Botnet-A Network Threat

Sonal P.Patil
Student, M. Tech 2nd year,
TIT, Bhopal

Swatantra Kumar
Software Engineer, OSS Cube Solutions,
Pvt. Ltd., Mumbai

ABSTRACT

Botnet are network threats that generally occur from cyber attacks, which results in serious threats to our network assets and organization's properties. Botnets are collections of compromised computers (Bots) which are remotely controlled by its originator (BotMaster) under a common Command-and-Control (C&C) infrastructure. Among the various forms of malware, botnets are emerging as the most serious threat against cyber-security as they provide a distributed platform for several illegal activities such as launching distributed denial of service attacks against critical targets, malware dissemination, phishing, and click fraud. The most important characteristic of botnets is the use of command and control channels through which they can be updated and directed. The target of the botnet attacks on the integrity and resources of users might be multifarious; including the teenagers evidencing their hacking skills to organized criminal syndicates, disabling the infrastructure and causing financial damage to organizations and governments. In this context, it is crucial to know in what ways the system could be targeted. The major advantage of this classification is to identify the problem and find the specific ways of defense and recovery. This paper aims to provide a concise overview of major existing types of Botnets on the basis of attacking techniques.

General Terms

Botnets are emerging as the most significant threat facing online ecosystems and computing assets. Malicious botnets are distributed computing platforms predominantly used for illegal activities such as launching Distributed Denial of Service (DDoS) attacks, sending spam, trojan and phishing emails, illegally distributing pirated media and software, force distribution, stealing information and computing resource, ebussiness extortion, performing click fraud, and identity theft. The high light value of botnets is the ability to provide anonymity through the use of a multi-tier command and control (C&C) architecture. Moreover, the individual bots are not physically owned by the botmaster, and may be located in several locations spanning the globe. Differences in time zones, languages, and laws make it difficult to track malicious botnet activities across international boundaries. This characteristic makes botnet an attractive tool for cybercriminals, and in fact poses a great threat against cyber security.

Keywords

Botnet, Bots, Botnet Detection.

1. INTRODUCTION

Botnet is a network of compromised computers called "Bots" under the remote control of a human operator called "Botmaster". The term "Bot" is derived from the word "Robot"; and similar to robots, bots are designed to perform some predefined functions in automated way. In other words, the individual bots are software programs that run on a host computer allowing the botmaster to control host actions remotely. And here the term net

Botnets are one of the most dangerous species of network-based attack today because they involve the use of very large, coordinated groups of hosts for both brute-force and subtle attacks. A collection of bots, when controlled by a single command and control infrastructure, form what is called a botnet. Botnets obfuscate the attacking host by providing a level of indirection, the attack host is separated from its victim by the layer of zombie hosts, and the attack itself is separated from the assembly of the botnet by an arbitrary amount of time. The technological advancements are pushing the human life towards ease and trouble simultaneously. Emerging information technologies have made access to information so easy that was never before. But on the other hand, it has worsened the security level. BOTNETS are proving to be the most recent and disastrous threat to the field of information technology. The understanding of a layman about Botnets is that it is a network facilitating the malicious attacks on the user machines but technically speaking "Botnets are a collection of computers on which ,a software, 'bot', is automatically installed without user intervention and are remotely controlled via command and control server". Despite of the fact that this network can be implied both for nefarious and beneficial purposes, its extensive deployment in the criminal and destructive purposes has made the title 'botnets' tantamount to malware. An active Botnet initializes its attack by first exploiting vulnerabilities in the user computers. It then downloads the malicious binary and executes it locally. This program logs on to the Command and Control Server (C & C) and notifies its Host, commonly known as 'Botmaster' or 'Botherder', that the computer is now converted to a 'Bot'. It can now be used to forward its affect to other computers by repeating the same procedure. The major difference between botnets and other security threats is that a botmaster communicates regularly with the bots either via centralized communication channel or decentralized network. These bots perform any type of destruction on receiving the commands from the botmaster. These botmasters send the commands, control all the bots, and then can attack a victim as a unit. Botnets are developing at a very fast rate making it difficult to detect and recover from their side effects. However, some of their types extensively deployed can be classified to provide for their remedy. This report mainly deals with three major types of botnets: IRC botnets, peer-to-peer and HTTP botnets and suggests some techniques to identify and detect them. Section 1 gives an introduction of botnets. Section 2 reviews their history and topologies. Section 3 is all about their lifecycle, Botnet Command and Control, Botnet Topologies according to the Command-and-Control(C&C) channel and Botnet life cycle. Three major types of botnets and their detection scenarios are considered in Section 3.1.1, 3.1.2 and 3.1.3 respectively. Section 3.2 proposed botnet detection framework & components; Section 4 proposes some of expected advances in this particular field as future work. Section 5 is dedicated to the overall conclusion of our study.

2. TYPESET TEXT

2.1 HISTORY OF MALICIOUS BOTS

Before the evolution of Botnets; the major sources of malware were viruses, worms, Trojan Horses that used to affect only a single machine. With the evolution of Botnets; the concept of destruction was enhanced from a single machine to a network as a whole. The history of undertaking botnets for destruction roughly dates back to 1990. Prior to this, botnets were the major sources of maintaining control of the IRC channels. Their mischievous applications mainly took advantage of the centralized control of IRC for command and control. But centralized control structure was relatively easy to discover and track. Due to insecure nature of IRC botnets; they completely changed their structure form centralized to a peer-to-peer nature, which is a decentralized control structure. This ultimately makes it much harder to spy the communication among the bots and to track their origin. The most recent improvement is again the implementation of centralized C&C in HTTP botnets; but here the distinguishing feature is that the Botnets periodically connect and disconnect with the bot master. This further aggravates the problem of detection [7].

2.2 COMMAND AND CONTROL CHANNEL

The backbone of botnet is command and control channel; which is responsible for setting up the botnet, controlling the activities of the bots, issuing commands, and ultimately reaching the goals [2]. The command and control channel is stable during the operation of botnets i.e. once a botnet is established; the command and control channel remain the same throughout its operation. But on the other hand, once a C&C channel is detected; then the whole botnet is exposed.

2.3 BOTNET TOPOLOGIES

According to the Command-and-Control(C&C) channel, we categorized Botnet topologies into two different models, the Centralized model and the Decentralized model [1].

*A. **Centralized model***

The oldest type of topology is the centralized model. In this model, one central point is responsible for exchanging commands and data between the BotMaster and Bots. Many well-known Bots, such as AgoBot, SDBot, Zotob and RBot used this model. In this model, BotMaster chooses a host (usually high bandwidth computer) to be the central point (Command-and-Control) server of all the Bots. The C&C server runs certain network services such as IRC or HTTP. The main advantage of this model is small message latency which cause BotMaster easily arranges Botnet and launch attacks. Since all connections happen through the C&C server, therefore, the C&C is a critical point in this model. In other words, C&C server is the weak point in this model. If somebody manages to discover and eliminates the C&C server, the entire Botnet will be worthless and ineffective. Thus, it becomes the main drawback of this model.

Since IRC and HTTP are two common protocols that C&C server uses for communication, we consider Botnets in this model based on IRC and HTTP. Figure 1 shows the basic communication architecture for a Centralized model.

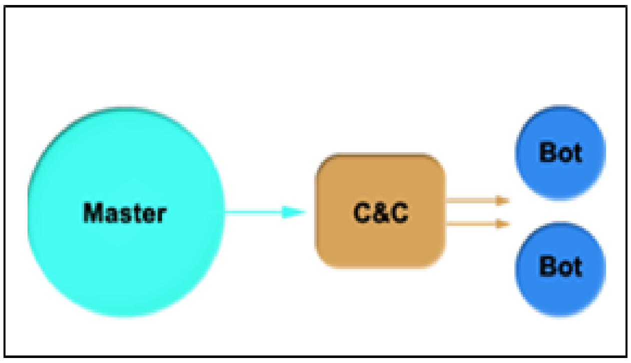


Fig 1: Command and control architecture of a Centralized model

*1) **Botnet based on IRC**:* The IRC is a form of real-time Internet text messaging or synchronous conferencing [8]. The protocol is based on the Client-Server model, which can be used on many computers in distributed networks. Some advantages which made IRC protocol widely being used in remote communication for Botnets are: (1) Low latency communication; (2) Anonymous real-time communication; (3) Ability of Group (many-to-many) and Private (one-to-one) communication; (4) simple to setup and (5) simple commands. The basic commands are connect to servers, join channels and post messages in the channels; (6) Very flexibility in communication. Therefore IRC protocol is still the most popular protocol being used in Botnet communication.

In this model, BotMaster's can command their Bots as a whole or command a few of the Bots selectively using one-to-one communication. The C&C server runs IRC service that is the same with other standard IRC service. BotMaster usually creates a designated channel on the C&C servers where all the Bots will connect, awaiting commands in the channel which will instruct each connected Bot to do the BotMaster's command.

*2) **Botnet based on HTTP**:* The HTTP protocol is another popular protocol used by Botnets. Since IRC protocol within Botnets became well-known, more internet security researchers gave attention to monitoring IRC traffic to detect Botnet. Consequently, attackers started to use HTTP protocol as a Command-and-Control communication channel to make Botnets become more difficult to detect. The main advantage of using the HTTP protocol is hiding Botnets traffics in normal web traffics, so it can easily bypasses firewalls with port-based filtering mechanisms and avoid IDS detection. There are some known Bots using the HTTP protocol, such as Bobax, ClickBot [8] and Rustock. Guet al pointed out that the HTTP protocol is in a "pull" style and the IRC is in a "push" style.

*B. **Decentralized Model***

Due to major disadvantage of Centralized model – Central Command-and-Control(C&C) attackers started to build alternative Botnet communication system that is much harder to discover and to destroy. Hence, they decided to find a model in which the communication system does not heavily depending on few selected servers and even discovering and destroying a number of Bots. As a result, attackers exploit the idea of Peer-to-Peer (P2P) communication as a Command-and-Control(C&C) pattern which is more resilient to failure in the network. The P2P based C&C model will be used dramatically in Botnets in the near future, and definitely Botnets that use P2P based C&C model impose much bigger challenge for defense of networks. Since P2P based communication is more robust than Centralized C&C communication, more Botnets will move to use P2P protocol for their communication.

In P2P model, as shown in Figure 2, there is no Centralized point for communication. Each Bot keeps some connections to the other Bots of the Botnet. Bots act as both Clients and servers. A new Bot must know some addresses of the Botnet to connect there. If Bots in the Botnet are taken offline, the Botnet can still continue to operate under the control of BotMaster. P2P Botnets aim at removing or hiding the central point of failure which is the main weakness and vulnerability of Centralized model

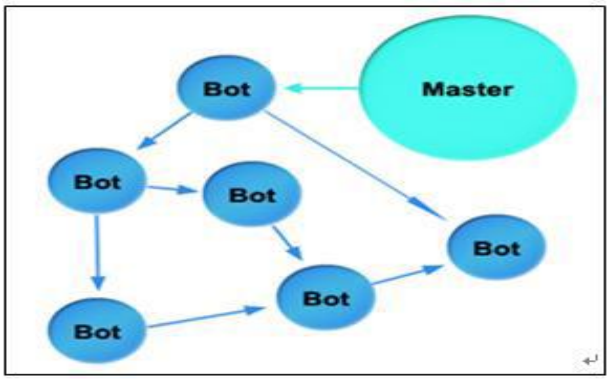


Fig 2: Example of Peer-to-peer Botnet Architecture

3. BOTNET LIFE CYCLE

A typical botnet can be created and maintained in five phases including: initial infection, secondary injection, connection, malicious command and control, update and maintenance. This life-cycle is depicted in Fig. 3

A typical botnet can be created and maintained in five phases including: initial infection, secondary injection, connection, malicious command and control, update and maintenance. This life-cycle is depicted in Fig 3.

During the initial infection phase, the attacker, scans a target subnet for known vulnerability, and infects victim machines through different exploitation methods. After initial infection, in secondary injection phase, the infected hosts execute a script known as shell-code. The shell-code fetches the image of the actual bot binary from the specific location via FTP, HTTP, or P2P. The bot binary installs itself on the target machine. Once the bot program is installed, the victim computer turns to a "Zombie" and runs the malicious code. The bot application starts automatically each time the zombie is rebooted. In connection phase, the bot program establishes a command and control (C&C) channel, and connects the zombie to the command and control (C&C) server. Upon the establishment of C&C channel, the zombie becomes a part of attacker's botnet army. After connection phase, the actual botnet command and control activities will be started. The botmaster uses the C&C channel to disseminate commands to his bot army.

Bot programs receive and execute commands sent by BotMaster. The C&C channel enables the botmaster to remotely control the action of large number of bots to conduct various illicit activities. Last phase is to maintain bots lively and updated. In this phase, bots are commanded to download an updated binary [4].Bot controllers may need to update their botnets for several reasons. For instance, they may need to update the bot binary to evade detection techniques, or they may intend to add new functionality to their bot army. Moreover, sometimes the updated binary move the bots to a different C&C server. This process is called server migration and it is very useful for botmasters to keep their botnet alive. BotMaster try to keep their botnets invisible and portable by using Dynamic DNS (DDNS) which is a resolution service that facilitates frequent updates and changes in server locations. In case authorities disrupt a C&C server at a certain IP address, the botmaster can easily set up another C&C server instance with the same name at a different IP address. IP address changes in C&C servers propagate almost immediately to bots due short time-to-live (TTL) values for the domain names set by DDNS providers. Consequently, bots will migrate to the new C&C server location and will stay alive.

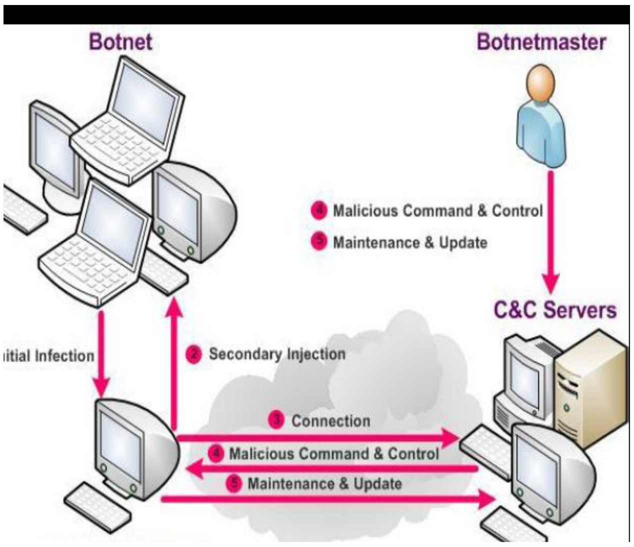


Fig 3: Botnet Life-cycle

The success of any process mainly lies in how well the sequence of steps is organized. The major reason of dramatic success and spread of botnets is their well organized and planned formation, generation and propagation. The lifecycle of a botnet from its birth to disastrous spread undergoes the following phases [2]:

1. Bot-herder configures initial bot parameters.
2. Registers a DDNS.
3. Register a static IP.
4. Bot-herder starts infecting victim machines either directly through network or indirectly through user interaction.
5. Bots spread.
6. Bot joins the Botnet through C&C server.
7. Bots are used for some activity (DDoS, Identity Theft etc.)
8. Bots are updated through their Bot operator which issues update commands.

3.1 TYPES OF BOTNETS

There is a variety of botnets causing the mass destruction. As already discussed in section II, the three major categories that we have considered in our study depend on the type of command and control they are based on [2]. They are as follows:

- IRC botnets
- P2P botnets
- HTTP botnets

Now we will consider each one of them to briefly view their operation and detection mechanism.

3.1.1 IRC

The IRC (Internet Relay Chat) protocol was initially designed for real-time Internet text messaging. The building ground of IRC is TCP/IP protocol. It works by making a central location and then all the required users (clients) connect to that central location; and that central location is called ***server;*** while anything except server is called ***client.*** Clients are distinguished from each other by their nickname; which is a string composed of 9 characters. Any server must know the real name of the host the client is running on, the username of the host the client is running on, the user name of client on that host, and the corresponding server.

As IRC came into extensive use several variations in the protocol and structure were adopted. Automated clients called bots emerged as a new concept and the success was obvious. They served as a permanent point of contact for information exchange. With their popularity, their deployment in several unexpected tasks increased manifold. One of these was the emergence of botnets for nefarious purposes.

This emergence grew into a massive network that allow its operators to use it for running games, file distribution, or use it for user misbehavior. [2] the most vulnerable feature of an IRC is its server. The IRC channel operator is connected to this server. If the server is crashed due to some reason, then the connection of this operator would automatically die and another member from the same channel would automatically be assigned the server status. This behavior proved to be disastrous, and allowed any user to snatch the server's honor, and therefore use the channel according to its own will.

The IRC bot is an assembly of programmed codes that behave as a client in an IRC channel. But unlike the traditional clients providing interactive access, it performs self-propelled functions.

The key feature of pioneer legitimate IRC bots called botnets; was to allow secure assignment of privileges between bots, sharing of user/ban lists and to control floods. This allowed the IRC operators to utilize the congregated power of many modules of bots together.

IRC Detection Techniques

A lot of techniques have been proposed for IRC Botnet detection. The basis of all these techniques is hounding of packets either at network layer or application layer.

In the mechanism of detection is suggested on the network layer level. Here the hierarchy between routers and the IRC server is explored in bottom-up manner i.e. the tracking initiates from the victim and follows the path of infecting routers till the origin (bot-herder).

The author has proposed a frame work in which sniffs the network traffic, filters it on the basis of application layer protocol, and then segregates them into either righteous or saboteur IRC traffic just by contemplating the IRC chat contents. The separating foundation between a normal human and botnet conversation is that the human language is alternating while the Botnet conversation is repeating.

It presents a pipelined approach which accomplishes the detection procedure in a number of steps. First it separates the black and white list traffic based on the DNS queries; this separated traffic is classified according to applications i.e. extract chat-like traffic. Next pair wise correlation of the traffic flows is done to identify similar traffic considering it to be originating from same botnet.

The study of these IRC detection techniques reveals that choice of the suitable detection technique depends on the required scenario. If the solution has to be managed at the network layer, serves as the best option; while on application layer and serve the purpose. Regardless of their applications, each technique has its respective shortcomings which leave a large room for further suggestions and research.

3.1.2 P2P BOTNETS

Preliminary botnet architecture was based upon centralized architecture but that was much prone to detection; as the entire botnet can be apprehended just by tracking down a single central command [1].Botnet was referred to a cluster of computer infected by the computer virus, each of which is so called as "bot". Real hackers behind the bots took advantage of such communication to command and control the bots to send spam mails, steal valuable ID and password of on-line game or cause DDoS. With technologies evolves, Botnet also developed various structures such as IRC, HTTP and P2P, etc. The P2P botnet, a new type originating from botnet, operated as in Figure 1 by imitating Peer-to-Peer (P2P) technologically. First, the P2P botnet imitated P2P applying multiple main control to avoid single point failure.

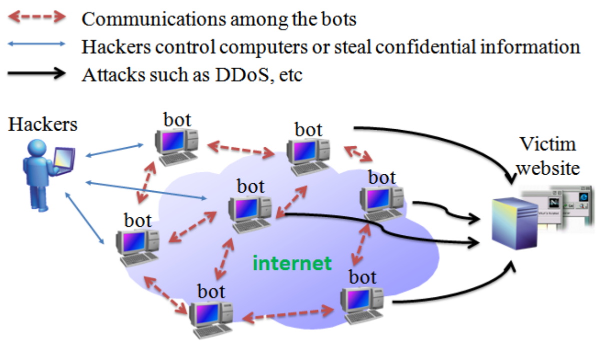

Communications among the bots
Hackers control computers or steal confidential information
Attacks such as DDoS, etc
Hackers
bot
bot
bot
bot
bot
bot
bot
internet
Victim website

Fig 4: Diagram of P2P botnet Operation

Plus, it used encryption technology, making it impossible for us to analyze communication contents and discover botnet communication in the legal network flow. At present, internet hackings can be detected by misuse detection and anomaly detection. The misuse detection was the method of signature comparison to judge rascal software only by in-depth scanning the communication contents. It worked in detecting unencrypted IRC bot, but not P2P botnet. Anomaly detection, the other primary technology, was also seemed ineffective in botnet, since it need resulted data to define normal and abnormal behaviors, causing errors in false positive and false negative judgment inevitably [8].

Comparing with misuse detection, anomaly detection was better since it only judged characters of communications in terms of behaviors and statistics without reading encrypted communication contents, meaning that even malice communication after being encrypted can also be used. Since the method focused on the object's behavior characters, using P2P botnets original characters to sort out its natures statistically was an important task. The research suggested the detection method on the basis of the following three hypotheses: communication via P2P botnet imitated P2P structure to set up numerous sessions; bot sessions kept on transmitting data to maintain the malicious network works; and botnet communication used data at minimum level as much as possible to keep its privacy. In order to improve accuracy of anomaly detection, not only the necessary data under the experiment internet environment were collected, but also data mining technology was used to make judgment more accurately.

To overcome this drawback, a rather new technology in the field of Botnets is peer-to-peer Botnets; where a peer (host) can act as both client and server alternatively. To enter the network a peer can connect to any other peer of the network using its IP address that was already present in its database. Finally when this peer is part of the network; it continually updates its database by interacting with other peers. Using this approach when any peer tries to send commands to the botnet, it sends a library call to its database to get the addresses of other bots; thus acting as commander and controller of the P2P botnet. This Commander and Controller now send orders that are to be followed by the remaining peers of the network.

To track down a peer-to-peer network, initially the simplest possible solution was for the hacker to enter the botnet by pretending to be a new bot. This newly entered bot will now be able to connect to any other peer of the network and thus be able to track down its activities. The biggest disadvantage of this approach is that the intruder can monitor the activity and thus track down only a single peer; the entire botnet activity can neither be monitored nor can be tracked down immediately. The entire Botnet tracking is obviously a time consuming operation [1].

3.1.3 HTTP BOTNETS

The most recent Botnet till date is HTTP botnet. It works by exchanging web requests using port 80. It sets up its communication with certain URL's using internet with an HTTP message. This HTTP message contains unique identifiers for the bots. The server under consideration will reply to these HTTP messages with further investigation commands (e.g. GET). This interrogating command ultimately becomes the reason of downloading the infecting malicious commands. Again it uses the centralized command and control channel as IRC botnet uses but a few advantages compared to IRC exists:

• Here the command and control server is web server as compared to IRC botnets where IRC serves as the C&C.

• In IRC bot once connected to C&C doesn't disconnect but here the bots regularly connects with the server after regular intervals of time; which is set by the web server.

The traffic of the HTTP botnets flows with the regular traffic. However, the bot packets are different from normal packets making the detection procedure easy [7]. Discusses the most commonly deployed detection technique for HTTP botnets. Here a degree of periodic repeatability (DPR) is employed. This parameter represents the repeated reconnection of bots with botmaster after regular interval that is configured by the botmaster. The more number of times, same client connects to the same server after same interval of time, depicts greater probability of a client being a bot and server being a botmaster.

More work on several other techniques is underway to timely detect the modern HTTP botnet attacks.

3.2 PROPOSED BOTNET DETECTION FRAMEWORK AND COMPONENTS

Our proposed framework is based on passively monitoring network traffics. Consequently this model is not provided for detecting botnet at the very moment when hosts are infected with bots. This model is based on the definition of P2P botnets that multiple bots within the same botnet will perform similar communication patterns and malicious activities. Figure 6 shows the architecture of our proposed botnet detection system, which consist of 4 main components: Filtering, Traffic Monitoring, Malicious Activity Detector and Analyzer. Filtering is responsible to filter out irrelevant traffic flows. The main benefit of this stage is reducing the traffic workload and makes application classifier process more efficient. Malicious activity detector is responsible to analyze the traffics carefully and try to detect malicious activities that internal host may perform and separate those hosts and send to next stage. Traffic Monitoring is responsible to detect the group of hosts that have similar behavior and communication patterns by inspecting network traffics. Analyzer is responsible for comparing the results of previous parts (Traffic Monitoring and Malicious Activity Detector) and finding hosts that are common on the results of both parts.

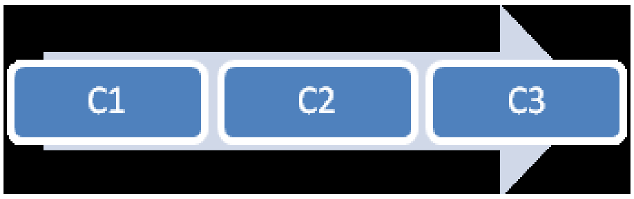


Fig 5: Traffics filtering stages

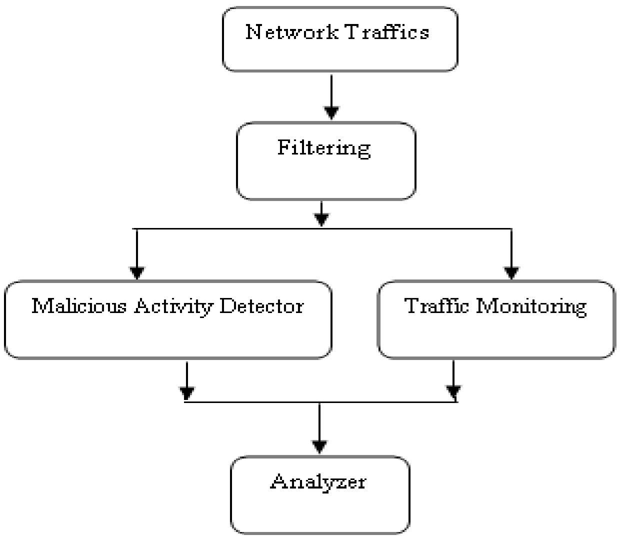


Fig 6: Architecture overview of our proposed detection framework

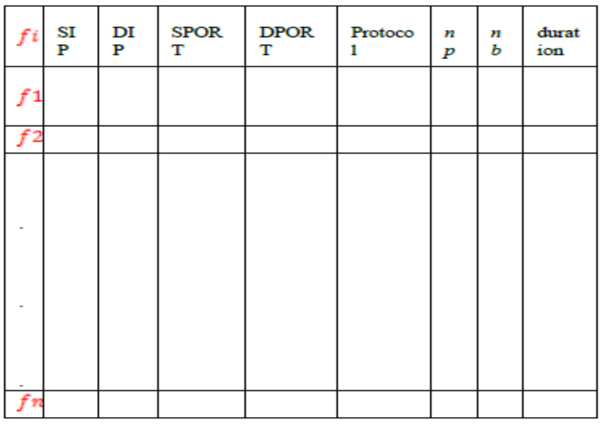
A. *Filtering*

Filtering is responsible to filter out irrelevant traffic flows. The main objective of this part is for reducing the traffic workload and makes the rest of the system perform more efficiently. Figure 6 shows the architecture of the filtering.

In C1, we filter out those traffics which targets (destination IP address) are recognized servers and will unlikely host botnet C&C servers. For this purpose we used the top 500 websites on the web (http://www.alexa.com/topsites), which the top 3 are google.com, facebook.com and yahoo.com. In C2, we filter out traffics that are established from external host towards internal hosts. In C3, we filter out handshaking processes (connection establishments) that are not completely established. Handshaking is an automated process of negotiation that dynamically sets parameters of a communications channel established between two entities before normal communication over the channel begins. It follows the physical establishment of the channel and precedes normal information transfer. A good example that usually we face with that in network is TCP protocol operations. To establish a connection, TCP uses a three-way handshake; in this case we filter out the traffics that TCP handshaking have not completed. Like a host sends SYN packets without completing the TCP handshake. Based on our experience these flows are mostly caused by scanning activities.

B. *Traffic Monitoring*

Traffic Monitoring is responsible to detect the group of hosts that have similar behavior and communication pattern by inspecting network traffics. Therefore we are capturing network flows and record some special information on each flow. We are using Audit Record Generation and Utilization System (ARGUS) which is an open source tool for monitoring flows and record information that we need in this part. Each flow record has following information: Source IP(SIP) address, Destination IP(DIP) address, Source Port(SPORT), Destination Port(DPORT), Duration, Protocol, Number of packets(*np*) and Number of bytes(*nb*) transferred in both directions.



f_i	SIP	DIP	SPORT	DPORT	Protocol	n_p	n_b	duration
f_1								
f_2								
. . .								
f_n								

Fig 7: Recorded information of network flows using

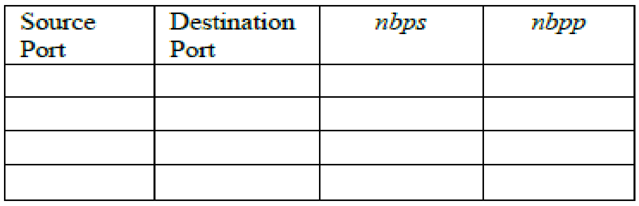
Then we insert this information on a data base like Figure 2, which are network flows. After this stage we specify the period of time which is 6 hours and during each 6 hours, all n flows that have same Source IP, Destination IP, Destination port and same protocol (TCP or UDP) are marked and for each network flow (row) we calculate Average number of bytes per second and Average number of bytes per packet:

a) Average number of bytes per second(nbps) = Number of bytes/ Duration

b) Average number of bytes per packet(nbpp) = Number of Bytes/ Number of Packets

Then, we insert this two new values (*nbps* and *nbpp*) including SIP and DIP of the flows that have been marked into another database, similar to figure 3 . Therefore, during the specified period of time (6 hours), we might have a set of database, which each of these databases have same SIP, DIP, DPORT and protocol (TCP/UDP). We are focusing just at TCP and UDP protocols in this part.

As we mentioned earlier, the bots belonging to the same botnet have same characteristics. They have similar behavior and communication pattern, especially when they want to update their commands from botmasters or aim to attack a target; their similar behaviors are more obvious.



Source Port	Destination Port	*nbps*	*nbpp*

Fig 8: Database for analogous flows

Therefore, next step is to looking for groups of Databases that are similar to each other. For finding similar communication flows among databases, one solution is using clustering algorithm like X-means clustering algorithm. X-means is one of the most famous clustering algorithms.

We proposed a simple solution for finding similarities among group of databases. For each database we can draw a graph in x-y axis, which x-axis is the Average Number of Bytes per Packet (*nbpp*) and y-axis is Average Number of Byte Per Second (*nbps*). (X, Y)= (bpp, bps)

For example, in database (), for each row we have *nbpp* that specify x-coordinate and have *nbps* that determine y-coordinate. Both x-coordinate and y-coordinate determine a point (x,y) on the x-y axis graph. We do this procedure for all rows (network flows) of each database. At the end for each database we have number of points in the graph that by connecting those points to each other we have a curvy graph. We have an example, figure 7, for two different databases based on data in our lab that their graphs are almost similar to each other

Next step is comparing different x-y axis graphs, and during that period of time (each 6 hours) those graphs that are similar to each other are clustered in same category. The results will be some x-y axis graphs that are similar to each other. Each of these graphs is referring to their corresponding databases in previous step. We have to take record of SIP addresses of those hosts and send the list to next step for analyzing.

C. *Malicious Activity Detector*

In this part we have to analyze the outbound traffic from the network and try to detect the possible malicious activities that the internal machines are performing. Each host may perform different kind of malicious activity but Scanning, Spamming, Binary downloading and exploit attempts are the most common and efficient malicious activities a botmaster may command their bots to perform. In this report we just focus on scanning and spam-related activities. The outputs of this part are the list of hosts which performed malicious activities.

1) Scanning: Scanning activities may be used for malware propagation and DOS attacks. There has been little work on the problem of detecting scan activities. Most scan detection has been based on detecting N events within a time interval of T seconds. This approach has the problem that once the window size is known, the attackers can easily evade detection by increasing their scanning interval. Snort are also use this approaches. Snort version 2.0.2 uses two preprocessors. The first is packet-oriented, focusing on detecting malformed packets used for —stealth scanning by tools such as nmap. The second is connection oriented. It checks whether a given source IP address touched more than X number of ports or Y number of IP addresses within Z seconds. Snort's parameters are tunable, but it suffers from the same drawbacks as Network Security Monitor (NSM) since both rely on the same metrics. Other works that are focusing on scan detection is by Stanford et al. on Stealthy Probing and Intrusion Correlation Engine (SPICE). SPICE is focusing on detecting stealthy scans, especially scans that spread across multiple source addresses and execute at very low rates. In SPICE there are anomaly scores for packets based on conditional probabilities derived from the SIP and DIP and ports. It uses simulated annealing to cluster packets together into port scan using heuristics that have developed from real scans. An important need in our system is prompt response, however reaching to our goals which are promptness and accuracy in detecting malicious scanners is a difficult task. Another solution is also using Threshold Random Walk (TRW), an online detection algorithm. TRW is based on sequential hypothesis testing.
After assessing different approaches for detecting scanning activities, the best solution for using in this part is Statistical sCan Anomaly Detection Engine(SCADE), a snort processor plug-in system which has two modules, one for inbound scan detection and another one for detecting outbound attack propagation.
a) Inbound Scan Detection (ISD): In this part SCADE has focused on detection of scan activities based on ports that are usually used by malware. One of the good advantages of this procedure is that it is less vulnerable to DOS attacks, mainly because its memory trackers do not maintain per-external-source-IP. SCADE here just tracks scans that are targeted to internal hosts. The bases of Inbound Scan Detection are on failed connection attempts. SCADE in this part has defined two types of ports: High-Severity (hs) ports which representing highly vulnerable and commonly exploited services and low-severity (ls) ports. For make it more applicable in current situation SCADE focused on TCP and UDP ports as high-secure and all other as low-secure ports. There are different weights to a failed scan attempt for different types of ports.

The warning for ISD for a local host is produced based on an anomaly score that is calculated as based on this formula:

$$S = (w1Fhs+w2Fls)$$

Fhs: indicate numbers of failed attempts at high-severity ports.
Fls : shows numbers of failed attempts at low-severity ports.

b) Outbound Scan Detection (OSD): OSD is based on a voting scheme (AND, OR or MAJORITY). SCADE in this part has three parallel anomaly detection models that track all outbound connection per internal host:

• Outbound scan rate (s1): Detects local hosts that perform high-rate scans for many external addresses.

• Outbound connection failure rate (s2): Detects unusually high connection fail rates, with sensitivity to HS port usage. The anomaly score s2 is calculated based on this formula:

$$S2 = \frac{(w1Fhs+w2Fls)}{C}$$

Fhs: indicate numbers of failed attempts at high-severity ports.
Fls : shows numbers of failed attempts at low-severity ports.
C : is the total number of scans from the host within a time window.

Normalized entropy of scan target distribution (s3): Calculates a Zipf (power-law) distribution of outbound address connection patterns. A consistently distributed scan target model provides an indication of a possible outbound scan. It is used an anomaly scoring technique based on normalized entropy to identify such candidates:

$$S3= \frac{H}{Ln(m)}$$

H: is the entropy of scan target distribution
m : is the total number of scan targets
pi : is the percentage of the scans at target

2) ***Spam-related Activities****:* E-mail spam, known as Unsolicited Bulk Email (UBE), junk mail, is the practice of sending unwanted email messages, in large quantities to an indiscriminate set of recipients. More than 95% of email on the internet is spam, which most of these spams are sent from botnets. A number of famous botnets which have been used specially for sending spam are Storm Worm which is P2P botnet and Bobax that used Http as its C&C.

A common approach for detecting spam is the use of DNS Black/Black Hole List (DNSBL) such as (http://www.dnsbl.info/dnsbl-list.php). DNSBLs specify a list of spam senders' IP addresses and SMTP servers are blocking the mail according to this list. This method is not efficient for bot-infected hosts, because legitimate IP addresses may be used for sending spam in our network. Creation or misuse of SMTP mail relays for spam is one of the most well-known exploitation of botnets. As we know user-level client mail application use SMTP for sending messages to mail server for relaying. However for receiving messages, client application usually use Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) to access the mail box on a mail server. Our idea in this part is very simple and efficient. Our target here is not recognizing which email message is spam, though for detecting group of bots that sending spam with detecting similarities among their actions and behaviors. Therefore the content of emails from internal network to external network is not important in our solution. All we want to do is determining which clients have been infected by bot and are sending spam. For reaching to this target, we are focusing on the number of emails sending by clients to different mail servers. Based on our experience in our lab, using different external mail servers for many times by same client is an indication of possible malicious activities. It means that it is unusual that a client in our network send many emails to the same mail server (SMTP server) in the period of time like one day. Therefore, we are inspecting outgoing traffic from our network(gateway), and recording SIP and DIP of those traffics that destination ports are 25(SMTP) or 587(Submission) in the database. Based on network flows between internal hosts and external computers(SIP belong to mail servers) and the number of times that it can happen we can conclude which internal host is behaving unusual and are sending many emails to different or same mail servers.

D. Analyzer

Analyzer which is the last part of our proposed framework for detection of botnets is responsible for finding common hosts that appeared in the results of previous parts (Traffic Monitoring and Malicious Activity Detector).

4. FUTURE WORK

Botnets is a center of inclination for both the attackers and the researchers. This concept evolved two decades ago and proved to be a blitz for internet fraternity in this short period. There seems to be a state of war going on between the botnet attackers and defenders or researchers. The researchers are implementing more advanced and organized strategies to detriment the internet users and researchers are consistently trying to cope with their advances. Being an emergent field there is an open room for research and future work.

Deep analysis of different classifications can lead to one generalized model of botnets. Furthermore, every technique mentioned has false positives and negatives which can be improved. The most recent issue which has called for the consideration of researchers is that now the botnet headers try to track honey pots by injecting the binary into the network and examine who is spying their activities; thus banning the hackers when they find them out.

All this discussion reveals that botnets are still in evolutionary phase and provide a capacious field for research.

5. CONCLUSION

Botnet detection is a challenging problem. In this report we proposed a new P2P botnet detection framework. This proposed framework is based on our definition of botnets. We define a botnet as a group of bots that will perform similar communication and malicious activities pattern within the same botnet. In our proposed detection framework, we monitor the group of hosts that show similar communication pattern in one stage and also performing malicious activities in another step, and finding common hosts on them. The point that distinguishes our proposed detection framework from many other similar works is that there is no need for prior knowledge of botnets such as botnet signature. In addition, we plan to further improve the efficiency of our proposed detection framework with adding unique detection method in centralized part and make it as one general system for detection of botnet and try to implement it in near future. It is impossible to defy the significance of botnets in the current circumstances. The ravage they have caused to the finances and solidarity of several government and private organizations has devoted attention of the IT specialists to find the remedy. To be well prepared for future botnet attacks, we should study advanced botnet attack techniques that could be developed by botmasters in the near future..

In this paper we discussed briefly the emergence of botnets, their organization and architecture and botnet life cycle steps. Next the reputed botnet types; the architecture they use and their different possible detection techniques are presented. Although different in architectures, all types of botnets are of great threat to the internet community. They can be used both for good and bad botnet world, giving a concise but complete view of different flavors of botnets. This report gives you a roller coaster ride of the International Conference on Emerging Security Information, Systems and Technologies.

6. ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful comments for improving this paper.

7. REFERENCES

[1] Hossein Rouhani Zeidanloo, Azizah Bt Abdul Manaf, Rabiah Bt Ahmad, Mazdak Zamani, Saman Shojae Chaeikar, "*A Proposed Framework for P2P Botnet Detection*" IACSIT International Journal of Engineering and Technology, Vol.2, No.2, April 2010.

[2] Fatima Naseem, Mariam shafqat, Umbreen Sabir, Asim Shahzad, "*A Survey of Botnet Technology and Detection*" International Journal of Video & Image Processing and Network Security IJVIPNS-IJENS Vol: 10 No: 01.Fröhlich, B. and Plate, J. 2000. The cubic mouse: a new device for three-dimensional input. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems

[3] Hailong Wang, Zhengu Gong, "*Collaboration-based Botnet Detection Architecture",* 2009 Second International Conference on Intelligent Computation Technology and Automation.

[4] Maryam Feily, Alireza Shahrestani, Sureswaran Ramadass, "*A Survey of Botnet and Botnet Detection*" 2009 Third International Conference on Emerging Security Information, Systems and Technologies.

[5] Alireza Shahrestani, Maryam Feily, Rodina Ahmad, Sureswaran Ramadass, "*architecture for applying data mining and visualization on network flow for botnet traffic detection",* 2009 International Conference on Computer Technology and Development.

[6] Hossein Rouhani Zeidanloo, Azizah Bt Manaf, Payam Vahdani, Farzaneh Tabatabaei, Mazdak Zamani, "*Botnet Detection Based on Traffic Monitoring*", 201O International Conference on Networking and Information Technology.Y.T. Yu, M.F. Lau, "A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions", Journal of Systems and Software, 2005, in press.

[7] Jae-Seo Lee, HyunCheol Jeong, Jun-Hyung Park, Minsoo Kim, Bong-Nam Noh, "*The Activity Analysis of Malicious HTTPbased Botnets using Degree of Periodic Repeatability*", IEEE International Conference on Security Technology, 2008.

[8] Wen-Hwa Liao, Chia-Ching Chang, "Peer to Peer Botnet Detection Using Data Mining Scheme", IEEE 2010=