the packet capture file .pcap. Cons will be that we cannot really measure the amount of flooding rather we can try to observe the interval of each IP whenever the packet capture is live.

We can determine that if the packet is TCP or UDP accordingly we can measure the data it is carrying and what type of raw data is being sent as it will expose the source as well as the destination IP address of the bot. After that, we will monitor the traffic from a set of IP's which are on our blacklist then try to track their packets. Still this project is in progress therefore we cannot mention a lot of pros and cons currently at this point of time, but in near future, we expect that this will do great similar to IDS systems (Intrusion Detection Systems)

## **5.4 Summary**

So, in this particular chapter, we have covered the main aspect of our project that is a theoretical analysis that has the evaluation of nearly all the p2p botnet detection systems and their author and accuracy. Furthermore, we discussed some pros and cons of our project but like we said before this is still under development. More topics related to our project is discussed in other chapters.

# **CHAPTER 6: PROJECT OUTCOME AND APPLICABILITY**

## **6.1 Outline**

The Project is aimed to develop a mechanism or model which can be used to prevent the DDoS attack factor i.e. Botnet and especially the p2p botnets because of their difficulty in tracing and higher resilience.

It is expected that our project will still be under process for more variation and covering of more different approaches and behaviors to the machine learning algorithm.

## **6.2 Key implementations outline of the system**

The key implementation of the system are as follows:

- Packet filtering of network packets from live traffic using network analyzers.
- Extraction & selection of a few specific network traffic features for hosts.
- Applying Decision Tree classifier to detect p2p hosts.
- Extraction & selection of a few specific network traffic features for botnets.
- Applying Random Forest classifier to detect p2p botnets.

## **6.3 Significant project outcomes**

1. The project aims to discover which type of botnet detection is better to use at different organization and situations
2. This can help the researchers to understand what kind of classifier programs and more specifically it will focus on what machine learning technique is being used in the efficient detection of botnets.
3. The data-results are taken from a dataset but we hope it should perform well in real-world-applications.
4. This project helped our group members to explore the most out of an important cyber field of research that is botnet, as in near future for sure the countries can start a cyberwarfare for which this field is to be explored more.

## **6.4 Project applicability on Real-world applications**

The project is applicable in the scenario of places where the IT sector -lower the chances of risk management and increases the stability of the organization by improving security. It is clear that the world is moving towards the digitization of everything.

There will be more and more cyber threats occurring therefore the path for this is likely to not end but improve the sector to build more robust devices and algorithms in the future time to enhance our security and become sustainable in nature. Therefore, this project is just a beginning and a good beginning for this long run marathon for cyber security based on the organization's security to strengthen the security.

## **6.5 Inference**

Nowadays, botnets are more diverse, resilient, widespread, and utilized in many cyber-attacks. Therefore, there is a pressing need for a better botnet detection method. This study presents a hybrid rule-based approach for detecting DNS-based botnet. New features are proposed and used to form new rules. Some rules have been extracted using Decision tree and Random Forest classifier machine learning algorithms are used to detect P2P-based botnets in the datasets.

Finally, this research opens avenues for future research in the following aspects:

- Adapting the proposed rules to detect blockchain-based P2Pbotnets,
- Hybridizing the resulted rules with other approaches, such as the signature-based approach, could improve DNS-based botnet detection accuracy further,
- Investigating and study the impact of encrypted P2P traffic, such as UDP and TDP flows.
- Scaling behavior analysis to better understand the applicability of the proposed approach in the real world.

## CHAPTER 7: CONCLUSIONS AND FUTURE IMPROVEMENTS

### 7.1 Outline

In this chapter, we are going to discuss the limitations of our proposed approach. We have listed below the improvements that can be made to improve or extend the work of this project.

### 7.2 Future Enhancements

There are several improvements that can be made to improve or extend the work of this project, some of which are listed below:

- **Train the model in real-time traffic:** Feature extraction from static pcap data is utilized to train the current models in the system. The system can be enhanced to train models on live network data and to understand the pattern of regular traffic behavior in order to detect botnet network traffic.
- **Different feature selection algorithm:** Although, in our study, we employed feature selection to minimize the dimensionality of the feature vector without compromising performance. To produce a superior feature subset, another feature selection algorithm might be utilized.
- **More informative features:** PeerClear chooses 14-dimension feature vector for P2P host detection and an 18-dimensional feature vector for P2P botnet detection based on prior work and our studies. By studying network traffic from P2P applications and P2P botnets in-depth, more informative features such as mutual contact ratio can be discovered.

## 7.3 Inference

Botnets are one of the biggest threats to the Internet today, and they are linked to most forms of Internet crime. Most spam, DDoS attacks, spyware, click fraud, and other attacks originate from botnets and the shadowy organizations behind them. P2P botnets are the most dangerous botnets among the known botnets.

A number of botnet countermeasures exist, but most are focused on bot detection and removal at the host and network level. Some approaches exist for Internet-wide detection and disruption of entire botnets, but we still lack effective techniques for combating the root of the problem: the bot masters who conceal their identities and locations behind chains of stepping-stone proxies.

The three biggest challenges in bot master traceback are stepping stones, encryption, and the low traffic volume. Even if these problems can be solved with a technical solution, the trace must be able to continue beyond the reach of the Internet.

We have developed an approach for detecting P2P botnets. The approach is divided into two stages. In the first phase, we use a host-based approach to detect P2P hosts in the network and extract host-based features based on properties displayed by P2P hosts such as failed connections, non-DNS connections, and so on. For each host, the features are extracted for a 10-minute time span. We discover P2P bots from the identified P2P hosts in the second phase. We employed a flow-based approach for bot detection, extracting flow-based features based on host access pattern and data exchange pattern for a window of 1 hour.

## **References**

<https://en.wikipedia.org/wiki/Botnet>

<http://www.cerc.iiitd.ac.in/spsymp15/papers/11.pdf>

<https://www.irjet.net/archives/V7/i7/IRJET-V7I7891.pdf>

[GeeksforGeeks | A computer science portal for geeks](#)

A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks - PubMed (nih.gov)

[1] Stevanovic, Matija, and Jens Myrup Pedersen. "Machine learning for identifying botnet network traffic." (2013).

[2] Maryam Feily , Alireza Shahrestani , Sureswaran Ramadass, A Survey of Botnet and Botnet Detection, Proceedings of the 2009 Third International Conference on Emerging Security Information, Systems and Technologies, p.268-273, June 18-23, 2009

[3] Thomas S. Hyslip, Jason M. Pittman, A Survey of Botnet Detection Techniques by Command-and-Control Infrastructure. 2015.

[4] M. Bailey, E. Cooke, F. Jahanian, Y. Xu, and M. Karir, A Survey of Botnet Technology and Defenses, Cybersecurity Applications & Technology Conference for Homeland Security, IEEE Computer Society, L.A

[5] Wang, T. S., Lin, H. T., Cheng, W. T., & Chen, C. Y. (2017, Jan.). DBod: Clustering and detecting DGA-based botnets using DNS traffic analysis. Comput. Secur., 64, 1-15

[6] P. BURGHOUWT, Detection of botnet command and control traffic in Enterprise networks, PhD dissertation, The Hague University of applied Sciences, 2015.

[7] D. C., Peer-to-peer botnet detection using netflow, (2014).

[8] H. CHOI, H. LEE, AND H. KIM, Botnet detection by monitoring group activities in dns traffic, 7th IEEE International Conference on Computer and Information Technology, (2007).

[9] D. DITTRICH AND S. DIETRICH, P2p as botnet command and control: a deeper insight, Computer Science Department Stevens Institute of Technology, (2008).

**(THIS PAGE IS LEFT BLANK)**