ANALYSIS OF BOTNET CLASSIFICATION AND DETECTION BASED ON C&C CHANNEL

UNIVERSITY OF TURKU
Faculty of Science and Engineering
MDP in Information Security and Cryptography
Networked Systems Security
November 2018
Anup G.C.

Supervisors:
Antti Hakkala
Petri Sainio

UNIVERSITY OF TURKU
Department of Future Technologies

ANUP G.C.: Analysis of botnet classification and detection based on C&C channel

Master of Science in Technology Thesis
Networked Systems Security
November 2018

Botnet is a serious threat to cyber-security. Botnet is a robot that can enter the computer and perform DDoS attacks through attacker's command. Botnets are designed to extract confidential information from network channels such as LAN, Peer or Internet. They perform on hacker's intention through Command & Control(C&C) where attacker can control the whole network and can clinch illegal activities such as identity theft, unauthorized logins and money transactions. Thus, for security reason, it is very important to understand botnet behavior and go through its countermeasures. This thesis draws together the main ideas of network anomaly, botnet behavior, taxonomy of botnet, famous botnet attacks and detections processes.

Based on network protocols, botnets are mainly 3 types: IRC, HTTP, and P2P botnet. All 3 botnet's behavior, vulnerability, and detection processes with examples are explained individually in upcoming chapters. Meanwhile saying shortly, IRC Botnet refers to early botnets targeting chat and messaging applications, HTTP Botnet targets internet browsing/domains and P2P Botnet targets peer network i.e. decentralized servers. Each Botnet's design, target, infecting and spreading mechanism can be different from each other. For an instance, IRC Botnet is targeted for small environment attacks where HTTP and P2P are for huge network traffic. Furthermore, detection techniques and algorithms filtration processes are also different among each of them. Based on these individual botnet's behavior, many research papers have analyzed numerous botnet detection techniques such as graph-based structure, clustering algorithm and so on. Thus, this thesis also analyzes popular detection mechanisms, C&C channels, Botnet working patterns, recorded datasets, results and false positive rates of bots prominently found in IRC, HTTP and P2P.

Research area covers C&C channels, botnet behavior, domain browsing, IRC, algorithms, intrusion and detection, network and peer, security and test results. Research articles are conducted from scientific books through online source and University of Turku library.

Keywords: Botnet detection, Cyber Security, Algorithm, Clustering, Command and Control, IRC, HTTP, P2P

Acknowledgments

I would like to express my sincere gratitude towards my family on supporting me for abroad studies financially as well as motivating me towards achieving my goals and success.

This Thesis is developed out of a series of lectures, materials, meeting, and papers provided by the University of Turku. Thus, I am grateful towards University Library and Network for providing all the materials required for the Research.

I would like to make a special note to my Tutor (Antti Hakkala), Lecturer, Master's Degree Program in Information Security and Cryptography on supervising contents, references, research process, suggestions, and supports. Regarding my difficult situations, proper encouragement led me for the motivational environment.

I would also like to thank (Seppo Virtanen) Head/Lecturer, Master's Degree Program in Information Security and Cryptography for meetings, flexible tasks, and support on any issues related to my Degree.

I want to include (Apil Karki) Graduate, Master's Degree Program in Information Security and Cryptography, for giving good support on Thesis topic and research processes. And to all my friends and classmates, living in Turku who has directly or indirectly helped me cross this venture.

Table of Contents

Abbreviations and Acronyms

ADT	Anomaly Detection Technique
AOL	America Online
ART	Activity Response Detection
C&C	Command and Control
DDoS	Distributed Denial of Service
DNS	Domain Name System
DPR	Degree of Periodic Repeatability
DTRAB	Does This Ring A Bell
FP	Foreign Policy
HTTP	Hypertext Transfer Protocol
HIDS	Host-based Intrusion Detection System
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IRC	Internet Relay Chat
IRC PRIVMSG	Internet Relay Chat Private Message
ISP	Internet Service Provider
MRP	Message Response Detection
NIDS	Network-Based Intrusion Detection System
NIST	National Institute of Standards and Technology
OOB	Out of Band
P2P	Peer-to-Peer
SMTP	Simple Mail Transfer Protocol
SOM	Self-Organizing Machines
SVM	Support Vector Machines
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UI	User Interface
UID	User Identification Detection
UPA	United Progressive Alliance

1 Introduction

Botnet is mostly used for illegal activities such as identity theft, unauthorized logins and DDoS attacks to Network (LAN, Peer-to-Peer & Internet) or protocol (IRC, HTTP & P2P). For the process, attacker first designs bot structure which is based on attack intentions and durations. We will go through possible attack intentions on individual botnet types in coming chapters. Next, on completion of bot structure, bot is formed and is deployed to the targeted network. After that, mutual communication between network and attacker's computer is established where the infected network is known as botnet. Finally, attacker is able to perform any intended actions such as identity theft, unauthorized logins and confidential information retrieval.

Botnet is also called a bot which refers to robot and operates through Command and Control (C&C) channel. C&C is a method to deploy command on machine and control through it. In botnet, C&C is used by programming code to establish communication between bot and network as well to operate any illegal activities as mentioned above. In C&C process, attackers sending commands to bots are known as "Botmaster". Botmasters are often called for "Botherder" or "Botcontroller too" [1]. Botmasters can send commands from a remote location without prior knowledge of the network administrator and can infect multiple computers at a time. Such infected computers are known as "Zombie Computers". When infected, zombie computers starts to operate through Botmaster commands and becomes bot [2].

A bot is designed initially through object-oriented programming such as Java and C#. Designed structure is then deployed to targeted network through malware. While designing, it is taken into consideration that bot follows botmaster's commands and maintains the connection. On deployment to network, bot starts infecting the computer and establishes third-party control (i.e. C&C). During the process, malware can be passed through different means like an external drive, online hacking, human elements, driver installation and many more. When a bot communication is established, it performs activities according to botmasters command for intended purposes, especially illegal activities such as email spam, identity theft, DDoS, monitoring browser history, access files or bank card details and spreading malware. Furthermore, Botnet can expand over every node of the network and can infect every single computer inside the network. This way an entire network can be under control of attacker through Botnet and C&C [3]. Thus, Botnet is a serious cyber-security threat today affecting numerous scientific fields, financial fields, education, healthcare, finance and law enforcement fields.

While briefing about Botnets behavior, each Botnet's nature and intrusion processes are different based on network protocols. In this thesis I have decided to cover mainly 3 protocols (IRC, HTTP, and P2P) where individual Botnet examples, behavior and detection processes are explained thoroughly in individual chapters. In this thesis, we begin at Chapter 2 with botnet overview, botnet lifecycle about initial infection, bot's purpose, and possible threats.

Next in Chapter 3, Anomaly Detection Technique (ADT) is explained about cyber-security and based models. After that, detailed information's on individual Botnet's behavior with possible threats, detection methods, algorithms, results and evaluation are presented in Chapter 4, 5 and 6 respectively. Furthermore, existing theories and results like Botsniffer and graph-based C&C algorithms over local servers and Peer networks are also explained under the corresponding topics.

Finally, in Chapter 7, my comparative analysis over individual Botnet (IRC, HTTP, and P2P), botnet's behavior, vulnerability, similarities, detection process, experimental evaluation and error rates are shown.

2 Botnets

The Internet is widely used all over the world for communication and data transfer. Internet has also changed means of security surveillance, money transactions through online payments, operating devices (IoT), travel and appointment bookings and many more. Whenever people face difficulties with something like travelling to new city or curiosity on information, they can simply turn on their internet device such as phone, tablet and computers to check for solution. So far, internet has made human daily life as possibly as it can in many ways and has always been with us as a closest friend. With all these great features also come online security and threats where some unknown user can be monitoring the device and activities to retrieve any personal information known for intrusion. Intrusion is any unauthorized activities happening to internet like identity theft, unauthorized logins, confidential file retrieval and unauthorized transactions without prior permission of the owner. This has been a huge cyber security threat today and are happening through various means like viruses, malware, phishing, spamming and many more. Among all these different threats, I find Botnet as the biggest threat because it can operate through C&C channel, more destructive in nature and can spread to millions in number. Furthermore, Botnet can act active or silent based on attack duration and botmasters commands. It can also update and maintain itself as per the received command making it strong and effective. They are also hard to be detected by antivirus or any simple IDs making it vulnerable [4].

The primary motive for creating Bot is to achieve financial gain or access unauthorized information. As per the motive, a bot is designed to be installed in a client machine i.e. computer device through external means like hard drive, phishing, spam or malware and after installation can be controlled by botmaster’s command establishing mutual communication. When communication is established, Botmaster can control the device and execute any intended actions such as DDoS attacks. On this process, Botmaster's computer is far away from the targeted place and being remote he can monitor targeted addresses. Botmaster often hacks another IP's and use them as its own to prevent from ID as well as to make the detection task difficult [4]. More information about Botmaster, Bot's lifecycle, purpose, and residing machine are covered on this chapter.

2.1 Botnet lifecycle

Botnet lifecycle in this section represents botnet working mechanism from initial infection on the computer to server destruction as shown in Figure 1. While entering the network, bot first hijacks single node of the network and spreads throughout the available devices. For hijacking single node, botmaster need to install bot-code via web-links, phishing emails and fake advertises. Depending on the protocol platform (IRC, HTTP or P2P), bot-code is installed, and bot is formed [1].

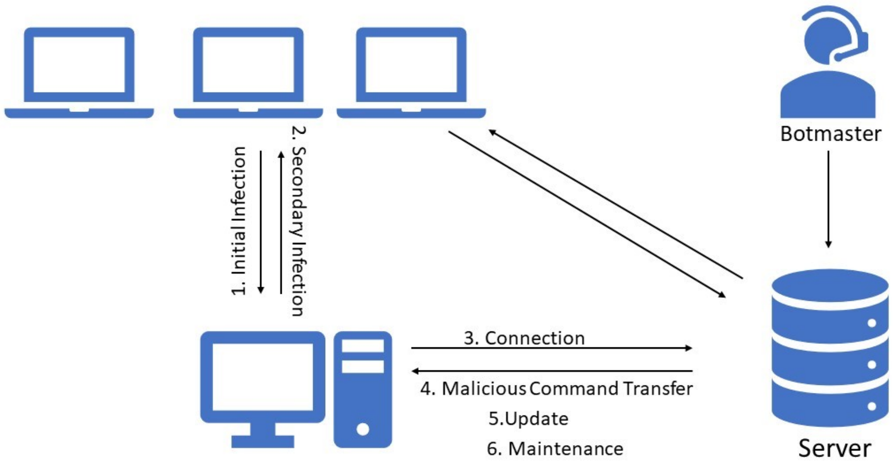


Figure 1: Infected Botnet Cycle [1]

Figure 1 shows Botnet lifecycle process starting with initial infection on the computer. On this process, botmaster uses remote IP address to deploy bot into targeted network's computer and initial infection happens. After that, commands are returned, and mutual communication is established as shown in step 2. Next, botmaster uses C&C channel on the infected machine for mutual communication and bots are expanded through every node of the network and the main server too. Once the communication is established on the main server, botmaster can perform any malicious activities like information retrieve and identity theft as shown in step 3-4. Furthermore, botmaster can update and provide maintenance to bots as per the need through C&C. Finally, when the DDoS attacks are performed, installed bots can be removed or can be inside the network un touched. This is a typical Botnet Lifecycle and may vary based on protocol and network (IRC, HTTP, and P2P). They are explained separately in upcoming chapters with examples [1].

2.2 Bots Purpose

Similar to many other cybercrimes, bots' purpose is also to monitor user's activity, unauthorized logins, identity theft, and data retrieval. With received information, attacker can transfer funds, open accounts, and gather any needed information. As bot start to spread, they can also collect sensitive information like keys user presses on the keyboard, monitor screen information, health information and retrieve web browsing history. With such information's, attacker can simply extract victim's behavior, emotions, part of interests or any hidden secrets. It is close enough to read people's mind or personal diary [5]. Key point is that, once bot is installed in the computer, it has many open spaces for attacks and its purpose is determined from botmaster's intention. Thus, all user should have proper information on botnet and proper detection techniques should be applied to their devices. Botnet detection techniques are explained according to targeted network on upcoming chapters separately. But before that, it is important to know about bot residing machines and Anomaly Detection Technique explained in the next sections.

2.3 Potential Targets

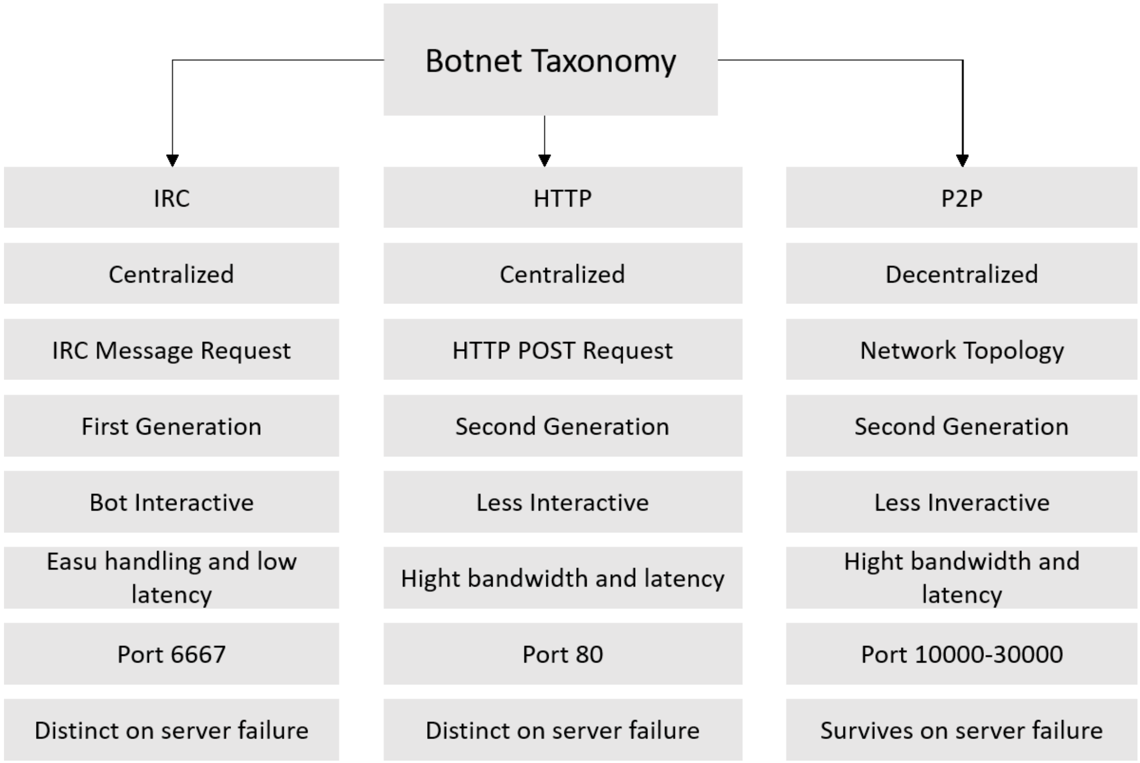
Approximately 3 billion people, i.e. half of the world's population, are Internet users [4]. These users have several communicating devices connected to the Internet every day. Users can be tricked to perform activities such as downloading malware and visiting malicious content. While visiting those contents, bots are also installed on the device without prior knowledge of the user and the device becomes bot residing machine or zombie computers. All installed bots inside the computer stays quiet until commands are applied or discovered by the bot detector. Thus, the user won't even notice when bots are on their devices and unauthorized data transfer might occur leading to serious threats. This way without prior knowledge of the user, millions of bots are installed to devices all over the world. Furthermore, Cisco estimates, by 2020, internet devices will be 50 billion and with potential chances of bot-residing machine, bots will be spreading in huge size [6].

On the other hand, IoT devices are on high demands these days where modern devices like GPS devices, kitchen appliances, home-security, heating-system, and air-flow are monitored throughout the internet. Imagine someone is monitoring all these basic activities far away from user's location through bot and all cameras, entered passwords, etc. are recorded. Depending on the attacker's intention, he can accomplish anything needed including health concerns and private information leading to blackmailing as well as huge financial destruction [4]. Thus, proper detection plan is needed for all electronic devices operating through internet too. Meanwhile, destruction level and vulnerability in this process also depends on the type of Botnet and are explained in Botnet Taxonomy section below.

2.4 Botnet Taxonomy

Botnet was first noticed at instant messaging application in 1999 as "Pretty Park" [7]. By that time, bots used to be less vulnerable and could affect only one specific device at a time. But then, bot started to evolve with more stability and robustness. Slowly, it started affecting a huge area of browsing environment and multiple servers. Destructions started to count in billions of dollars. Thus, it is very important to understand Botnet nature and Taxonomy where Botnet Taxonomy is based on Botnet behavior to their server-side application, destructive nature and evolution.

IRC, HTTP, and Peer-to-Peer Botnets are mainly 3 types of Botnet that falls under Botnet Taxonomy and are counted as the biggest threat to cyber-security. They are categorized through centralized, distributed and peer network channel which covers instant messaging application, browsing application, and server-side environment. We are going to discuss on individual Botnet Taxonomy working mechanism, destructive nature, examples, attack methods, and detection processes in upcoming chapters separately where Figure 2 below gives rough diagram on the framework of Botnet Taxonomy [3].



IRC	HTTP	P2P
Centralized	Centralized	Decentralized
IRC Message Request	HTTP POST Request	Network Topology
First Generation	Second Generation	Second Generation
Bot Interactive	Less Interactive	Less Inveractive
Easu handling and low latency	Hight bandwidth and latency	Hight bandwidth and latency
Port 6667	Port 80	Port 10000-30000
Distinct on server failure	Distinct on server failure	Survives on server failure

Figure 2: Botnet Taxonomy [3]

3 Anomaly Detection Technique

Anomaly refers to abnormal data flow behavior for unauthorized logins, intrusions, or unusual transactions. In this chapter, we are going to consider Botnet as our anomaly and focus mainly on its detection processes. For the detection process, Anomaly Detection Technique (ADT) is widely used technique, where detection principles and algorithms are used to observe anomalies. Detection technique contains the layer of filters process to separate doubtful traffic from normal traffic and algorithms are implemented accordingly detecting the bug [8].

Depending on the server-side platform, ADT is categorized into Host-based Intrusion Detection (HID) and Network based Intrusion Detection [9]. HID can perform only one specific server at a time and secures the system by monitoring over incoming as well as outgoing traffic. During the process, unusual behaviors are scanned, and alert messages are delivered to the administration. On the other hand, Network-based ID are targeted for peer networks and are meant to cover whole available servers. Network-based ID are installed on prime points of the incoming and outgoing traffic to detect unusual behaviors. Firewall is one good example of Network based ID [10].

ADT has been widely researched and discussed in literature. One example is DTRAB, introduced by Fadlullah et al. [11] and updated at [12], which works by detecting anomalies in the protocol. Furthermore, DTRAB can also be used to construct a defensive platform for tracing back the attacker [11]. Another example is the framework for observing 5G networks proposed by Yan, Zhang & Vasilakos [13]. It is targeted mainly for future network generations which will be highly virtualized and software-defined. Furthermore, Shu [14] has also presented a qualitative comparison of particular advantages and disadvantages on a Software Defined Network(SDN). This will help to understand data forwarding layer, control layer and application layer to apply required countermeasures.

Although lots of theories and experiments have been conducted, ADT provided for current generation and next evolving generation might fall short [10]. It is because botnets can easily adapt to encrypted commands and do not cover rule-based architecture. Such way, botmasters can examine existing countermeasures and modify their bots preventing detections. Thus, a long-term vision is needed for botnet detection and network security. We will discuss anomaly architectures, methods for IDS, and evaluation of results.

3.1 Anomaly Detection Architecture

ADT based architecture is carried out mainly from three factors i.e. sensors, analyzers, and UI. Architecture is designed in a way that firstly Network data is collected through software components [10] and are passed through the sensor. After that, collected data is passed to an analyzer where algorithms are implemented to check for the intrusion behavior. Software and algorithms used in this analyzing process may vary depending on the network protocol. When the intrusion is detected, an alert message is sent to the system. Administration department will be able to see the alert through UI and are able to react accordingly. Figure 3 shows how data is passed through the FW, Sensor, Analyzer, and UI. [10]

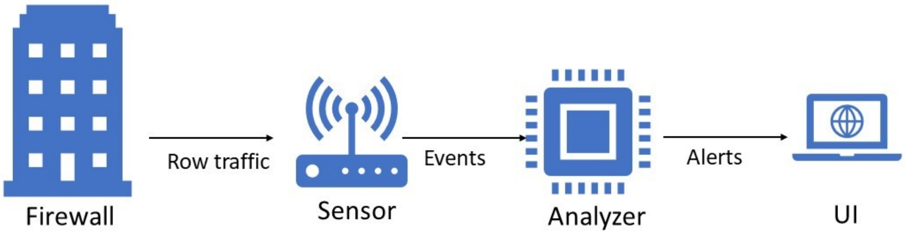


Figure 3: ADT components [10]

Implementing anomaly architecture and going through its supervision are both important roles from the security point of view [6]. Algorithms and software are one part of security as explained above but using them on a specific location is also very important for significant results. Figure 4 below shows an implementation of an architecture for Network-based ID where the attacker creates multiple Bots through C&C and tries to enter the system, but Firewall prevents them from entering and protecting the whole network. The firewall in Figure 4 is kept in a way that secures entire perimeter of the victim's network and monitors doubtful traffic. Such way any suspicious incoming traffic like bots are blocked and are passed through security department by alerts.

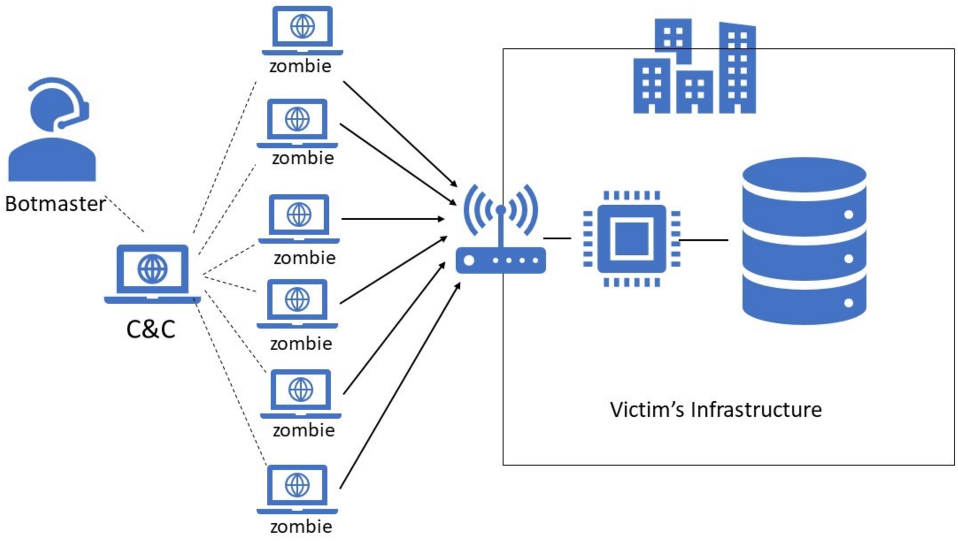


Figure 4: Network-based IPS [6]

After Firewall situations are taken into proper consideration, comes supervision on dataset and labeling. Based on normal or abnormal behavior, labeling is done to all collected data sets. This helps to understand the nature of Bots and protect the network in future from similar intrusions too. Methods for labeling are supervised, semi-supervised and unsupervised based on the destructive nature of anomalies. Some popular examples containing datasets labeling are K-nearest neighbor, Multi-layer perceptron, Support Vector Machines (SVM), k-means clustering, self-organizing maps (SOM) and so on [10]. All above examples have their own data sets to compare the incoming/outgoing traffic and separate the doubtful traffics. Thus, such labeling are used in many detection applications to compare the nature of intrusions too. We will be discussing few detection applications like Botsniffer, graph-based clustering algorithm and neutral classification analysis in each botnet taxonomy section through upcoming chapters starting with popular ADT methods in next section.

3.2 Methods for ADT

Several methods are introduced for ADT among which Statistical based, Cognition-based, User-Identification based, and Machine Learning based are popular methods used for Botnet Detection. Figure 5 gives a tree-structure of these Detection-based mechanisms and are shortly described in individual sub sections below.

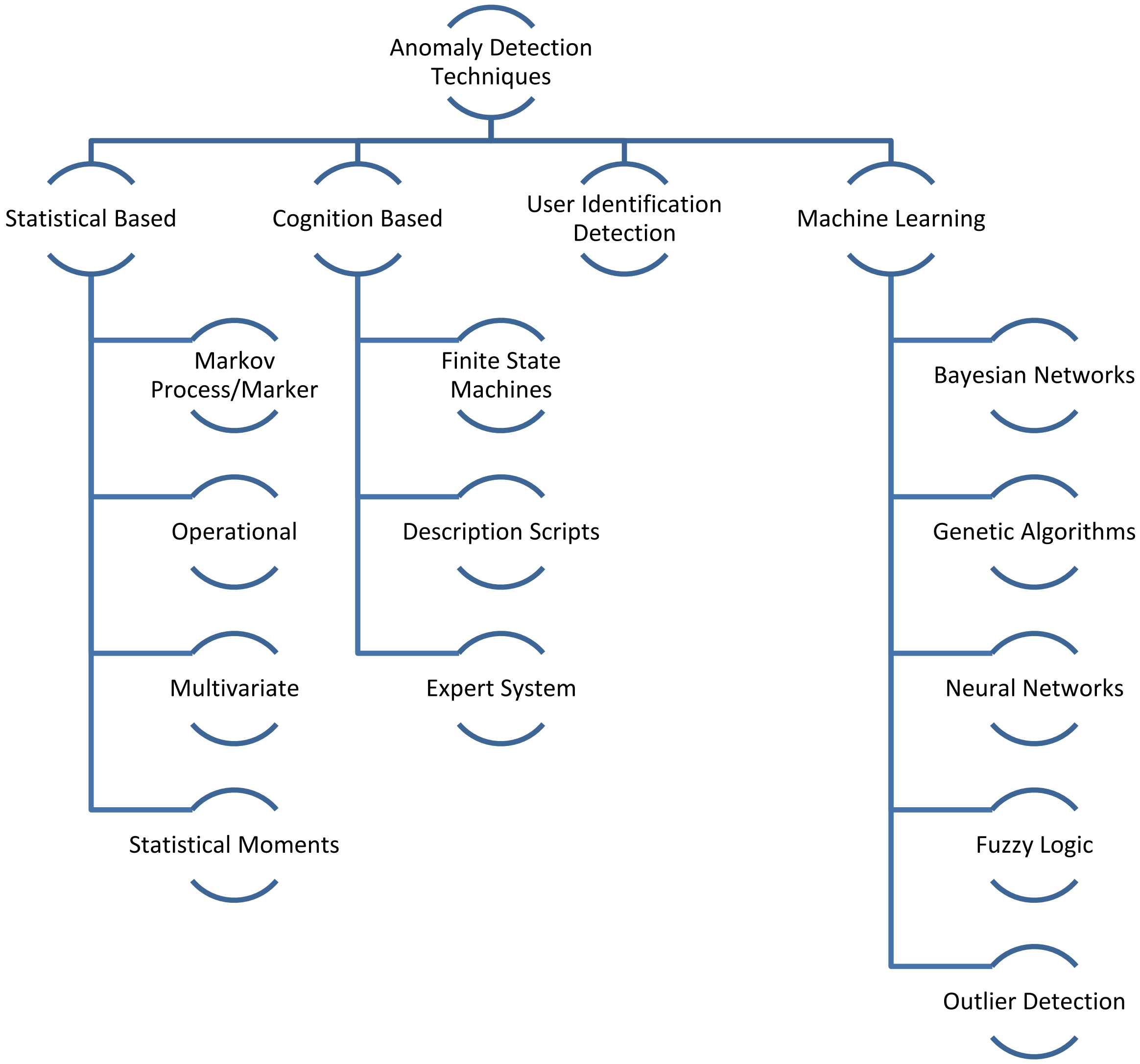


Figure 5: ADT Methods [15]

3.2.1 Statistical Based

Statistical based method is carried out on the observation of statistics collected from network data traffic. Here, the algorithm collects network traffic data for a long time and their patterns are observed. After that, Threshold metrics [16] are implemented to analyze data and any information deviated from the normal data are counted as an anomaly. Next, required removal processes are implemented depending on the nature of anomaly and network traffic observation processes are updated. Ghorbani et al. [16] have proposed wavelet time series analysis and system identification theory through Statistical based method for anomaly detection. In his process, normal window is separated using wave approx. and regressive tool to compare the incoming traffic. Furthermore, other ID methods such as Markov Process/Marker, Operational, Multivariate and Statistical methods are also carried out through Statistical based method. On one hand, statistical based method is suitable to filter doubtful traffic and detect bots but on the other hand, process is expensive to implement and operate [15].

3.2.2 Cognition Based

Cognition-based method is also known for Cognitive models or Expert Systems. It is focused mainly on the audit of data classification. In this method, whitelist separation rules and awareness procedures are predefined. With classification rules, huge traffic is separated from the suspicious traffic and botnets are detected. Decision trees, adept systems, and finite state machines are few examples of cognition-based ID method [10].

3.2.3 Machine Learning Based

Machine Learning Based method uses vector machines to detect Bots from doubtful traffic. Here, vector machine collects the information on doubtful data traffic and depending on the network, HID or NID is installed to detect possible threats. The big drawback on the process is that most systems are passive and communication between the sender and the receiver is not in real time. Furthermore, IDS can observe and identify the administration but cannot take actions due to the false alarm issue. Thus, the method needs manual effort for threats removal on detected intrusions. On the other hand, hackers can control the system in the absence of an administrator even though strong access and authentication are implemented. Thus, Machine learning can be a good platform to control ID issues only on large networks with the presence of an observer [17]. Some popular methods under Machine Learning platform are explained shortly in Table 1 [10]:

Methods	**Description**
Fuzzy logic	Based on fuzzy data sets with unidentified boundaries.
Bayesian Networks	Graph model based to encode relation among data sets.
Genetic Algorithm	Chooses successful parental generation and updates solution for the problem.
Neutral Network	Uses parallel simple algorithms to classify complex issues
Distance based	Calculates the distance between two data class and observes similarity.
Probabilistic method	Works on the assumption of parametric and nonparametric approach.
Kernel based	Uses the method to project non-linearly separable data to space where data is linearly separable.

Table 1: Machine Learning based Methods [10]

3.2.4 User Identification Based

User Identification Based model monitors user behavior and figures out data sets pattern to separate doubtful traffic. On the process, any unnatural acts of data flows are taken as an anomaly. But the false alarm is high in this process because user's behavior might vary all the time. This way, from attacker's point of view, it isn't hard to analyze software detection model and divert attacking pattern preventing from being detected [10].

3.2.5 Computer Immunology Based

Computer immunology-based methods uses principles similar to human body immune system, ‘prevention is better than cure'. Like anti capsules are taken for the disease, anti-intrusion patterns are used in this model. When intruders try to enter the network, their connection attempts are monitored and mobilized. Many AI systems based on software and algorithms are deployed in this process for immunity. During the analyzing process, software also develops its ability to detect vulnerabilities and adapt to the self-learning environment [10].

3.3 Evaluation of Results

By this section, we are already familiar with intrusion, botnet, botnet detection methods, and Algorithm. Furthermore, detailed structure of taxonomy and detection examples will be discussed in coming chapters. So far, it is understood that Botnet detection principles are based on analysis of datasets collected and implementation of software algorithms. During this process, error rates are the main issues to evaluate the detection application. Error rates refer to possibility of errors that can happen during the experiment. Error rates can be in any algorithm or software to examine the efficiency of method. Applications resulting on high error rates might show secured data as anomaly and doubtful data as secured leading to false results. So, higher the error rates refer to less efficiency of the application. Thus, lower error rate is necessary in an application to get effective result. Error rates can be determined by several terms below:

- False Positive : Known as a false alarm, when expected for something negative but comes positive
- False Negative : Known as missed detection error, when expected for something positive but comes negative
- True Positive : Doubted datasets marked exactly an anomaly
- True Negative : Secure datasets marked as safe [10].

While detecting botnet, false rates are important to consider in algorithms and improve the efficiency of experiments [10]. We will be going through individual botnet behavior, intrusions, detection model and false rate evaluation in individual botnet types starting with IRC in the next section below.

4 Internet Relay Chat Botnet

Internet Relay Chat (IRC) is an application theme protocol operating in the form of text. In IRC, text conversation known for "chat" works as a centralized server networking model. Internet Relay Chat clients are computer program user who runs the chat application installed on their computer. IRC was initially designed for group communication and with the evolving process covered private chat, data transfer and file transfer. As the program began to expand, loopholes were also generated resulting cyber security threats, and among which, one is IRC Botnet. This chapter covers information about IRC Botnet behavior, examples, vulnerabilities, security, detection methods, algorithm and test results [18].

Among botnets, IRC Botnet is popular among botmaster due to its easy architecture and understandable working mechanism. It is targeted for small attacks where algorithms and designs are kept simple making it easy to handle. The first IRC Botnet was created by JellFisher in 1993 with the name "Eggdrop" and still being maintained with IRC based backdoor. Eggdrop was widely used over the time as non-malicious bot [19] due to less vulnerable. Eggdrop that time was able to withdraw only small section of chat information's. After that many other IRC Botnets were introduced and one is "PrettyPark" established also in 1993 [7]. "PrettyPark" sends information to Botmaster every 30 seconds and make sure it stays alive within the channel. While being in the channel, it can receive commands and access information's such as Computer name, Product name, Product key, registrations, ICQ details, email addresses, and passwords. [7]

4.1 IRC Botnet Behavior

IRC Botnets are First generation and less vulnerable botnets in comparison to HTTP and P2P botnets. IRC botnets are less vulnerable because they can target small network or specific one computer at a time. On the other hand, anti-malware can easily detect most of the IRC Botnets and can remove them from the system. Though IRC Botnets are less destructive and have less efficiency, they are popular among hackers due to its good interactive behavior, low latency, easy handling and maintenance. In terms of result, it has low false alarm as well as high accuracy and if left undetected, can take down big servers too. Thus, botmaster's first choice has always been IRC botnet when attack intentions are targeted for small network to achieve specific details [8]. Furthermore, on how IRC attacks are performed is explained below in intrusion section.

4.2 IRC Intrusions

Botmaster can perform various actions like extract confidential information, spy, monitor chats, identity-theft, access bank details, passwords and so on through IRC botnet intrusion. For the process, botmaster first designs IRC bot and deploy it for installation to the targeted host where mutual communication between host and botmaster is established. Figure 6 shows IRC botnet intrusion process with steps and are explained below [20].

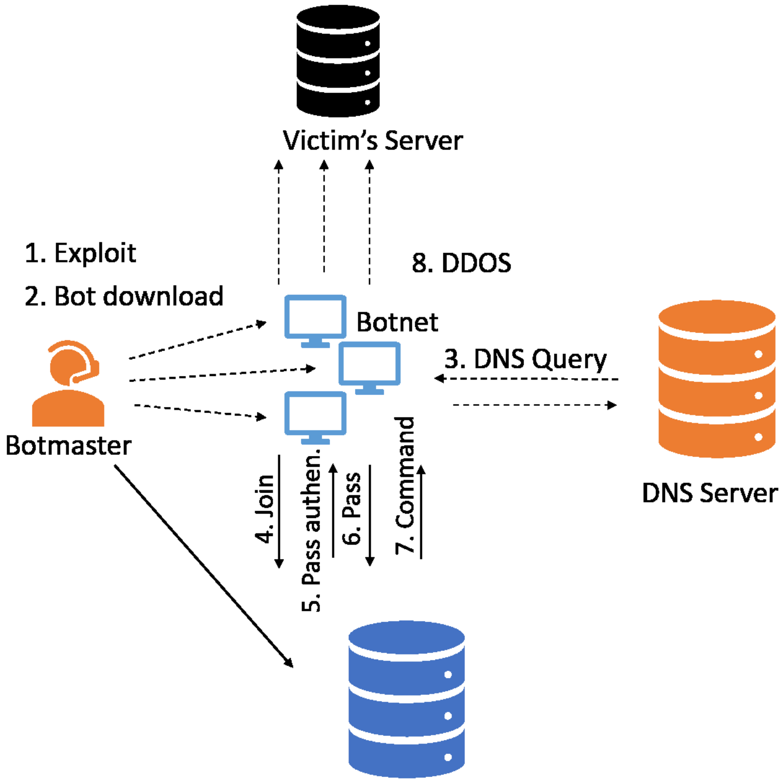


Figure 6: IRC based Botnet Lifecycle [20]

As shown in Figure 6, Botmaster first uses remote IP address and exploits bot to the host. Upon exploitation, bots are downloaded, and binary code are installed. Once installation is completed, every bot on botnet is under control of botmaster through C&C channel and IRC server is connected through DNS query as shown in Step 3 of the figure. Here, bot starts with PASS message to the IRC server and server authenticates the bot by validating its password establishing the communication. In many situations, not only bots but botmaster also needs to validate himself to the server. Once the authentication process is completed, the connection criteria between bot, botmaster, and server is completed. Finally, the system is now under the control and to begin a DDoS attack, botmaster can simple send command as "ddos.start victim_ip" where all bots get the command and attack the server as shown in step 8 of the figure above [20].

4.3 IRC Intrusion Records

In early periods of internet, hacks and attacks were meant to be for information, curiosity, bug exposure and unauthorized login. But modern IRC attacks not only exploited stealthy mechanisms but also disrupted their resources and connections to get into the system. Amount of loss covered by intrusion is getting bigger, mostly related to huge financing sector [21]. Record [22] says that millions of IRC users just in America and many more in world-wide uses instant messaging services like AOL messenger and ICQ being potential IRC botnet victims. Many intrusions are encountered on a daily basis within these applications and some large attacks encountered are listed below caused by IRC intrusion:

- In Romania, one of the largest IRC Network was attacked by DDoS and much of its services were shut down [23].
- Many ISP have been hit in France and Netherlands within last few years from DDoS IRC botnet attacks [23].
- Undernet is known for one of the largest networks with 45 servers in 35 countries communicating over 100,000 people every week. According to Undernet, many servers were controlled with tidal waves of IRC data.
- In February 2017, huge sites like Yahoo, eBay, Amazon, the FBI and many more were shaken with fake information requests and was shut down in many cases [23].

Though IRC botnets are considered less vulnerable, it can lead to huge loss if necessary actions are not applied in time. Thus, it is important to know the behavior of latest recorded IRC Bots as well as provide necessary precautions to the server and in next section below we are going to shortly discuss popular IRC Botnet detections methods.

4.4 IRC Botnet Detection

Many theories and experiments were conducted to detect IRC intrusion and prevent the system from intrusion [21]. In this section, popular IRC botnet detection programs are explained briefly.

In 2006, Dagon et al., [24] explained a program focusing on time patterns to monitor botnet activities based on IRC protocol. IRC protocol was very congenial at this period and IRC channel operators used to handle the channel as well as monitor them automatically. While monitoring, any doubtful traffics were separated and preventive measures were applied for the security.

Furthermore, W.T. Strayers and his researcher friends developed metrics-based flow analysis [25] where IRC sessions are filtered out first and flow metrics are applied to detect IRC botnet from normal traffic.

Later, network layer and application layer combine analysis research was delivered to cyber-security [21] that filters the traffic based on centralized or decentralized server and observes the result separating an anomaly. Besides them, many machine learning techniques and AI were added in the detection process giving buzz, ring or notification alarms to the operator about IRC intrusion and making the detection task stronger [21].

In [26], Rajab et al. approached distributed monitoring based on network (centralized and decentralized), and separate doubtful traffic from the dataflow. Similarly, many other monitoring services have been introduced including domain and DNS for blacklisting IRC botnet from the data traffic. Though many approaches have been introduced, security implementation depends on organization point of view, data-traffic, security importance, previous hacks, and loopholes. No matter what, all IRC detection process follows common Traffic Application Classification process to separate IRC traffic from whole network and detect botnet and is explained in next section below [21].

4.5 IRC Traffic Application Classification

Traffic application Classification based security is a list of process to determine intrusion from the big data traffic. In our case, IRC Traffic Analysis is used to detect the IRC Botnet and the process is as shown in the Figure 7.

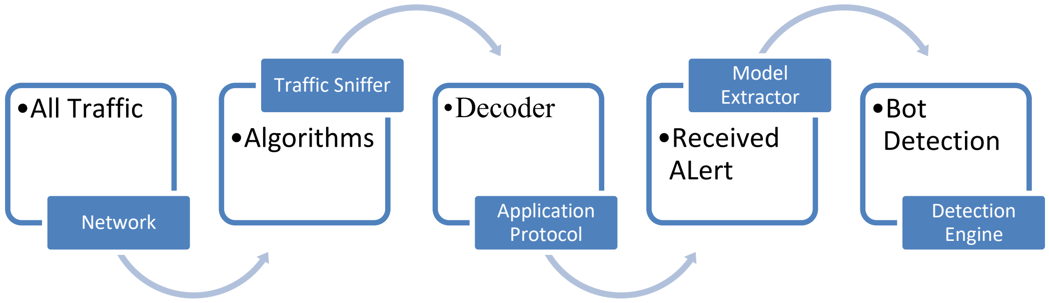


Figure 7: Traffic Analysis Framework [21]

In the beginning, all incoming and outgoing data-traffics are separated into IRC or non-IRC traffic. The separation process is not mandatory but reducing big volume of traffic for computation gives significant result and very low false alarm. Then, IRC traffics are passed through the sniffer to be decoded where alerts are used for intrusion and after observation, doubtful IRC botnets are labeled as anomaly. Finally, IRC doubted traffics are analyzed and passed through the detection engine as shown in Figure 7 where IRC Botnet is detected [21].

Among many Traffic Analysis framework applications, "Botsniffer" is one application that uses similar process for IRC detection and is a good example to see algorithms as well as results explained in the next section.

4.6 Botsniffer

Botsniffer is a life-cycle approach for IRC Botnet detection. It starts with filtration of doubtful traffic, implementation of algorithm for detection process and leads through alert message to the administration department. Botsniffer works without prior knowledge of signatures on server addresses and follows IRC Traffic Analysis method as explained in previous section. Botsniffer is also based on the study of spatial-temporal correlation and similarity with previously recorded IRC botnets. Spatial-temporal correlation refers to any unnatural activities like coordinated communication, unauthorized logins, and intrusions that are observed from the bot's behavior. In this section, we are going to cover its behavior, botnet attacks, Botsniffer architecture, algorithm, test results and evaluation for IRC botnet detection [27].

4.6.1 Architecture and Algorithm

In centralized C&C system, Botsniffer can perform mainly two types of operation i.e. "push" and "pull". IRC based C&C falls under push operation where messages are pushed or commanded to bots. On the other hand, Pull style is used for HTTP based browsing operation. Detection methods and implemented algorithms are different in both push and pull style operations as one is messaging application based and another is internet browsing based. Lots of existing Botnets such as Phatbot, GTBot, Sdbot, Spybot, Rbot/Rxbot and so on uses IRC push C&C. Figure 8(a) shows botnet's push-style C&C on IRC and 8(b) shows communication process with explanation below [27].

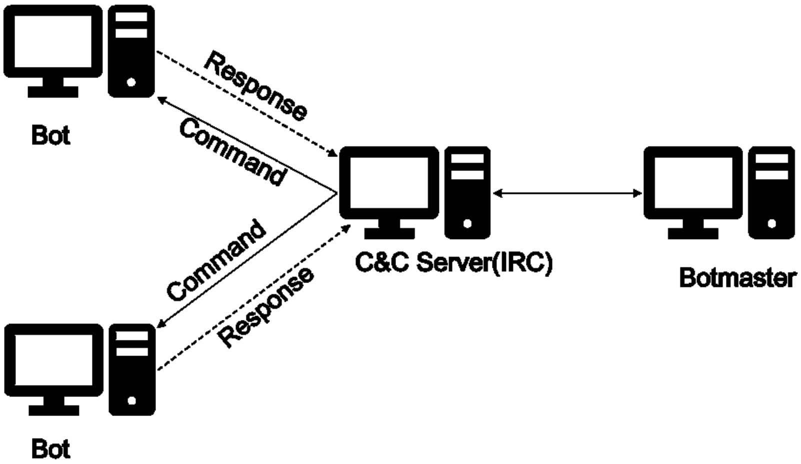


Figure:8(a) C&C: IRC Push Style [27]

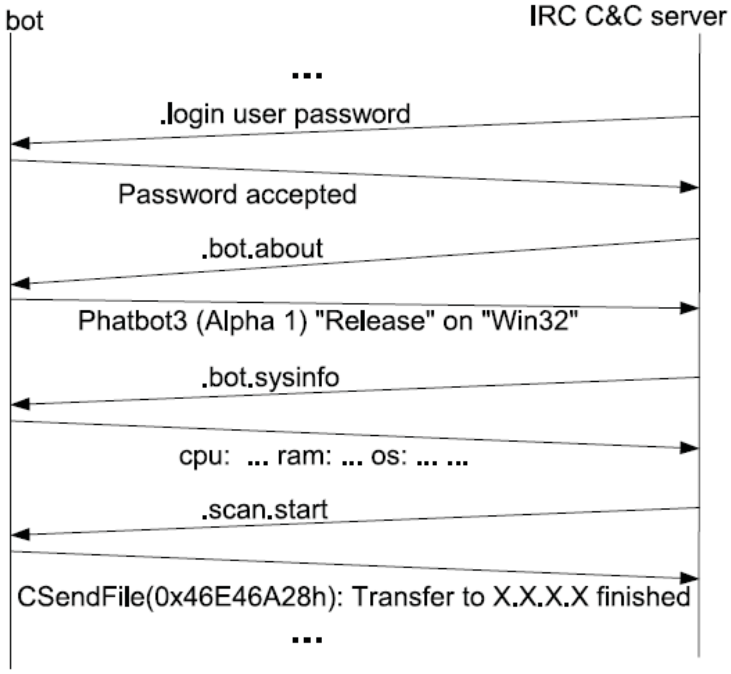

bot
IRC C&C server
...
.login user password
Password accepted
.bot.about
Phatbot3 (Alpha 1) "Release" on "Win32"
.bot.sysinfo
cpu: ... ram: ... os: ... ...
.scan.start
CSendFile(0x46E46A28h): Transfer to X.X.X.X finished
...

Figure 8(b) C&C Communication [27]

Figure 8(a) and 8(b) shows initial infection and bot code for push style attacks through C&C channel where botmaster uses IRC intrusion process as explained in previous section [4.1.2]. With initial deployment of IRC bot, establishes communication through authentication. When communication platform is established, various commands like '.bot.about' to receive information about the bot, 'bot.sysinfo' to achieve information about the system, 'scan.start' to scan the network as well as receive every details and 'CSendFile(0x46E--):Transfer to X.X.Finished' to transfer files to the designated address are deployed. These are typical IRC botnet intrusion processes and commands for the intrusion. To determine the detection process, such command behaviors are studied properly, and architecture is designed. After that algorithms are generated to detect unusual activities as Botnet giving a final result as Botsniffer [27].

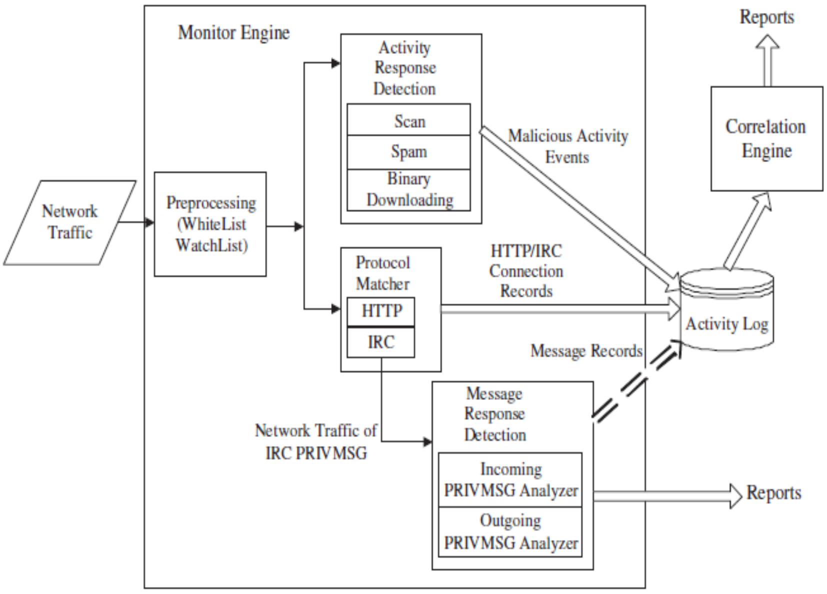

Monitor Engine
Activity Response Detection
Scan
Spam
Binary Downloading
Malicious Activity Events
Reports
Correlation Engine
Network Traffic
Preprocessing (WhiteList WatchList)
Protocol Matcher
HTTP
IRC
HTTP/IRC Connection Records
Activity Log
Message Records
Message Response Detection
Network Traffic of IRC PRIVMSG
Incoming PRIVMSG Analyzer
Outgoing PRIVMSG Analyzer
Reports

Figure 9: Botsniffer Architecture [27]

Botsniffer architecture is mainly based on two components: monitor engine and correlation engine as shown in Figure 9. Monitor engine is exploited at the perimeter of the design where it monitors traffic activities and senses suspicious traffics. In this process, firstly whitelists such as Internet Control Message Protocol(ICMP) and User Datagram Protocol(UDP) are filtered out to reduce traffic load. ICMP and UDP encountered during the process are filtered out as whitelist because they are less suspicious and doesn’t exchange data between the servers. After that, filtered traffic's behavior are studied and IRC traffic related spam as well as doubtful traffics are passed through the activity log and through the correlation engine. In activity log, information received from monitor engine is studied properly and in correlation engine, algorithm tests are applied where botnets are confirmed, and reports are submitted with alert notification. An example of detection algorithm is explained in next section below [27].

4.6.2 Botsniffer Detection Algorithm

Botsniffer uses multiple algorithms on IRC botnet detection process such as IRC filtration process, whitelist separation process and monitoring message response process. Detailed information on all these algorithm processes can be achieved from [27] and among all we are going to discuss message response i.e. Response Crowd Homogeneity algorithm to get an idea on how botnet detection algorithm and testing works. Response Crowd Homogeneity algorithm monitors message response traffic and triggers on suspicious traffic passing through correlation engine. For this process, mainly 2 algorithms (i) hypothesis for being botnet (E[n|H_1]) and (ii) hypothesis for not being botnet (E[n|H_0]) is applied. Here, hypothesis parameters are taken as well as probability of being botnet is determined from algorithm below:

For being Botnet:

$$E[N|H_1] = \frac{\beta \ln (\beta/1-\alpha) + (1-\beta) \ln (1-\beta)/\alpha}{\theta 1 \ln (\theta 1/\theta 0)+(1-\theta 1) \ln (1-\theta 1)/(1-\theta 0)}$$

Where,

H_1 = Hypothesis for botnet

H_0 = Hypothesis for not botnet

θ_1 = Probability for H_1

θ_0 = Probability for H_0

α = User chosen false positive probabilities

β = User chosen false negative probabilities

And hypothesis for not being botnet, the equation is:

$$E[N|H_0] = \frac{(1-\alpha)\ln\left(\frac{\beta}{1-\alpha}\right) + \alpha \ln\left(\frac{1-\beta}{\alpha}\right)}{\theta 0 \ln(\frac{\theta 1}{\theta 0}) + (1-\theta 0)\ln[(1-\theta 1)/(1-\theta 0)]}$$

On this calculation, θ, α and β values are obtained from earlier filtration and Response crowd density check algorithm processes [27]. Finally, a table is made with the encountered data and are analyzed as datasets from all algorithms. Based on the collected datasets, analysis and evaluation operation is conducted.

4.6.3 Datasets and Evaluation

In [27], multiple network traces were observed from the university and Botsniffer was used for IRC traffic flow observation. The total duration of the traces were 189 days and eight ports for IRC traffic with port no. 6667 were labeled as IRC-n [n = 1-8]. Other five traces contain the complete packet of all traffic and were labeled as All-n [n = 1, …., 5].

Trace	Trace size	Duration	Pkt	TCP flows	(IRC/Web) Servers	FP
IRC-1	54MB	17h	189,421	10,530	2,957	0
IRC-2	14MB	433h	33,320	4,061	335	0
IRC-3	516MB	1,626h	2,073,587	4,577	563	6
IRC-4	620MB	673h	4,071,707	24,837	228	3
IRC-5	3MB	30h	19,190	24	17	0
IRC-6	155MB	1686h	1,033,318	6,981	85	1
IRC-7	60MB	429h	393,185	717	209	0
IRC-8	707MB	1,010h	2,818,315	28,366	2,454	1
All-1	4.2GB	10m	4,706,803	14,475	1,625	0
All-2	6.2GB	10m	6,769,915	28,359	1,576	0
All-3	7.6GB	1h	16,523,826	331,706	1,717	0
All-4	15GB	1.4h	21,312,841	110,852	2,140	0
All-5	24.5GB	5h	43,625,604	406,112	2,601	0

Table 2: Normal traffic traces(left column) and Detection results(right column) [27]

Table 2 as shown above can be separated into two columns: left column and right column. Left column shows information of all traces with trace size, duration and packets received during the observation. On the other hand, right side shows doubtful data flows and traffics that went through server with false positive rate. FP reading as 0 means that there is no error in the calculation and greater the number results to decrease in efficiency of the result [27].

Finally, when doubtful traffics were passed though correlation engine, multiple bots were encountered and encountered bots are mentioned in Table 3 as B-IRC, V-Bots and HTTP bots with its traced size, duration, packet size, and TCP-flow. This test result was retrieved with separate algorithms for HTTP bots also and in our case for IRC Botnet Detection, targeted bots resulted as B-IRC bots. Results from the conducted experiment show very low false alarm and high accuracy for the IRC hypothesis algorithm but many false readings were also conducted in other network traces [27].

BotTrace	Trace Size	Duration	Pkt	TCP flow	Detected
B-IRC-G	950k	8h	4,447	189	Yes
B-IRD-J-1	-	-	143,431	-	Yes
B-IRC-J-2	-	-	262,878	-	Yes
V-Rbot	26MB	1,267s	347,153	103,425	Yes
V-Spybot	15MB	1,931s	180,822	147,921	Yes
V-Sdbot	66KB	533s	474	14	Yes
B-HTTP-I	6MB	3.6h	65,695	237	Yes
B-HTTP-II	37MB	19h	395,990	792	Yes

Table 3: Botnet Traces and detection results [27]

5 HTTP Botnet

Hypertext Transfer Protocol (HTTP) is a communication protocol based on logical links, texts, pictures and so on through centralized web browsing server. HTTP services are widely chosen services all over the world and are being used every day for online bookings, data transfer, emails, and payments and many more. HTTP services can auto-update as well as self-refresh too. With these great features also comes HTTP botnet as a drawback to HTTP services. Botmaster leaving bots to HTTP protocols are known for HTTP botnets and is an evolution to IRC Botnet. In IRC channel, bots are always connected to the system, but in HTTP, botmaster can turn bots on or off depending on the attack types. It means that, once bot is installed in the protocol, it can be kept silent until needed without any suspicions [28] and when needed can be activated through C&C channel making HTTP botnet more vulnerable than the IRC botnet.

In system state, HTTP botnet uses port number 80 allowing traffic to bypass the firewall and making it hard to detect during ID [28]. So, HTTP botnet can lead to huge loss, and it is very important to understand HTTP botnets behavior as well as go through its countermeasures. This chapter covers detailed analysis of HTTP botnet's behavior, intrusions, algorithms, detection, and evaluations.

HTTP Botnet is also known for Second Generation botnet which started in 2003 as ‘Phabot' [29]. HTTP botnets purpose is also similar to IRC such as DDoS attack, identity theft, malware spreading, spam, and fraud. When HTTP protocol was introduced in 1999, no one thought that it will be widely used all over the world and could lead to huge destruction as well as breach in privacy. And blocking all HTTP services as precaution is also not effective solution [30]. On the other hand, research on HTTP based botnets is found to be very low, especially when generation is mobile friendly, and intrusion is happening regularly.

Comparing to scalability on IRC, HTTP botnets can have wide number of bots which is comparatively dangerous but less interactive than IRC. Depending on the attack size, HTTP bots can be multiplied easily through C&C channel and pass through the designated area. It also uses simple code to control the bot and uses Bulletproof hosting for efficient online spamming. HTTP botnets are installed initially through fake advertises on the webpage (especially adult pages and gambling sites). For the installation, people are tricked to go through the advertisement and meanwhile, IP details and related information's are sent to the botmaster. More information on intrusion will be discussed in the next section [3].

5.1 HTTP Botnet Intrusion

HTTP Botnet uses HTTP POST REQUEST method for the intrusion and are passed through malicious websites into victim's computer [31]. Intrusion can relate many attacks depending on botmaster's needs, among which one is port scanning where HTTP bot scans ports to figure available services and active ports are analyzed. Another is the DDoS attack, where numerous unauthorized requests are passed through the server making it occupied resulting decrease in efficiency of the server. Next one is HTTP error attacks which is activated by requesting header section or packet in the data traffic with undefined functions [28]. Once header section is received, data information inside it can also be extracted and altered. Urgent pointer attack is also HTTP attack that sends a string of Out of Band (OOB) data that are independent in nature and leads to the malicious TCP packet. Urgent pointer is rarely used in the TCP header field but can lead to serious server failure and finance loss situations. And finally, Reset Attack that refers to resetting the system and it's done by sending reset flat information through the bot. The main function of the reset attack is to make the server busy and avoid ID processes. Among all the attacks mentioned above, common thing is that they are all controlled through C&C channel where attack types are based on attack intention [28].

As mentioned from news and enterprise [32], a study was conducted about dangerous botnets in 2012. The analysis showed that 9 dangerous botnets were encountered and 6 out of 9 were related to HTTP Botnet [33]. The reason is that HTTP services are widely used all over the world and threats are transferring in huge amounts too. Those 6 dangerous HTTP Bots and their field of destruction is mentioned Table 4 [32].

HTTP Bots	Description
1. Festi	• Also known for the king of spam. • DDoS attack since 2009. • Intruded at least 250,000 IP addresses.
2. Grum	• Also known as Tedroo and Reddyb, largest botnet found. • Sends 18 billion spam messages per day. • Shutted down on July 19, 2012 due to 18% infection of worldwide spam traffic. • Operation relies on push configuration updates to the infected system and secondly sending through spam emails. • In 2010, infected approx. 840,000 computers delivering 39.9 billion spam messages making it largest botnet.
3. Zeus	• A Trojan horse that steals bank data also called "God of DIY". • Infects usually by downloads and through phishing. • Identified in July, 2007 while infecting US Department of Transportation. • By 2009 spreaded over 74,000 FTP accounts. • Infection sectors: ABC, Bank of America, Oracle, Cisco, Amazon and so on
4. SpyEye	• Like Zeus i.e. steal bank information and consumer credited information. • 278 SpyEye still active on stealing bank information. • First hacks the account and slowly hold the system, password and so on. • Follows transactions from victim's network to an unknown destination.
5. Citadel	• The fastest spreading botnet, within 1-year citadel

	attacks were increased by 20%. • Information is sold to criminals.
6. TDL-4	• One of the latest botnets. • Directly infected 250,000 victims. • Also known for TDSS or Alureon. • Hides from antiviruses and installs master boot record. • Can generate C&C domains making it difficult to track

Table 4: HTTP Botnet and description [32]

While analyzing on HTTP Botnets above, bot can send spam in millions of numbers per day and can take down thousands of FTP accounts at a time which is big intrusion. Thus, proper preventions or detection mechanism seems necessary to keep the protocol and devices safe. We will go through the detection section soon but before that let's discuss few famous related researches and detection processes in the next section below.

5.2 Related Research Work

Many research papers and experiments have been delivered for HTTP vulnerabilities and types of C&C. Choi [34] proposed HTTP botnet detection using study on Domain Name System(DNS) traffic, where the algorithm can monitor group activities and DNS resulting separation of doubtful traffic. Jae-Seo [29] followed United Progressive Alliance(UPA) process to detect periodic repeatability that monitors repeated similarity information between HTTP client/server. During the process, incoming and outgoing traffic's behavior is compared with the normal traffic and any deviations are alerted as anomaly. Furthermore, many detection ideas were delivered covering data mining with its classification, clustering, machine learning and many more where popular HTTP based approaches are explained shortly below:

i. Automated Layered approach for layered detection

In this approach, an automated layered mechanism is used for the detection. Intrusion during the process is detected when the number of outgoing packets is greater than the incoming threshold value. Next, the database is updated automatically with the intrusion record. After an update, the database is studied and detected intrusion is filtered out through the unit. Here, accuracy and performance of the threshold effects significantly to the result on detection [28].

ii. Genetic algorithm-based ID

Genetic algorithm-based ID use information and similarity from previously detected botnets. The method contains pre-processing, extracting, testing and training phase where new traffic is compared through genetic operators. During the process, any similarity to mutation and crossover rings alarm for the intrusion. On one hand, process might be significant but since attack processes are evolving all the time, comparing security with old bots is not so good [28].

iii. DPR based botnet detection

The Degree of Periodic Repeatability analyzes HTTP client/server relation by calculating direction and standard deviation repeatability of data traffic. The analysis helps to understand user's intention and intrusion process within the browser. Assumptions are made in a way that, if DPR of a connection is less, then there is the presence of HTTP botnet. DPR based detection is probabilistic, time-consuming and can only detect DDoS attacks [28].

iv. DNS traffic-based botnet detection

DNS traffic-based botnet detection monitors group activities in DNS traffics and analyzes doubtful traffic behavior if it is a botnet or not. Algorithms in DNS based detection are implemented for botnet to use only DNS service at the initializing state and can never get into the system. In this process, botnet’s behavior in initial state is analyzed and bots are separated [28]

All detection processes mentioned above are popular researches for HTTP botnet detection. HTTP botnet detection processes can also be based on Genetic algorithm and pattern set mining. Furthermore, Botsniffer was discussed in the previous chapter for IRC channel and can also be used in HTTP botnet detection with similar process but different filtration algorithms. Just a little drawback is that Botsniffer works more efficiently for IRC Botnet than in HTTP Botnet [27] and thus in this chapter, we will discuss Adaptive Learning Neural Network method for HTTP detection. I found Adaptive Learning Neural Network method more efficient due to its adaptive behavior in detecting intrusion and is explained in next section below.

5.3 HTTP Botnet Detection Using Adaptive Learning Neural Network

Neural network is known as the future generation of computing that allows computers to think and react according to the situation. Neural networks models can classify internet users through its significant behavior like parallel processing of information, ability to recognize patterns and so on. In this section, we are going to discuss one of Neural Network method known as Adaptive Learning Neural Network for HTTP Botnet Detection [35].

Adaptive Learning Neural Network contains detection process with efficient learning rate that can adapt to previously recorded intrusion and can react efficiently to the doubtful traffic. The process can be operated significantly even when browser is refreshing, updating or operating with huge data flow in encrypted form [35]. Furthermore, the process has low FP rate than other neural models such as C4.5 Decision Tree, Random Forest and Radical Basis Function being effective to detect dangerous bots like SpyEye, Zeus, etc [35]. In this section, combine study of detection algorithms, results, and evaluation is discussed that uses Multilayer Feed-forward Neural Network with adaptive learning. So, let's start shortly with detection process as shown in the Figure 10 [35].

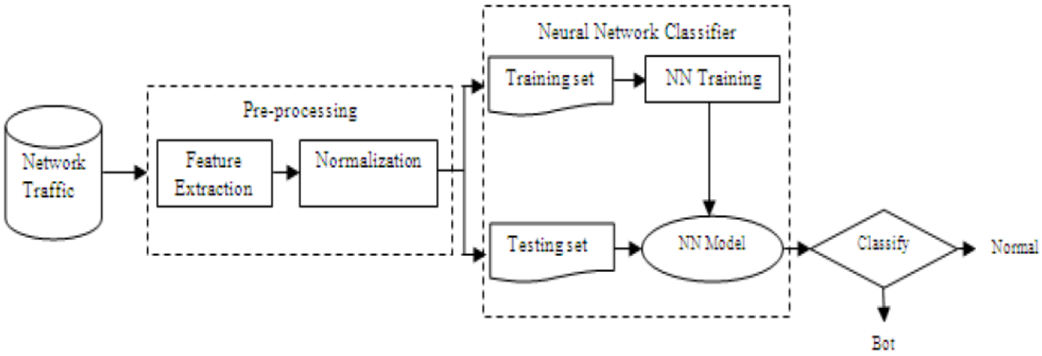


Figure 10: Block Diagram of HTTP botnet detection [35]

For HTTP botnet detection, firstly all network traffics are passed through pre-processing where data information and moral behavior of traffic is studied as shown in Figure 10. Then based on the result, HTTP traffic is filtered out and is passed through testing/training phase for Neural Network testing. During the process, any doubtful traffic is directed through the classifier where the algorithm classifies presence of botnet or not. The whole process with algorithm for data filtration and evaluation is explained in coming sections below.

5.3.1 Preprocessing and Normalization

Preprocessing is the phase of separation on HTTP and doubtful traffic from the normal ones by studying TCP packet's behavior. For an instance, when an infected bot is connected to the system, half of the open connections will increase. This results in variation of one-way connection ratio with the ratio of incoming and outgoing TCP packets [35]. On the other hand, in normal traffic, the number of SYN flag, FIN flag, and PSH changes randomly but when infected with HTTP bots these numbers stay mostly in fixed ratio. Thus, study on these features separates the traffic and doubted information's can be passed through normalization. The process can be performed as:

One way ratio of TCP = $\frac{Number\ of\ one\text{-}way\ connection\ TCP\ Packets}{Number\ of\ TCP\ packets}*100$

- A ratio of Incoming Outgoing TCP packets = $\frac{Number\ of\ Incoming\ TCP\ Packets}{Number\ of\ Outgoing\ TCP\ Packets}$
- A ratio of TCP Packets = $\frac{Number\ of\ TCP\ Packets}{Total\ Number\ of\ packets}$

- SYN Flag count = total number of TCP packets with SYN flag
- FIN Flag count = total number of TCP packets with FIN flag
- PSH Flag count = total number of TCP packets with PSH flag [35]

5.3.2 Normalization

Doubtful traffic obtained from the Preprocessing is filtered through Normalization to prevent neuron saturation. If preprocessed data is passed directly through the classification process, then there is high chance of neuron being saturated. If neuron gets saturated, then it leads to FP alarm in the output value. To prevent this situation, data is passed through normalization and is done through normalize feature extraction value "x” with Min-Max normalization [35].

$$X' = \frac{x - min"x"}{max"\mathrm{x}" - \min"x"}, \text{ where}$$

X’ = Normalized value
Min “x” = minimum value
Max “x” = maximum value

The normalization value obtained is then passed through classification where neural network training and tests are operated.

5.3.3 **Neural Network Classification**

Obtained result from the normalized value is passed through training and testing sessions followed by Multilayer networks. Firstly, in the training phase, normalized value is taken as input to a neural network where network weights and biases are organized for the degree of accuracy. Then trained values are passed through a testing phase where bold driver back-propagation algorithm is applied. The process contains 6 neurons for the dimensionality of the input vectors resulting as one neuron.

Furthermore, hidden layers are applied based on the complexity of the process [35]. Each neuron "i" in the input layer has signal "X_i" as networks input and each neuron "j" in the hidden layer receives signal as ln(j) and are performed by the equation below [35].

5.3.4 Results and Evaluation

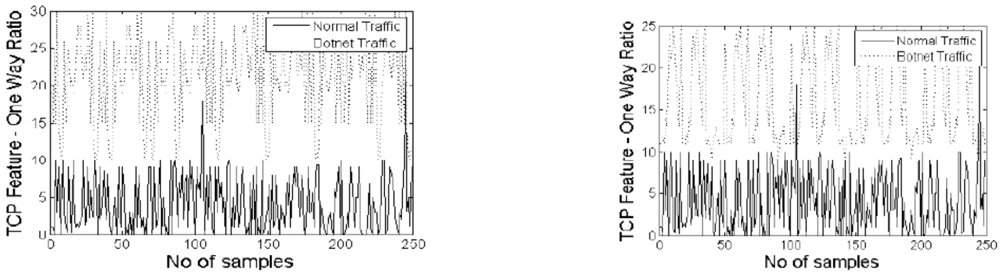
Results and Evaluations are based on the experiments conducted by [35] in SSE lab with star physical topology and 100baseTX ethernet cable. During the experiment, TCP packets were retrieved from network traffic by taking sample in an interval of 5 seconds. Traces were collected as 2 hours a day for five days in a week. In the traces, normal web traffic details were also obtained from National Knowledge Network having a bandwidth of 100Mbps.

Table 5 shows traces collected from experiment as normalization and were analyzed for the selected features from normal web traffic. SpyEye and Zeus bots on the process were found as the zombies for email spamming.

Bot Trace	Trace Size	Packets Number
Zeus-1	5.85MB	53,220
Zeus-2	4.13MB	37,252
Spyeye-1	25.17MB	1.75.870
Spyeye-2	3.90MB	35,180

Table 5: Traces of different Web-based Botnets [35]

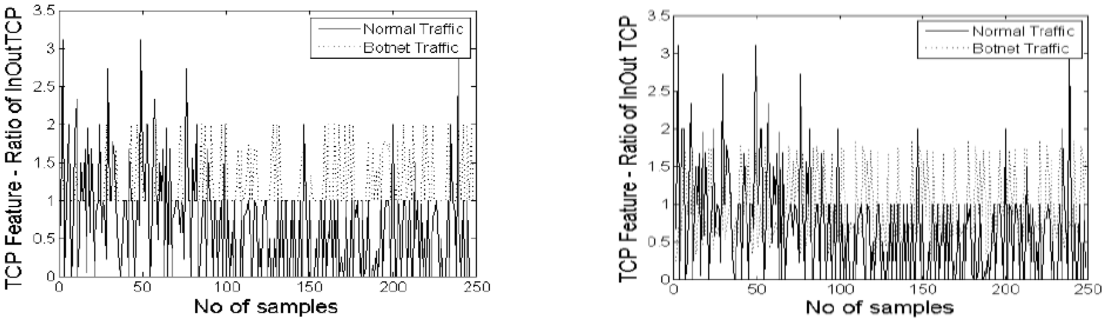
According to datasets [35], 10250 normal traffic samples with 8500 Zeus and 8250 SpyEye bot samples were taken for the experiment and ratio of one-way connection, incoming outgoing packets, as well as flag counts, were observed. Figure 11 shows high ratio and irregularity in bot's traffic from the normal traffic where normal traffic data flow ratio, PUSH Flag and Incoming-Outgoing ratio remained low level with less complexity in correspondence to HTTP bots.



(a) For Normal and Spyeye flow

(b) For Normal and Zeus flow

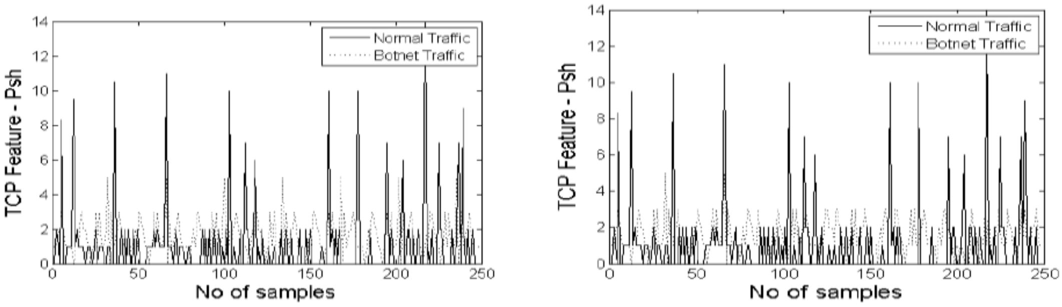
Figure 11: One-way connection ration of TCP Packets [35]



a) For Normal and Spyeye flow

(b) For Normal and Zeus flow

Figure 12: Ratio of Incoming Outgoing TCP Packets [35]



(a) For Normal and Spyeye flow

(b) For Normal and Zeus flow

Figure 13: TCP PUSH Flag [35]

After obtaining values from normalization, doubtful traffic were passed through Neural Network Multilayered process resulting up to 99% accuracy on detection. Table 6 below shows comparative results on detections using Adaptive learning method with Decision Tree, Random Forest, and RBF resulting SpyEye and Zeus as HTTP botnets.

Method	Botnet	Precision	Recall	F-Measure	Accuracy
Decesion Tree		0.968	0.931	0.949	96.5333
Random Forest	Spyeye	0.968	0.934	0.950	96.667
RBF		0.976	0.927	0.950	96.5333
Proposed Model		0.964	0.983	0.973	99.03
Decision Tree		0.965	0.930	0.941	96.1333
Random Forest	Zeus	0.952	0.930	0.940	96.00
RBF		0.959	0.922	0.940	95.8667
Proposed Model		0.948	0.992	0.969	99.04

Table 6: Performance Measures of Spyeye and Zeus Botnet [35]

With above experiment and analysis, Adaptive learning method with neural network seems to give efficient result and very less False Positives for HTTP Botnet identification [35].

In this section, we discussed about Botnet behavior and Detection method with result analysis in HTTP protocol. So far, we understood the nature of botnets in the centralized server with its detection processes and test results. In next chapter, we will be diving through decentralized server and P2P Botnet which is more vulnerable and destructive in nature.

6 Peer-To-Peer Botnet

P2P is a computer network where two or more than two computers are connected as peer for easy means of data transfer. All the connected computers in peer network act as both client and server and P2P botnet itself refers to botnet targeted for P2P networks [5]. Since all bots in the network can connect and communicate with each other, P2P botnet is an evolution to IRC, HTTP and efficient means of botnet for bigger attacks. All computer acting as client and server demolishes use of centralized network necessity and botmaster can operate efficiently even when ID filters certain bots. Upon filtration, other bots can continue spreading and cover the network as it was infected before. This feature of self-propagating brings quite a challenge in Computer Security and ID. Figure 14 explains similar information for communication among bots on a P2P network for intrusion [9]. In the figure, hacker from a remote location has attacked the targeted server through the help of available peer servers. Hacker takes the advantage of all computer acting as server and commands are passed generating individual computer as bot. After that, intended actions are passed to victim's server for the intrusion [36].

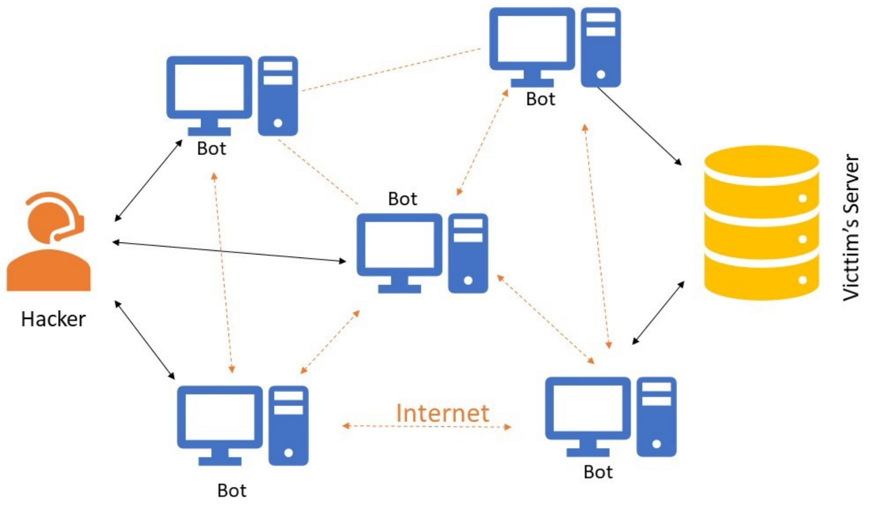


Figure 14: P2P Botnet Operation Cycle [37]

There are many P2P bots from the early situation and first P2P botnet was noticed in 2005 with the name ‘sinit'. After that, P2P Botnet has been evolving every year and some popular P2P botnets encountered are Napster, Direct Connect, Gnutella, eDonkey, Fast Track, WinMX, Ares, BitTorrent, WASTE, Kademlia, Zbot and many more [38]. More information on these botnets are explained in "Intrusion Example" section below starting with a common field as ‘Bootstrapping'.

6.1 Bootstrapping

Bootstrapping refers to a self-starting process of a computer known for booting. Bootstrapping is one of the start points for the P2P intrusion [39] For a bot to join the network, information on IP address of minimum one node is required and this is the situation where bootstrapping is used. Firstly, the bot is coded by botmaster with bootstrap servers and are deployed to the targeted network. When bot gets into the network, bootstrap server gathers information from other node IP addresses and provides the information to new bots being introduced to the network. During the process, bootstrap servers usually require signing credentials to prevent from ID and in many other cases for ID prevention, hacker provides invalid IP addresses remaining undetected [40].

In P2P network, all bootstrap servers are central point servers and one way to prevent intrusion is by shutting down all the servers. But, bringing down all the servers is not practically adequate, and the process wouldn't help much since bots are already on the system and becomes active whenever the server is turned on. However, the process can prevent other bots from joining. Taking advantage of such botnet's resilient behavior and decentralized server behavior, many P2P Botnets are designed and delivered in the network. Among many P2P Botnets, I am going to discuss few famous P2P Botnets in the next section below [40].

6.2 Famous P2P Botnets

Most of the P2P Botnet attacks are found in Windows OS among which one is Sality. Sality was developed in 2008 and is estimated to be around 1,000,000 in number [9]. Computers infected with Sality can communicate over P2P for proxy communication, spam mails, web server compromise, distributed computing, filtration of sensitive data and so on [9].

Another famous P2P Botnet is Conficker, also known for Downup and was introduced in 2008. It attacks MS OS and is estimated to be over 10,500,000. It uses dictionary attacks in Windows OS and extracts administration passwords as well as any sensitive information's [9].

Ramnit, introduced in 2011 is a P2P botnet that affects Windows OS too. Ramnit is estimated to be 3 million and has infected around 800 000 Windows devices between September and December 2011. Later, it was dismantled by Europol and Symantic Security in 2015 [9].

ZeroAccess, also known for Sirefef or Max++ is a botnet introduced in 2011 that affects bit mining and click fraud. They are estimated to be around 2 million and uses Trojan horse to download other malwares. It has ability to infect computers, staying hidden inside the network [9].

Nitol, was introduced in 2012 and almost 85% of detected Nitol was found in China. It spreads malware and performs DDoS. It is also said that Nitol Botnets are installed during assembling and manufacturing process of the device. Thus, they are mainly found in brand new computers and devices [9]. Based on above information, I have created P2P botnet table for clear estimation in number and description below:

Botnet	**Year**	**Estimation**	**Description**
Sality	2008	1,000,000	Proxy communication, spam emails.
Conficker	2008	10,500,000	Windows OS dictionary attack.
Ramnit	2011	3,000,000	Windows OS attack.
ZeroAccess	2011	2,000,000	Bit mining and Click fraud.
Nitol	2012	Unknown	Spreads malware and DDoS attack.

Table 7: P2P Botnets [9]

6.3 Related Intrusion Detection Approaches

Taking down all infected nodes have still been a challenging issue. But based on the size of P2P networks, many researches are delivered for P2P botnet detection. Among them, one is peer poisoning where a malicious node can be introduced for detection of doubtful IP addresses and activities. In this approach, all infected addresses can be separated from secured IP's through enough malicious node distributed in the system [41].

P. Narang, et al. [42] on the other hand, approached for 'PeerShark', a research process to identify P2P botnet traffic and separate them from the normal traffic nodes. The process evolves 5-tuple flow-based detection to 2-tuple flow-based detection. Traditional 5-tuple flow contains IP address, port number, destination IP, port number, and protocol. However, in 'PeerShark' 2-tuple flow approach included port and protocol oblivious only.

C. Dillion, et al. [43] explains detection of individual P2P bots within the network perimeter. The process is covered with P2P overlay network and NetFlow protocol for the analysis.P2P overlay network and NetFlow protocol compares incoming/outgoing data traffic within the node and doubtful traffic is considered as anomaly. During the testing, Zeus malware was taken in a limited access network where all incoming requests were taken inside the network and botnet was detected.

D.Zhao, et al. [44] explained an approach to detect P2P botnet by machine learning classification techniques. The process explains feasibility of detecting an intrusion through time intervals, classification behaviors and experimental evaluation in comparison within the available network data.

C. Rossow, et al. [45] presents graph model to observe the intrinsic feature and vulnerabilities of P2P botnets. Studying its behavior, matching doubtful data traffics are separated and detection measures are applied.

Karagiannis [36] approached an algorithm to separate botnet from the network. The idea is also based on the calculation of successful bot connections, analysis on node actions, application, and network. Based on observed value and analyzing it through the algorithm, P2P botnets are detected.

Gizzard [8] has presented deep aspect on the storm of botnet detection where the process covers intrusion and communication behavior. Antti [2] has researched P2P botnet action feature and explained an idea on how to control them through Single Shot Text Detector(SSDT).

Mathew [28] figured an approach to separate the net stream of Botnet from the normal traffic. Similarly, many ideas and research have been presented for P2P detection concerning communication delay time, hosts action, correlative behaviors and similarity. In next section, one of P2P botnet detection approach based on Algorithm Network Stream Analysis is explained to have clear view on how P2P botnets are filtered and detected.

6.4 Algorithms Based on Network Streams Analysis

P2P botnet shows character of paroxysm and distribution making botnets difficult to detect. Here, paroxysm relates to sudden attack and distribution relates to spreading behavior over network. In P2P, all machine works as server and when affected, all bots can perform sudden attacks and well as spread through the whole network. Furthermore, in P2P botnet, it is impractical to shut down all servers and single server failure doesn't stop bots to perform the attack. Upon server failure, other bots can spread over the remaining active servers and continue the attack process. These features are marked important while approaching for P2P botnet detection method. P2P botnet can be detected mainly with two methods: Detection based on Protocol Feature Codes (DoPF) and Detection based on Network Streams (DoNS) [41]. Despite both methods are used for botnet detection, DoNS method is found to be more efficient than DoPF. It is because, in DoNS, the application is passed through the network data streams in Network Layer as well as Transport Layer. Furthermore, stream analysis process is relevant to decentralized platform with efficient result as well as veracity [41]. In our case, Algorithm Based on Network Stream Analysis also uses DoNS to extract P2P Network traffic as well as P2P botnet behavior.

In this section, DoNS model is presented based on three algorithms. Firstly, a method is applied to detect P2P nodes based on the paroxysm and distribution characteristic of Network stream. Then, clustering algorithm is applied, derived from K-mean Clustering. Finally, botnet detection algorithm is applied to doubtful actions of the bots. The overall process is followed by the model presented in Figure 15.

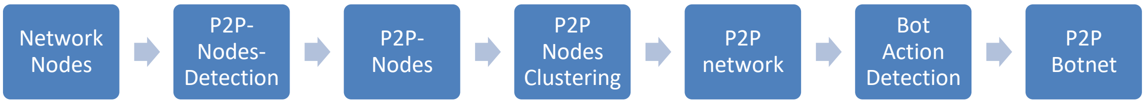


Figure 15: P2P Botnet Detection Model [41]

6.4.1 P2P Nodes Detection

P2P Nodes Detection is the first step for Algorithm Based on Network Stream Analysis where P2P nodes are initially separated from whole traffic to reduce work load. This bypasses unwanted traffic and improves efficiency in ID. Since P2P nodes are decentralized, each node is connected with more subnet and network nodes. So, paroxysm degree(Sa) and distributed degree of node (Sd) is calculated respectively at time period(T) to receive P2P network node (S). Since, whole process is structural and lengthy, we will only talk about the experimental datasets and results received from the algorithm and filtration process [41].

During the filtration process, three different kinds of P2P protocol applications (Bittorent, Emule and Kazaa) were analyzed with network stream characteristics. In this test, 100 nodes with 30 P2P nodes at time interval(T) = 60 minutes was taken to calculate (Sa) and (Sd). After applying the algorithm, P2P nodes were filtered and separated from other nodes where threshold values were recovered as (Sa) from 5 and

(Sd) from 0.6. Since common nodes have lower values, we can see that P2P nodes were separated from other common nodes as shown in Figure 16.

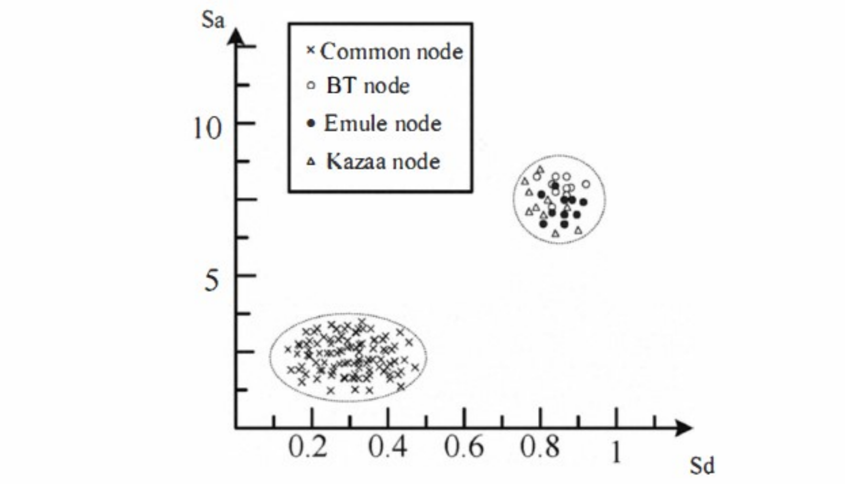


Figure 16: Nodes Distribution [41]

6.4.2 P2P Nodes Clustering

P2P Nodes clustering is second step on detection of P2P Botnet. P2P nodes received from P2P node detection section above is passed through clustering algorithm and are analyzed with P2P pair nodes behavior, stream, symmetry stats, quantity and frequency of data. In this process, suspicious traffic is filtered from normal traffic and the load for detection is reduced. Complete algorithms and explanation can be obtained from [41].

As for the example, let's take a small section algorithm example of pair nodes to calculate connection degree of pair nodes Y(i,j),

$$Y_{(i,j)} = \lambda_1(N_{ij}+N_{ji}) + \lambda_2 \times k_c \text{, where}$$

$N_{(i,j)}$ = number of nodes(i,j),

K_c = time of connection between the pair nodes (i,j),

λ_1 & λ_2 = constants

After that, distance between the node (D_{ij}) is calculated as, $D_{ij}=l/Y_{ij}$. Finally, when the values are obtained, symmetry degree of pair or nodes are determined [41] and passed through K-means Clustering to cluster P2P nodes. K-means Clustering is a popular method for data analysis using algorithms and understanding node behavior. It helps to determine similar behavior of doubtful traffic from the recorded datasets [41].

Regarding the experiment, 30 nodes each from Bittorent, Emule and Kazaa were taken and results obtained from K-mean clustering algorithm with different time intervals is shown in Table 8 below. Table gives list of protocols applied in time interval 1 hour, 1 day, 10 days and 30 days respectively. It shows that by the time of 30 days, all nodes were clustered and matched with the Botnet behavior in K-means of Clustering. After separation, all nodes were passed through final step for botnet detection and determine their attack types.

Clustering Time	**Protocol**	**Number of Nodes**	**Number of Clustering Nodes**	**Percent of Clustering(%)**
1 hour	Bittorrent	30	16	53
	Emule	30	18	60
	Kazaa	30	12	40
1 day	Bittorrent	30	24	80
	Emule	30	30	100
	Kazaa	30	26	87
10 days	Bittorrent	30	29	97
	Emule	30	30	100
	Kazaa	30	30	100
30 days	Bittorrent	30	30	100
	Emule	30	30	100
	Kazaa	30	30	100

Table 8: Result of Clustering [41]

6.4.3 P2P-Botnet Detection

P2P botnet detection is the final step for ID where suspicious traffics received from clustering process is passed through the detection algorithm. By comparing similar actions of the bots through algorithm, P2P botnets are detected. Algorithm used in this method is independent of botnet's communication protocol and content [41].

During the process, set of P2P nodes were taken as Sp = {S1, S2, Si….} and Ω_N as subset of Sp. Furthermore, Rc, R, Rd, Re, R0 were calculated as scanning behavior, spam behavior, binary file downloading behavior, exploit behavior and DDoS behavior respectively. These values are obtained as,

$R_c = C_{\Omega N} / N_{\Omega N,}$ where, $C_{\Omega N \text{ is}}$ number of nodes with similar scan behavior,

$R_s = S_{\Omega N} / N_{\Omega N,}$ where, $C_{\Omega N \text{ is}}$ number of nodes with similar spam behavior,

$R_d = D_{\Omega N} / N_{\Omega N,}$ where, $C_{\Omega N \text{ is}}$ number of nodes with similar downloading behavior,

$R_e = E_{\Omega N} / N_{\Omega N,}$ where, $C_{\Omega N \text{ is}}$ number of nodes with similar exploiting code behavior,

$R_o = O_{\Omega N} / N_{\Omega N,}$ where, $C_{\Omega N \text{ is}}$ number of nodes with similar DDoS behavior,

For the test, three different kinds of P2P Botnets (Storm, Nugache and Slapper) were passed through the P2P-Botnet Detection algorithm for time interval 24 hour and result achieved is shown in Table 9 below [41].

Samples	**R_c**	**R_s**	**R_d**	**R_e**	**R_o**	**R**
No bots	0	0	0.7	0	0	0.7
Storm	0.1	0.8	0.4	0	0.9	2.2
Nugache	0.9	0	0.6	0	0.3	1.8
Slapper	0.9	0	0.5	0	0.4	1.8

Table 9: P2P-Botnet Detection Result [41]

From the result above, it is analyzed that storm's main behavior is sending spams where as Nugache and Slapper does lots of scanning behavior. Furthermore, from the same table, Botnets individual activity in different attacks can also be seen. To summarize, P2P network stream analysis method is efficient to detect botnet, where P2P network is infected by bots.

7 Botnet Analysis

Analysis of Botnet in this chapter shows concrete result of my research which is drafted from online research papers, University of Turku library books and real-life botnet examples mentioned in each chapter above. In the beginning of this Thesis, we went through botmaster, botnet commands, bot residing machines, bots purpose, anomalies and methods for IDS and many more. After that we discussed about Botnet Taxonomy i.e. IRC, HTTP and P2P Botnet. By this time, we noticed that individual botnet's communication, scalability, server and interaction behavior varies among each other and detection algorithms are applied accordingly. Depending on Botnet behavior many software and applications like Botsniffer, Adaptive learning neural network and stream analysis were implemented to the network for Botnet detection. The identification, filtration techniques and algorithm used in Internet Relay Chat (IRC) is different from the ones used in Hypertext Transfer Protocol (HTTP) and Peer Network. These protocols are from different network and in individual chapter above, botnet's taxonomy, behavior, intrusion examples, intrusion process, vulnerability and method for detection were shown with algorithm examples. Based on such individual botnet's information, this chapter presents comparative analysis between each bot's activity, behavior and vulnerability in upcoming sections.

7.1 Botnet Behavior

After going through all individual botnet behavior, I have analyzed that botnet is created with a combine study of target host programming, resilient indentation, decoding, data transfer, communication and C&C. Based on attack purpose, bot is designed to retrieve required information. While thinking from botmaster point of view, target host and security measures to be applied in the network is very important to study. If I am supposed to be a botmaster, I will be focusing on network drawbacks to deploy designed bots whether through internal/external drive or by distracting inside user's behavior through phishing or fake advertises. After entering the system, bots can be deployed to decode commands and get access to the server performing any desired actions. Thus, before implementing any botnet detection approach, proper research within the network seems necessary and loopholes are needed to be figured out.

On the other hand, while comparing with viruses/malware, botnet is unique and highly destructive due to C&C. Computers infected with virus, bot and malware reacts on programming behavior and destructive nature. As virus enters the computer, it starts to spread and perform programmed attacks. Viruses cannot be controlled or changed once installed in the computer. But, bots can be controlled as silent and destructive nature on botmaster's commands. Beside them, attacker can send commands to multiple bots at a time and can be designed to match the network environment such as IRC, HTTP and P2P Botnet. This enables Bot to act wise, be stable, more vulnerable and destructive in comparison to viruses and malwares. Furthermore, about vulnerability, botnet taxonomy itself has huge variation and common aspects within IRC, HTTP and P2P botnets itself. They are explained in next section below.

7.2 IRC, HTTP and P2P

When Botnet architecture is planned, it is taken into consideration for certain network protocol such as IRC, HTTP and P2P. Their target hosts, decoding process, spreading process, vulnerability and behavior are different with each other. Though their individual behaviors are different among each other, there are few things that always stays common among them. In this section, first I am going to discuss shortly about common aspects in botnet as well as detection and after that, detail analysis of individual botnet is presented.

7.2.1 Similarity in Botnet Behavior

While saying about similarity, C&C channel comes in the beginning which we might have already figured out. Whether it is IRC, HTTP or P2P, Botnets are always controlled through C&C channel, but the targeted servers can be single or many depending on the network. Another common thing among botnet is their silent nature. All Botnets can be programmed in a way to act silent or destructive based on attack nature and programming. This makes Botnet unique than viruses too where viruses are simply deployed to the network and any antivirus scan can detect them easily. As an exception, in IRC, this feature is available but not much implemented since they are low budget botnets and targeted mainly for small attacks.

Another common fact among botnet is botmaster's trace. Botnets are controlled through botmaster's commands and botmaster's computer is usually far away from intrusion place. Being far, he can monitor targeted addresses. Not only that, botmaster can hack other IP addresses for the command to deploy and remain undetected. Furthermore, in all botnet intrusion, connection between bots, botmaster and server always needs to be active. Botmaster simply cannot just deploy bots and wait for the result instead need to establish communication and provide the commands.

7.2.2 Similarity in Botnet Detection

After common aspects among botnet behavior, let's go shortly through similarity among detection principles. I have observed that all detection principle uses traffic filtration process in the beginning to reduce the traffic load and separate data traffic such as IRC, HTTP or P2P nodes. After that algorithms are applied to determine the botnet. Another is that, all detection process ends with notification alarm and buzz to notify the administration. They simply don't act without observing the result manually by the observer and observer makes the clearing decision to avoid unwanted loss that might cause by false alarms. So, these were common behavior among Botnet as well as detection processes and in next section I will drive you through comparative analysis between IRC, HTTP and P2P botnets

7.2.3 Comparative Differences

Although this section targets for comparative differences, few botnet behaviors might have similarity among any two botnets and is different from the third one. For an instance, IRC and HTTP botnet operates through centralized server but P2P through decentralized. There are many other mutual similarities we will be discussing in this section targeting mainly for the differences.

To figure out comparative differences, firstly research papers were studied explaining overview of botnet and its taxonomy. Then based on the taxonomy, IRC, HTTP, and P2P Botnet's cases were studied as well as information's were collected. After that, individual ID algorithm and examples were analyzed with false alarms rate, accuracy, datasets, and results. Finally, with the help of collected points, I have merged a table where Botnets are compared with specific properties among each IRC, HTTP and P2P botnet. Table also shows similarities as well as summarized description regarding botnet behavior. Table holds lots of points and thus is divided into multiple sub-tables to explain the content and make it understandable to the reader. Furthermore, for better understanding, similar properties are marked with similar under botnet behavior below.

Properties	IRC	HTTP	P2P
Origin	Started in 1993 "Eggdrop"	Started in 2003 "Phabot"	Started in 2005 "sinit"
Generation	First	Second	Second
Scalability	Good for small amount bots to deploy communication.	Evolution to IRC. Can have a wide amount of bots.	Evolution to IRC and HTTP. Multiple servers means more stable and robustness.

Table 10: Botnet's origin, generation and scalability

Table 10 shows origin, scalability and generation properties of IRC, HTTP and P2P Botnet. It can be clearly observed that IRC is the first-generation Botnet that started in 1993 as 'Eggdrop' whereas HTTP and P2P botnet started in 2003 and 2005 respectively with 'Phabot' and 'sinit'. Regarding scalability, IRC botnet is targeted for small attacks where limited bots can be deployed but on the other hand, HTTP and P2P Botnets are used for larger attacks with wide range of bots that can bring any server down.

Properties	IRC	HTTP	P2P
Transmission Mediums	Applications, messages	Browsers	Network, peers
Command	PUSH command	PULL Command	Both PUSH and PULL command
Communication Port	Uses 6667 IRC port.	Uses TCP protocol and port number 80.	Uses the port number from 10000 to 30000.

Table 11: Botnet's Transmission, Command and Port

Table 11 shows transmission, command and communication port properties among IRC, HTTP and P2P Botnet. In IRC Botnet, transmission happens through text messaging applications by PUSH command whereas with PULL command in internet browsers HTTP Botnet and both PUSH and PULL in P2P Botnet network nodes. Furthermore, about communication port, all IRC Bot uses 6667 ports to host C&C channels where HTTP Bot uses port number 80 in TCP protocol and P2P botnet uses port number between 10000 and 30000 through mesh topology.

Properties	IRC	HTTP	P2P
Server	Centralized with the central server.	Centralized with the central server	Decentralized server.
Interaction	Allows specific interactive behavior with bots. Botmaster knows the specific exploited bot.	Need custom web-server software for interactive behavior	Allows interactive behavior with bots.
Performance	Easy handling and maintenance	Easy handling and maintenance	Hard to maintain and install but results in ID.

Table 12: Botnet's Server, Interaction and Performance

Table 12 shows server type, interactive behavior and performance of IRC, HTTP and P2P botnets. There we can see that; IRC and HTTP uses centralized server whereas P2P uses decentralized. On the other hand, about interaction, IRC is the most interactive botnet and easy to handle from botmaster point of view whereas HTTP is less interactive in compare to IRC. HTTP needs custom web server software for interaction between bots and botmaster. Despite that, HTTP botnet can manage even large attacks with simple codes whereas IRC cannot. HTTP Botnet also uses Bulletproof hosting which is efficient for spamming through online gambling, gaming and dealing. While briefing about P2P Botnet, it allows interactive behavior, but their codes are complex and hard to maintain since all acts as client and server.

Properties	IRC	HTTP	P2P
Latency	Low latency	Low latency	Long latency
Bandwidth	Uses low bandwidth.	Uses low bandwidth.	Uses high bandwidth. (BitTorrent, eMule, Kazaa and so on)
Server Failure	Single point sever failure	Single point sever failure	Single Server failure doesn't affect for intrusion.

Table 13: Botnet's Latency, Bandwidth and Server Failure

Table 13 shows latency behavior, bandwidth and server failure properties in IRC, HTTP and P2P Botnets. All three properties are somehow correspondent to centralized and decentralized server. IRC and HTTP has similar properties with low latency and bandwidth in correspond to P2P botnets. This is beneficial for centralized botnets but on the other hand, centralized Botnets are connected to only one centralized server and when that server is taken down, all bots inside it are passive too. Regarding this situation, P2P botnet doesn't seem to be affected at all. Since there are multiple servers inside the network and when one server is down, bots can rely on other remaining servers.

Properties	IRC	HTTP	P2P
Intrusion target	Effects inbound and outbound IRC messages.	Effects HTTP packets over web browsing through nodes.	Effects network nodes and topology shuts down.
Intrusion Requests	Through IRC PRIV Message	HTTP POST REQUEST for intrusion	First hijacks with small NODE section requests and spreads.

Table 14: Botnet's Intrusion target and requests

Table 14 shows intrusion target, requests and detection properties for IRC, HTTP and P2P Botnet. As IRC Botnet is designed for messaging application, it targets incoming and outgoing message information. On the other hand, HTTP botnet targets for HTTP packets throughout internet browsing history and P2P botnet targets network nodes within the topology. The process is handled in IRC Botnet through IRC PRIV request whereas HTTP POST REQUESTs in HTTP protocol and in P2P network, first small node section is hijacked through C&C requests and starts spreading throughout the whole network.

Properties	IRC	HTTP	P2P
ID Method	HIDS	DIDS	NIDS
ID Efficiency	Easy detection.	Mild	ID is complicated and difficult.
ID False Alarms	Low false alarm and high accuracy	Low false alarm and high accuracy	High false alarm and low accuracy

Table 15: Botnet's ID Method, Efficiency and FP

Table 15 shows ID method, efficiency and FP among IRC, HTTP and P2P Botnets. Since messaging is done through host-based application, IRC takes Host based Intrusion Detection System whereas HTTP takes Domain based IDS and P2P takes Network based IDS simultaneously.

Furthermore, about efficiency, IRC has small architecture and programs which is easy to be detected with high accuracy whereas P2P has complex designs and bot's behavior are more resilient. In IRC, though IDS can easily destroy communication between bots, clients are still infected but as Botmaster is not able to give instruction it is not a threat to the application. Thus, to prevent this situation, clever Botmaster can always try to find nearby possible server and leave backup. About HTTP bots, they are not as easy as IRC to be detected but eventually with traffic analysis, infected packets can be recovered and taken down. On the other hand, P2P bots have complicated design and are also encrypted making ID challenging as well as hard to remove. Furthermore, in

P2P, bots can travel to multiple server and act as silent making the ID task more difficult.

7.3 Intrusion and destruction

Starting from IRC Botnet, many intrusion examples are mentioned in IRC intrusion Record section above where many servers were infected including AOL and ICQ. IRC botnets are targeted mainly for small attack with low budget where certain small tasks like messaging, spy, file recover and so on can be performed and are destroyed. No matter what it can be taken to the next level by deploying huge number of bots to applications and keep the server busy as well as shut down. But the process will be very costly and must deal with all the strong IDS algorithm. So, I think IRC are good to target for single application and just for monitoring user's behavior. HTTP and P2P botnets on the other hand are introduced to reduce attack cost as well as increase efficiency for larger attacks.

Regarding HTTP Botnets, it is dangerous mainly through its domain attacking behavior. In the chapter above, it has already been discussed about internet users in today's world and browsing is being used in almost all corner of the world. Since, many people rely on browsing service, bots can be transferred easily and is able to hijack IP addresses of the device. Furthermore, browsing history, username and passwords can be recovered too leading for huge financial loss as well as blackmailing.

P2P botnets so far, I feel the most vulnerable botnet among all genre due to its fast spreading and complexity in IDS. It can also be seen in intrusion examples above that each botnet can spread to million bots in number and can stay active with army for attack and as well stay silent during the low traffic time being undetected by IDS. Furthermore, these bots rely on server and in P2P there can be multiple servers, so simply bots can be auto-generated even one server is down.

8 Conclusion

Botnet can easily infect and control private computers and networks through C&C channel. Based on servers, bandwidth, network medium, protocol and so on all 3 types of Botnets are designed for individual purposes where IRC Botnet attacks through messaging application, HTTP Botnet targets domain-based packets and P2P Botnet targets network with multiple servers. In chapter 1,2 and 3, Botnet overview with ADT is mentioned. Moreover, in chapter 4, IRC Botnet and its detection methods were discussed whereas HTTP and P2P in chapter 5 and 6 respectively.

In this thesis, I have analyzed Botnet taxonomy from an overview, origin, properties on origination and ID point of view and many more. Thesis holds all information about how and why Botnets are used as well as the reason for evolution to HTTP and P2P. Similarly, examples of ID are discussed in each Botnet chapter to show how detection algorithm works and false alarms are recorded.

To summarize, Botnets are serious cyber security threats now and for future too. To save the network from intrusion, Botnet behavior should be studied, and preventive measures should be applied in time. This thesis provides comparative analysis on individual Botnet and shows guidelines for further research work on algorithms and Botnet attacks in individual protocols.

9 References

[1] S. Chowdhury *et al.*, “Botnet detection using graph-based feature clustering,” *J. Big Data*, vol. 4, no. 1, p. 14, 2017.

[2] Veracode, “WHAT IS A BOTNET?,” *Veracode*, 2018. [Online]. Available: https://www.veracode.com/security/botnet. [Accessed: 14-Apr-2018].

[3] G. Gu, R. Perdisci, J. Zhang, and W. Lee, “Botminer: Clustering analysis of network traffic for protocol-and structure-independent botnet detection,” 2008.

[4] N. Mims, “The Botnet Problem,” in *Computer and Information Security Handbook (Third Edition)*, Elsevier, 2017, pp. 265–274.

[5] Microsoft, “TrojanSpy:Win32/Hesperbot.A.” .

[6] Cisco, “A Cisco guide to Defend against DDoS,” *Cisco*, 2017. [Online]. Available: http://www.cisco.com/c/en/us/about/security-center/guide-ddos-defense.html. [Accessed: 23-May-2017].

[7] Symantec, “PrettyPark Worm,” 2018. [Online]. Available: https://www.symantec.com/security-center/writeup/2000-121508-3334-99?tabid=2. [Accessed: 01-Aug-2018].

[8] J. Grizzard, V. Sharma, C. Nummery, B. B. Kang, and D. Dagon, “Peer-to-peer Botnets: Overview and case study. HotBots’ 07,” in *4th USENIX Symposium on Networked Systems Design & Implementation (NSDI’07). Cambridge (MA)*, 2007, pp. 11–13.

[9] E. Martins, J. Sa, J. Pedros Dias, and J. Pedro Pinto, “Historical list of botnets,” *Botnet Wiki*, 2018. [Online]. Available: http://jpdias.me/botnet-lab//history/historical-list-of-botnets.html. [Accessed: 15-Jan-2018].

[10] P. Mohavedi, “Fast Regularized Least Squares Method for Network Intrusion Detection}No Title,” University of Turku, 2014.

[11] Z. M. Fadlullah *et al.*, “Combating against attacks on encrypted protocols,” in *Communications, 2007. ICC’07. IEEE International Conference on*, 2007, pp. 1211–1216.

[12] N. Fadlullah, Z. M., Taleb, T., Vasilakos, A. V., Guizani, M., & Kato, “DTRAB: Combating against attacks on encrypted protocols through traffic-feature analysis.,” 2010, pp. 1234–1247.

[13] Z. Yan, P. Zhang, and A. V Vasilakos, "A security and trust framework for virtualized networks and software-defined networking," *Secur. Commun. networks*, vol. 9, no. 16, pp. 3059–3069, 2016.

[14] M. Shu, Z., Wan, J., Li, D., Lin, J., Vasilakos, A. V., & Imran, "Security in software-defined networking: Threats and countermeasures.," in *Mobile Networks and Applications*, 2016, pp. 764–766.

[15] D. P. Vinchurkar and A. Reshamwala, "A Review of Intrusion Detection System Using Neural Network and Machine Learning." IJESIT, 2012.

[16] W. Lu and A. A. Ghorbani, "Network anomaly detection based on wavelet analysis," *EURASIP J. Adv. Signal Process.*, vol. 2009, p. 4, 2009.

[17] Jamie, "Recap of Machine Learning For Network-Based IDS Study," *Bizety*, 2016. [Online]. Available: https://www.bizety.com/2016/03/18/machine-learning-network-based-ids/. [Accessed: 23-May-2017].

[18] J. Zhuge, T. Holz, X. Han, J. Guo, and W. Zou, "Characterizing the IRC-based botnet phenomenon," 2007.

[19] C. Li, W. Jiang, and X. Zou, "Botnet: Survey and case study," in *innovative computing, information and control (icicic), 2009 fourth international conference on*, 2009, pp. 1184–1187.

[20] W. Lu and A. A. Ghorbani, "Botnets detection based on IRC-community," *GLOBECOM - IEEE Glob. Telecommun. Conf.*, no. 1, pp. 2067–2071, 2008.

[21] C. Mazzariello, "IRC traffic analysis for botnet detection," *Proc. - 4th Int. Symp. Inf. Assur. Secur. IAS 2008*, pp. 318–323, 2008.

[22] A. Ramachandran and N. Feamster, "Understanding the network-level behavior of spammers," in *ACM SIGCOMM Computer Communication Review*, 2006, vol. 36, no. 4, pp. 291–302.

[23] Cnet, "Hacker hits IRC network undernet with DDoS attack," *Cnet*, 2002. [Online]. Available: https://www.cnet.com/news/hacker-hits-irc-network-undernet-with-denial-of-service-attack/. [Accessed: 10-Jun-2017].

[24] D. Dagon, C. C. Zou, and W. Lee, "Modeling Botnet Propagation Using Time Zones.," in *NDSS*, 2006, vol. 6, pp. 2–13.

[25] W. T. Strayer, R. Walsh, C. Livadas, and D. Lapsley, "Detecting botnets with tight command and control," in *Local Computer Networks, Proceedings 2006*

31st IEEE Conference on, 2006, pp. 195–202.

[26] M. A. Rajab, F. Monrose, and A. Terzis, “On the Effectiveness of Distributed Worm Monitoring.,” in *USENIX Security Symposium*, 2005, p. 15.

[27] G. Gu, J. Zhang, and W. Lee, “BotSniffer : Detecting Botnet Command and Control Channels in Network Traffic,” *Proc. 15th Annu. Netw. Distrib. Syst. Secur. Symp.*, vol. 53, no. 1, pp. 1–13, 2008.

[28] S. E. Mathew, A. Ali, and J. Stephen, “Genetic Algorithm based Layered Detection and Defense of HTTP Botnet,” *Proc. 11th Annu. Conf. Genet. Evol. Comput. GECCO 09*, vol. 5, no. 1, pp. 128–133, 2014.

[29] J.-S. Lee, H. Jeong, J.-H. Park, M. Kim, and B.-N. Noh, “The Activity Analysis of Malicious HTTP-Based Botnets Using Degree of Periodic Repeatability,” in *2008 International Conference on Security Technology*, 2008, pp. 83–86.

[30] R. F. M. Dollah, M. A. Faizal, F. Arif, M. Z. Mas’ud, and L. K. Xin, “Machine Learning for HTTP Botnet Detection Using Classifier Algorithms,” *J. Telecommun. Electron. Comput. Eng.*, vol. 10, no. 1–7, pp. 27–30, 2018.

[31] S. S. Garasia, D. P. Rana, and R. G. Mehta, “HTTP botnet detection using frequent patternset mining,” *Proc. [Ijesat] Int. J. Eng. Sci. Adv. Technol.*, vol. 2, pp. 619–624, 2012.

[32] E. Messmer, “The most dangerous botnets of 2012,” *Network World US*, 2012. [Online]. Available: https://www.computerworlduk.com/galleries/security/most-dangerous-botnets-of-2012-3409375/.

[33] P. Paganini, “Http-botnets: The dark side of an standard protocol,” 2013.

[34] H. Choi, H. Lee, H. Lee, and H. Kim, “Botnet detection by monitoring group activities in DNS traffic,” in *Computer and Information Technology, 2007. CIT 2007. 7th IEEE International Conference on*, 2007, pp. 715–720.

[35] G. K. Venkatesh and R. A. Nadarajan, “HTTP botnet detection using adaptive learning rate multilayer feed-forward neural network,” in *IFIP International Workshop on Information Security Theory and Practice*, 2012, pp. 38–48.

[36] T. Karagiannis, A. Broido, N. Brownlee, K. Claffy, and M. Faloutsos, “File-sharing in the Internet: A characterization of P2P traffic in the backbone,” *Univ. California, Riverside, USA, Tech. Rep*, 2003.

[37] G. Gu, R. Perdisci, J. Zhang, and W. Lee, “BotMiner: Clustering Analysis of

Network Traffic for Protocol-and Structure-Independent Botnet Detection.”

[38] S. Khattak, N. R. Ramay, K. R. Khan, A. A. Syed, and S. A. Khayam, “A taxonomy of botnet behavior, detection, and defense,” *IEEE Commun. Surv. tutorials*, vol. 16, no. 2, pp. 898–924, 2014.

[39] D. Dagon, G. Gu, C. P. Lee, and W. Lee, “A taxonomy of botnet structures,” in *Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual*, 2007, pp. 325–339.

[40] J. Nazario, “Bot and botnet taxonomy,” *Comput. Secur. Institute. Comput. Secur. Inst. Secur. Exch.*, 2008.

[41] D. Liu, Y. Li, Y. Hu, and Z. Liang, “A P2P-botnet detection model and algorithms based on network streams analysis,” in *Future Information Technology and Management Engineering (FITME), 2010 International Conference on*, 2010, vol. 1, pp. 55–58.

[42] P. Narang, S. Ray, C. Hota, and V. Venkatakrishnan, “Peershark: detecting peer-to-peer botnets by tracking conversations,” in *Security and Privacy Workshops (SPW), 2014 IEEE*, 2014, pp. 108–115.

[43] C. Dillon, “Peer-to-peer botnet detection using netflow.” Jul, 2014.

[44] L. Zhou, Z. Li, and B. Liu, “P2P traffic identification by TCP flow analysis,” in *Networking, Architecture, and Storages, 2006. IWNAS’06. International Workshop on*, 2006, p. 2--pp.

[45] C. Rossow *et al.*, “Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets,” in *Security and Privacy (SP), 2013 IEEE Symposium on*, 2013, pp. 97–111.