Botnets: The Anatomy of a Case

Sanjay Goel
School of Business
University at Albany,
1400 Washington Avenue, Albany, NY 12222

Adnan Baykal and Damira Pon
Center for Information Forensics and Assurance
University at Albany, Albany, NY 12222

Abstract

Botnets have become the dominant mechanism for launching distributed denial-of-service attacks on computer networks. In a recent incident, the computer network of an organization was attacked and disabled. This attack was initially identified by intrusion detection devices and verified by an on-site review of activity, audit of the log files, and subsequent detailed forensic analysis of the data, which revealed a botnet. The botnet was initiated via a worm infection consequent to which the infected machines attempted to join a bot network. The case presents a forensics analysis of the incident and provides the anatomy of the worm that was used to perform the attack. It also presents detection techniques for identifying botnets and disabling them in order to protect the network infrastructure.

1.0 Introduction

Hackers are constantly devising innovative schemes to exploit weaknesses in computer networks more effectively. A recent method that hackers use to launch distributed attacks on the Internet is through creation of networks of controlled computers. Computers that are controlled through installation of software to use their computing power for a specific purpose are also known as "bots". Bots are defined as "small scripts designed to perform automated functions"[1]. Bots can be useful such as agents whose uses include web indexing or spidering[2], collecting online product pricing[3], or performing duties such as chatting[4]. More negative connotations for bots are "remote access Trojan horses"[5] and zombie/slave computers which refers to those created for less favorable purposes. These bots are not always solitary entities, but can also exist as part of large networks described as "botnets". Behaviorally, botnets have been compared to hive colonies where a queen is analogous to a single point of contact which maintains full command over a host of workers or in this case, bots[6].

The computing power provided by the support of thousands of bots within a botnet make them prime tools for use in activities such as widespread delivery of SPAM email[7], click-fraud in pay-per-click advertising[8], installation of spyware, spread of viruses and worms, as well as distributed denial of service (DDoS) attacks. These networks are composed of machines that have been taken over surreptitiously by hackers through dissemination of worms or Trojans to those machines. According to Alfred Hugar, senior director of Symantec's Security response team, an average of "800,000 to 900,000 PCs at any given time are zombies infected with some type of bot". In addition, according to a Symantec report, the average number of bots grew 15 fold in the first half of 2004[9]. The most potentially damaging use of the botnet is to launch distributed denial-of-service attacks. The hacker

instructs all the machines in the botnet to launch an attack against a specific server and continues until the server crashes or is unable to accept any more connections. Such attacks are more potent since firewalls are unable detect rogue machines successfully when there are thousands of machines sending messages infrequently rather then a few machines sending messages very frequently. In case of a Denial of Service (DoS) attack from a single node, the resources of the attacking machine often limit the scale of the attack and if the server being attacked has sufficient resources, the DoS attack will be unsuccessful or only partially successful.

The first reported large-scale DDoS attack was in August 1999 at the University of Minnesota and involved the launch of more than 200 zombie computers, more than half of which were part of a high-speed Internet connection and resulted in the network being shut down for more than 2 days[10]. A high-profile DDoS attack was made upon Yahoo in February 2000. Around the same time, Buy.com, eBay, CNN.com, Amazon.com, ZDNET, E*Trade, and Excite were also attacked using DDoS attacks. All of these attacks involved the use of a colony of zombie computers that spread within these organizations as well as on other home and university computers. Not only did these attacks affect individual organizations but also the performance of the Internet itself[11]. After their initial appearance in 1999, botnets have progressively gained popularity among the hackers and have become a dominant tool for launching DDoS attacks and SPAM distribution.

According to the CSI/FBI Security Survey, within the first eight months of this year itself DDoS has already incurred losses of greater than 26 million dollars, second only to the amount lost from virus spread[12]. Gangs based in Russia and Eastern Europe have used botnets to blackmail companies with the threat of a potentially crippling DDoS attack[13]. Bots can also be created and herded into botnets to act as hired "mercenaries"[14]. The average botnet fees runs from a couple of cents to a dollar per machine[15]. Earlier this year (2004), the FBI unraveled the first known case where Internet Relay Chat, or IRC-operated DDoS attacks were used to gain competitive advantage in business. The CEO of an organization, motivated by the amount of financial damage that would be sustained by his competitors[16] orchestrated a DDoS attack on the network of a rival organization through the rental of a pre-existing bot network.

The bots can be categorized based on several criteria including their mode of operation and architecture. Based on the mode of operation, they can be organized as using Peer-to-Peer (P2P) networks or IRC networks[17]. However, most bots use "IRC networks and network shares to propagate". IRC bots can be further classified by their composition: 1) single binaries 2) a combination of binaries and source script files, and 3) those that are backdoors of other programs[18]. IRC botnets have both legitimate uses such as supporting IRC channel administrative operations and illegitimate uses such as for distributed attacks[19] and distribution of SPAM. According to the CERT Coordination Center[20], IRC is increasingly being used as the "communications backbone for DDoS networks".

This article discusses IRC channel-based attacks in general and presents the botnet case that occurred via a worm infection. The rest of the article is organized as follows: Section two provides a general description of IRC based bot networks and section three details the forensics analysis of the actual case. Section four presents that anatomy of the bot that was used in the attack and section five provide methods of detection and prevention against botnet attacks followed by some concluding remarks.

2.0 IRC-Based Bot Networks

IRC is a text-based open protocol developed for users to "teleconference". "Channels" are analogous to chat rooms and a select few users, called "operators" are given control over a channel's functions. A channel is denoted by a hash symbol followed by the name of a channel, i.e. #channel_name. While an individual IRC server node operates at a client-server level, the backbone of IRC and the file sharing capabilities of all clients demonstrates a peer-to-peer distributed architecture. Users are authenticated to an IRC server by inputting a nickname, user name, and password. Channels can be moved from one server to another and a specific channel's name can be modified easily[21]. IRC networks allow for flexibility and ease in controlling potentially thousands of bots in an almost anonymous manner since the protocol permits for concealment of identity[22].

The components of a botnet include victim machines, otherwise known as bots; an attacker or group of attackers, who configure and command the bots; a control channel, which is a channel in IRC where created bots join to wait for commands; and an IRC server, which is a server that provides IRC services. The process of creating and using IRC botnets involves first creating a bot, infecting multiple nodes, adding tools for tasks (i.e. adding backdoors, DoS, and scanning), and then loading of control code[23].

Bot software can be downloaded from online warez sites, or from file-sharing communities. Tailoring of the bot is done through manual configuration of the code. Infection usually occurs through "booby-trapped" files, email attachments or web pages"[24]. However, trends in bot technology indicate a blending of Trojan-horse, backdoor, and worm functionality. In fact, many bots are propagated through the spread of bot-worms, which take advantage of exploitable machines.

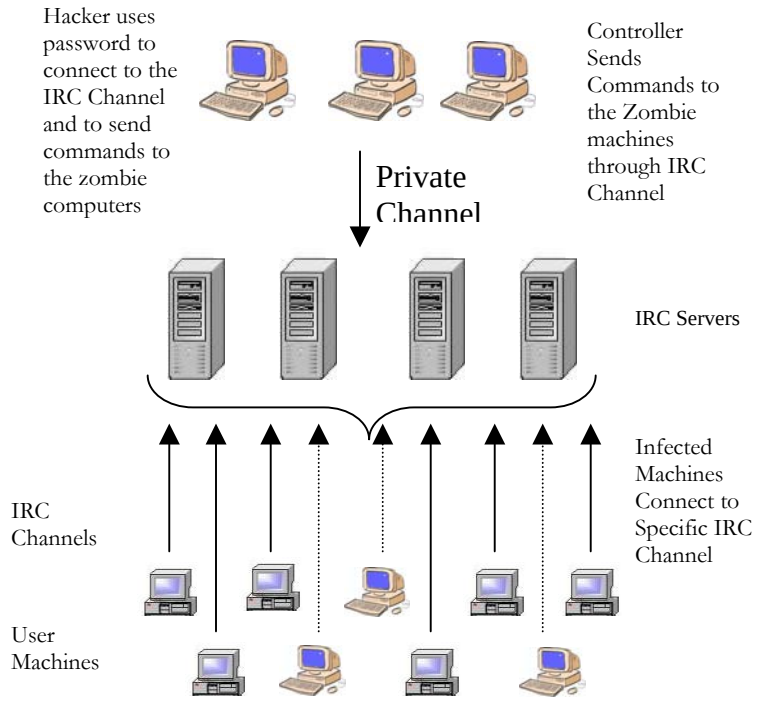

Figure 1: Shows a schematic architecture of a bot network

Figure 1 shows a schematic of the operation of a botnet. Once installed, a bot will attempt connect to an IRC server through a designated port. The default port is TCP 6667, however, an IRC server can be configured to listen to any port. The bot will use a unique generated nickname and a pre-selected password to join a private IRC channel set up by the attacker. An encrypted password is used to prevent other people from controlling and hijacking the botnet. The herded bots become a latent source of resources that are controlled within the channel and can be used by hackers to launch exploits. Once a bot receives the instructions from the control channel, it operates

in a loop and repeatedly connects to the IRC channel, executes the command and disconnects from the network.

3.0 Case Study

This case study discusses the worm infection experienced by an organization that led to the identification of an external bot network. . This incident was characterized by a rapid escalation of machines infected. At 9:00 AM, initial indications of the attack were observed. By the end of the day, this new worm infected approximately 7% of the entire network. As part of the eradication methodology, users were asked to stay off their computers and the Internet. As a result of the worm infection, Windows 2000 machines were observed to "freeze" while a few actually experienced a "blue screen of death". A plot of the infection rate of machines over time as observed by intrusion detection sensors is given in Figure 2.

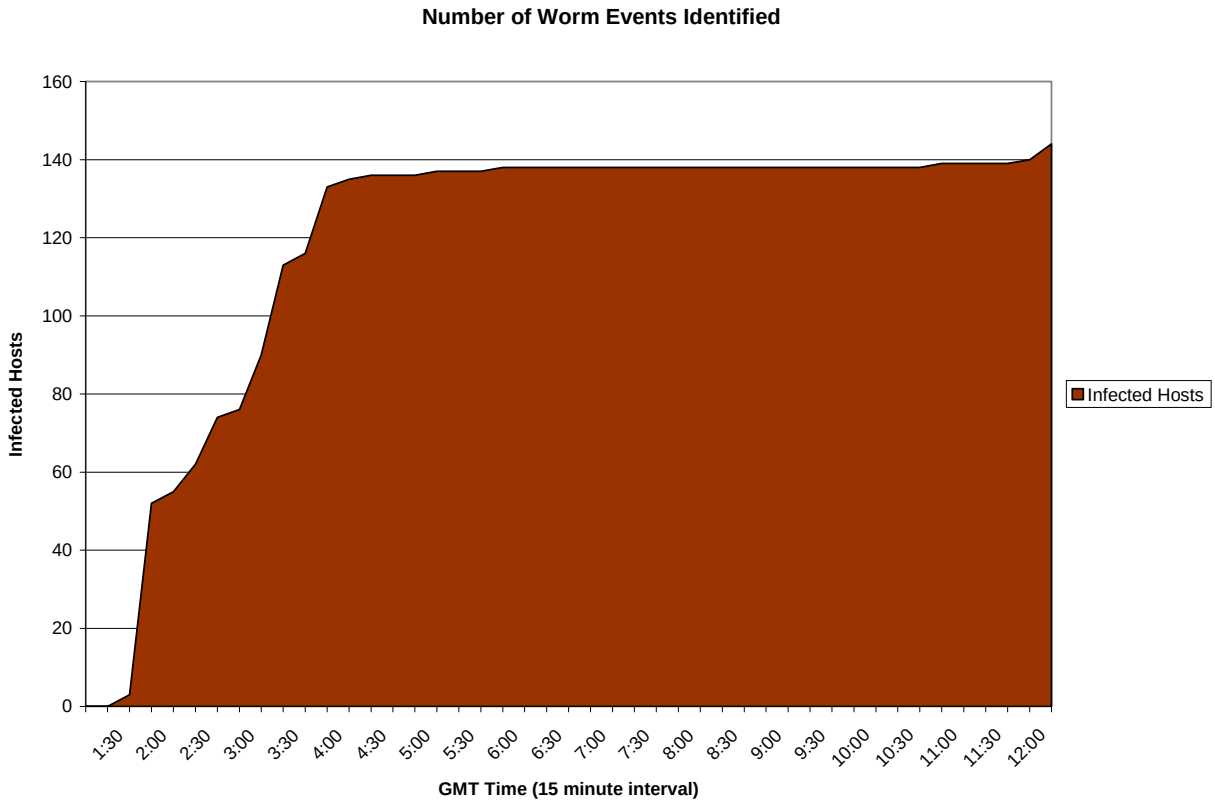

Figure 2: Shows the progression of the worm infection

Intrusion detection sensors initially identified the traffic pattern as scanning for port 445 vulnerabilities from an internal host, implying a possible Sasser, Gaobot, Welchia, or Korgo worm infection. The initial notice identified one infected host followed by three infected hosts within the first fifteen minutes. The rate of infections increased significantly to the point where the plateau actually represents that point in time when a conscious decision was made to begin dropping this logging information to maintain integrity and viability of the intrusion detection monitoring.

Although out-of-band virus definitions provided by the anti-virus software vendor were deployed, initially there were no observable changes within the system. Subsequent analysis indicated the

infected machines were experiencing repeat infections even after the virus was eradicated from the system and the signature files were updated. Although this organization maintained a rigorous patch management program, the infected machines were all missing the same patch, MS04-011. This particular patch required the system to be rebooted once the patch was applied and for these machines, the reboot never took place. Once the patch was reapplied and the worm eradicated, the reinfections ceased.

Logs from various devices across the network such as intrusion detection systems and firewalls were scrutinized to track changes that might provide clues for identifying the source of the attacks. An analysis of the netstat output of the infected machines showed attempted connections to a specific IP address. A resolution of the IP address identified the machine to be a desktop computer of a student at a private university. This machine was found to be a controller for a botnet. This machine was confiscated to perform forensics analysis on the data and logs of the machine.

In addition, this output was analyzed to detect other entities that might have been connected to this controller. An abstraction of the netstat file is presented in Figure 2. The file had approximately 7,000 unique entries, which consisted of either domain names or IP addresses. Among the domain names, none were .gov or .us sites. However, many .edu and broadband sites were present. The IP addresses were also resolved for presence of commercial and government addresses, but none were detected.

Active Connections

Proto	Local Address	Foreign Address	State
TCP	s0309353:1036	160.46.nnn.nnn:Microsoft-ds	SYN_SENT
TCP	s0309353:1036	160.204.nnn.nnn:Microsoft-ds	SYN_SENT
TCP	s0309353:1036	4.36.nnn.n:5608	ESTABLISHED
TCP	s0309353:1036	12.109.nnn.nn:30043	ESTABLISHED
TCP	s0309353:1036	12.144.nnn.n:6533	ESTABLISHED
TCP	s0309353:1036	12.144.nnn.n:16674	ESTABLISHED
TCP	s0309353:1036	cnq25-71.xxx.xxx.xx:1602	ESTABLISHED
TCP	s0309353:1036	host-24-225-nnn-nn.xxxxxxxx.xxx:4578	ESTABLISHED
TCP	s0309353:1036	62.117.nnn.nn:3783	ESTABLISHED
TCP	s0309353:1036	mtowern.xxxxxxx.xxx.xx:2518	TIME_WAIT

Figure 2. Sample Netstat Output

The analysis of the infection showed that a laptop that was infected with a worm, and was connected to the network. The infected laptop scanned the subnet to find candidate hosts to infect and propagate. The computers did not have current patches installed and the worm was able to exploit several machines in a very short period. As the number of exploited machines increased, network traffic inside the subnet simultaneously escalated. This traffic decreased network performance. Additionally, every exploited machine attempted to connect to an external machine. According to the anti-virus software vendor, the worm was programmed to connect to a very specific domain name that via dynamic hosting, pointed to a compromised desktop at a private university. This anomalous traffic was detected by intrusion detection sensors and resulted in the generation of event notifications.

Forensics analysis of a few of the infected machines revealed the IP address of the controller. Through examination of timestamps, a list was generated of suspicious files to be audited. Several tools were used to analyze the infected computers including *regmon*, *filemon*, and *tcpmon*[25]. One of the files showed a potential exploit of the LSASS buffer overflow vulnerability. This file was examined by security engineers at the anti-virus software vendor who generated a signature for the worm and confirmed it as a new variant of the Gaobot Worm. According to a Symantec Internet Security Threat Report, Gaobot was the "second most common attack over the first six months of 2004". During this same time, variants of the Gaobot family "accounted for 67,000 submissions received by Symantec". The next section describes the Gaobot worm and its mode of operation through analysis of source code.

4.0 Anatomy of a Botnet: The Gaobot Worm

The W32.Gaobot variant responsible for the attack has other aliases including Phatbot and Agobot. Due to availability of the code, Agobot 3.0.2.1[26] was analyzed instead of the specific Gaobot variant used in the attack. However, this analysis is closely representative of the function and operation of Gaobot. Agobot is a Trojan creation kit, which enables the development of a custom Trojan based on the specific requirements of a user. The software is written in a modular format, which allows for new exploits and/or scanners to be added. Mechanisms for worm propagation, creation of a network of infected machines, execution of commands, and defense against anti-virus scanners are included. The different aspects of the worm are discussed below.

4.1 Propagation

Agobot usually infects a network through an infected mobile device such as a laptop and spreads to other exploitable nodes in the network. After infecting a node, it starts an FTP server on the node for infected machines to download additional files including the Trojan binary. Once all the infected nodes in the subnet have been identified, the file download process begins. The default configuration file that comes with the source code of Agobot only has two servers: irc.ircd.com and irc2.ircd.com. By default, it uses the TCP port 6667 on these servers to establish a connection. All the configuration variables, along with available IRC servers and channels to which an infected node connects, can be modified from the IRC channel.

It was determined that Agobot propagates through attempted exploitation of several vulnerabilities that are listed below:

1. Weak/null password-protected administrative shares; the worm tries to spread through default administrative shares, namely: e$, d$, c, print$, c$, admin$.
2. LSASS (Local Security Authority Subsystem Service) buffer overflow vulnerability to remotely execute malicious code. More specifically, it creates a script in the exploited machine that instructs the machine to connect to an infected machine, as well as download and execute a copy of the malware using FTP on specified port. The LSASS exploit makes TCP 135, 139 and 445 vulnerable.
3. WebDAV bug in Internet Information Services (would only work on systems running an unpatched version of IIS).
4. RPC/DCOM bugs.
5. Backdoor ports that are opened by the Beagle and Mydoom families of worms.

6. Vulnerabilities in the Microsoft SQL Server 2000 or MSDE 2000 audit (described in Microsoft Security Bulletin MS02-061), using UDP port 1434.
7. The UPnP vulnerability (described in Microsoft Security Bulletin MS01-059).
8. The Locator service vulnerability (described in Microsoft Security Bulletin MS03-001) using TCP port 445. The worm specifically targets Windows 2000 machines using this exploit. The Microsoft Messenger Service Buffer Overrun Vulnerability (described in Microsoft Security Bulletin MS03-043).
9. The Workstation service buffer overrun vulnerability (described in Microsoft Security Bulletin MS03-049) using TCP port 445. Windows XP users are protected against this vulnerability if Microsoft Security Bulletin MS03-043 has been applied. Windows 2000 users must apply MS03-049.

Agobot uses a variety of scanners to exploit other vulnerable systems. In order to scan and spread, a scan command must be issued with specific options. Ranges can be specified for when to start and stop thread execution in scanning the network. Each scanner is run as a separate thread in order to do parallel scanning of the net range. This enables the trojan to scan and spread very quickly, but this also consumes many CPU cycles on the host machine. If configured improperly, the worm may cause a denial of service on the machine it is trying to infect and discontinue its' propagation. The bot should be intelligently configured to use only idle cycles of the host to prevent DOS attack on its host nodes. The Agobot version being evaluated has LSASS, optix, NetBios, dcom, sasser, UPNP, DW, WKS, SQL, WebDav and RAdmin scanners upon compilation. However, it is very easy to add new exploits to the scanner as they become available. Agobot uses FTP to transfer trojan binaries to exploited machines. The IP-addresses of the exploited machines are received from the scanner and the trojan binary is copied onto the new hosts. The windows shares: "admin$", "c$", "print$", "c", "d$", "e$" are searched and list of usernames and passwords are used to attempt exploitation of the machines.

4.2 Networking & Command Execution

Any bot that is infected by Agobot connects to a specific IRC channel on one of the IRC servers specified in the bot configuration file. When the bot initializes, it tries to connect to the first server available on the configuration file list. If the specified servers are unavailable, it sleeps for 30 seconds and tries the list again. The connection attempts continue until a successful connection is established. Additionally, Agobot can also control an IRC server, which enables the creation of a hierarchical botnet. Such an implementation reduces the number of connections made to the IRC channel and divides the bot network into subnet categories. For every subnet, the infected machines connect to one master controller. If the server for the specific botnet is not available, bots attempt to connect to another server contained in the server vector. This server vector is copied at the time of infection and is updated manually by the controller. There is a minimum and maximum number of servers to promote. If the minimum threshold value is reached, a randomly chosen infected machine is marked as server and its IP address is added to the server vector.
Once a connection is established, a bot enters into an infinite loop waiting for commands. The commands are plain text messages with a specific format. Each command consists of several dot-separated (".") commands. One of the dot separated command that a bot accepts is ".bot.uptime" which enables a controller to determine which bots have a very long uptime. It is our belief that the this command is meant to determine the most suitable candidates of the infected machines for promotion to IRC server status, since this would indicate non-frequent rebooting. The command

parser in the code tokenizes the line and looks for the identifier in order to classify it. Once the command is recognized and assigned a category, the command line is passed onto a command handler routine based on category. There are seven command categories:

1. Bot commands: Used to control the each bot.
2. Command Manager: Displays list of commands available
3. Cvar Commands: Used to modify configuration variables.
4. IRC Commands: Used to control all the bots connected to channel.
5. Mac Commands: Allows one to login/logout to bot.
6. Redirect Commands: Used to do various redirections.
7. Download Manager: Various FTP/HTTP download and execute routine.

The command handler routine parses commands and executes the appropriate code. The controller is able to do various tasks from an IRC channel such as the following:

1. Measure the bandwidth by posting messages to pre-specified servers.
2. Get available disk space on a host.
3. Check to see if AOL can be used for spamming.
4. Perform OS Fingerprinting of the host.
5. Log keystrokes of the user.
6. Obtain AOL Instant Messenger passwords.
7. Retrieve CD Keys from registries.
8. Acquire list of e-mails.
9. Procure MSN contact list.

For example, if one wants to perform a denial of service attack (DoS) on cifa.research.org, it is sufficient to execute following command: ".bot.dns cifa.research.org". After receiving this command, all the bots in the IRC channel would start number of threads to perform DoS on cifa.research.org.

4.3 Defense Mechanisms

Agobot has a pre-programmed defense mechanism that enables it to kill at least 610 anti-virus programs (this list can be modified and updated). It scans the list of processes running on the system every 20 seconds and kills those that match one of the listed programs. Additionally, Agobot contains commands to do process and service control. A controller can issue a command to view what processes are currently running on a system and kill a process suspected to be anti-virus not already on the list. A controller is also able to start/stop services on a windows machine using service control commands.

Like most IRC bots, Agobot uses password protection to limit the number of users that can control the botnet from an IRC channel. To control the bot, one must issue the command ".login<username><password>. If username and password matches the one that is contained in the trojan binary, the bots will print a message "password accepted" otherwise, the bot will quit the IRC channel. After a successful login, there are a maximum number of milliseconds a bot is available to accept commands from the controller. If this threshold is attained and no command is received, the bot logs the user out.

5.0 Detection of Botnets and Protection of Networks

According to Jim Jones of US-CERT, there are three stages in the process of dealing with botnets: 1) prevention, 2) detection, and 3) response. Some recommendations for prevention of bot infection are to disable unnecessary services and ports; take care of known vulnerabilities; ensure systems are well patched, updated, and tested regularly; use effectively configured firewalls; enforce complex and up-to-date passwords; and generate awareness among users. Large organizations are advised to install ingress and egress filters to stop Internet packets with spoofed IP return addresses from entering and exiting the network. However, purchase and installation is costly[27] and too much filtering can overwhelm routers[28]. In addition, adoption of IPSec (IP Security Protocol) and DNSSec (Domain Name System Security Protocol) would assist in identifying spoofed IP addresses within packets. However, this solution will take time to implement globally.

IRC botnets may go undetected for a long period. Various reasons for this include dissimilar pattern or signature of infection for differing incidents, firewalls may not bring attention to traffic if compromise has taken place on the client-side, and data packets sent before a command is issued to a bot are minimal[29]. There are various methods for IRC bot detection. Individual machines should be scanned for the existence of malware and logs generated from security devices should be constantly monitored for anomalies. In addition, high volumes of traffic or anomalous traffic may indicate the presence of a botnet. Packer sniffers can be used to detect and then isolate infected areas of a network. In addition, logs from a network sniffer can be used to find out IRC server used, the name of the private channel used by the botnet owner, and authentication key if communication is unencrypted. Once botnets are detected the channels that they connect to and the shell account that generates the anomalous traffic is usually closed. If a botnet is created on a fixed IRC server name or IP address, these methods are effective. However, many of botnets migrate to dynamic hosts, which make it easier for them to connect to other servers and more difficult for administrators to disable them. In such cases, if the dynamic address that the bots are using is identified the address can be null routed to prevent the botnet from migrating to other servers.

Hanna recommends the use of Snort to poll for IRC traffic for "a particular exploitation signature". However, he concedes that the payload for bots can change easily. In addition, rules must be created carefully to prevent false positives, which would restrict legitimate IRC traffic. In addition, a search for many outgoing connections would falsely implicate file-sharing traffic prevalent on university campuses. Honeypots and honeynets (networks of honeypots) have also been used for detection of botnets. Honeypots are relatively new technology and are used as bait for hackers and online attacks. The main purpose of honeypots is data collection and observation. In a study done by McCarty, a honeypot became part of a botnet that was used in a DDoS attack. The tools to infect machines into bots are adapting to the proliferation and detect whether vmware or other similar software is running and will not join because of the likelihood that they are honeypots. The IRC server IP addresses, IRC channel names, and other information necessary to access the botnet were contained within collected data. Within IRC itself, the challenge of bot detection is more difficult. Bollinger and Kaufmann[30] attempt to create algorithms for bot detection through analysis of IRC traffic. However, IRC relay daemons can be installed that mask the IP address of exploited computers when connecting to an IRC network[31].

Response to a botnet DDoS attack can be isolation of the subnet on which there is an infection and collection of data from all systems and defensive devices for forensic analysis. Research being done

in this area includes use of active networks to perform automated intrusion response[32] and a distributed approach to DDoS detection and response[33]. A feasible technique for detection is to perform forensics analysis on network traffic to analyze the botnet behavior. Figure 3 shows a simple process in which the network traffic is sniffed and classified based on different criteria. Figure 4 shows the number of DNS resolves for each node on the network and a few nodes with abnormal behavior are identified. The data on the nodes with abnormal behavior is further analyzed to identify the infected machines. High traffic on non-standard ports (ports with a number higher than 1023) leads to suspicion of bot traffic. Hackers are adapting to this detection technique and are using lower ports (including port 80) for bot communication making this mode of detection more difficult. Physically scanning of known hacked machines can then assist in identifying other infected machines. Simple forensics analysis coupled with traditional tools for analyzing the node for security can thus be used to effectively identify the bots.

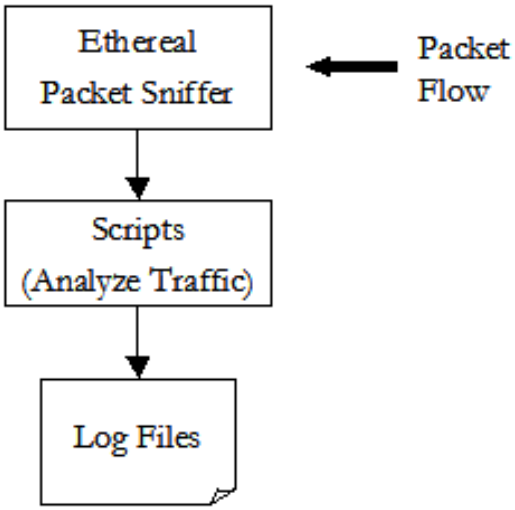

Figure 3: Forensics Analysis Process

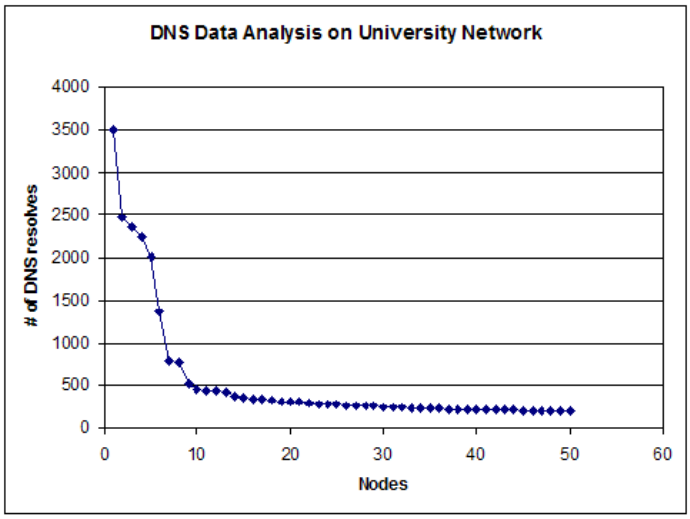

Figure 4: Data from Forensics Analysis

6.0 Case Analysis

Since the detection of the botnet discussed here, a large number of botnets have been discovered in various networks across government agencies. Especially vulnerable are higher education institutions where the network and computing environment are more loosely controlled. In some statistics, it is estimated that more than a third of university machines are infected by botnets. It is evident from the unfettered growth of botnets that current techniques for network defense (e.g. firewalls, intrusion detection systems, auditing, and monitoring) are not providing effective protection against the spread of botnets. It is relatively easier to detect botnets and zombie machines than to prevent hackers from infecting the machines and creating zombies.

A single vulnerable machine can provide a beachhead for a botnet to propagate across a network. It is difficult to harden the defenses enough to create an impenetrable system without sacrificing functionality. In addition, it is infeasible to ensure absolute compliance with security policies since irrational entities are involved in policy adoption and use. So far, the response to the rise of botnets has been ad hoc without cogent policies in place that are defensible based on financial rationalization. Thus rigorous risk analysis is required that determines the impact of botnets on the organization. Based on a risk analysis, rational policies that mitigate (or eliminate) the risk need to be

enacted and enforced. These policies should include several different elements, including, hardening of defense, network & computer forensics, user education, auditing, and enforcement. For instance, if a critical machine is suspected to be a bot client, should it be immediately quarantined to prevent further infection on the network or should it be allowed to operate until the machine has been thoroughly analyzed?

Another factor that merits further investigation is the impact of freeware and shareware on organizational security. It is very tempting to download free software, however, it is essential that we quantify the hidden costs associated with this activity. A lot of the popular freeware and shareware often comes bundled with trojan horses that then help in virus propagation. Some of this software is attractive to download and use on machines due to their functionality and cheapness. However, often users do not realize that they could be inadvertently downloading a trojan horse that will leave a back door in their machine allowing it to be controlled by a hacker. The two fastest growing applications on the network, those for instant messaging and peer-to-peer systems are vulnerabilities through which major security threats can manifest themselves. According to Grabowski[34], there are several disadvantages of using free programs, in which security is one of the most critical. He also states that instant messaging has several associated threats, such as, viruses & worms, trojan horses, hijacking, and denial of services. Similarly, threats from peer-to-peer systems exist in terms of spyware, worms, and trojans that come bundled peer-to-peer system software. Given the reach of the peer-to-peer systems, trojan horses can propagate rapidly across the network creating an explosion of zombie machines.

7.0 Conclusions:

Use of botnets has led to a rise in distributed denial-of-service (DDOS)[35] attacks on resulting in significant losses to the economy. These networks can be harnessed for constructive purposes; however, currently they are primarily being used to perpetrate computer-related crimes, such as, SPAM and DDoS. Response to a botnet attack is difficult. However, if proper precautions are undertaken prevention is possible. All organizations (government agencies, private organizations, and academic institutions) are vulnerable to botnet attacks. This situation is very dangerous considering statistics from Symantec, which indicate that about a million bot computers are present at any given moment. In addition, knowledge of the financial impact of a botnet attack should also encourage awareness of botnets when formulating security policy, as well as implementing procedures, and controls. Although bots and botnets have been present for the last five years at least, they are still a little understood threat, which needs to be seriously considered.

Acknowledgements:
This work is done with partial support of NSF 01-67 Grant 020657151 and FIPSE Grant P116B020477. The authors would like to acknowledge the support of Justin Azoff from the University at Albany for providing useful suggestions on botnet detection. The authors would like to thank the Center for Information Forensics and Assurance for supporting this project.

References:

[1] Munro, Jay. (2004, December 14). Bots March In: These worms could "zombify" your computer, but you can give bots the boot. PC Magazine, p. 90.
[2] Rosenfeld, J.M. (2002). Spiders and Crawlers and bots, Oh My: The Economic Efficiency and Public Policy of Online Contracts that Restrict Data Collection. Standford Technology Law Review, 1-31.

[3] Crane, E. (1999). Attention Shoppers! Shopping bots promise to gather the best bargains on the Web—but do they really work? We sent out dozens of automated shopping assistants. Find out which ones brought home the Bacon. *PC World.*

[4] Auslander, S. (2002). LIVE FROM CYBERSPACE or, I was sitting at my computer this guy appeared he thought I was a bot. Performing Arts Journal, 70, 16-21.

[5] McLaughlin, L. (2004). Bot Software Spreads, Causes New Worries. *IEEE Distributed Systems Online, 5*(6), 1-5.

[6] Elliott, J. (2000). Distributed Denial of Service Attacks and the Zombie Ant Effect. IT Pro, 55-57.

[7] Bruno, L. (2003). Baffling the Bots. *Scientific American*, 1-2.

[8] Olson, S. (2004). Lawsuit filed by clicked-off company. *Canberra Times*, p. A18.

[9] Turner, D., ed. (2004). Symantec Internet Security Threat Report Trends for January 1, 2004 – June 30, 2004. *Symantec, VI*, 1-55.

[10] Garber, L. (2000). Denial-of-Service Attacks Rip the Internet. Computer, 12-17.

[11] Garber, L. (2000). Denial-of-Service Attacks Rip the Internet. Computer, 12-17.

[12] Gordon, L.A., Loeb, M.P., Lucyshyn, W. & Richardson, R. (2004). *2004 CSI/FBI Computer Crime and Security Survey*, 1-16.

[13] Cowan, R. (2004, November 13). Hordes of web bots do crooks' bidding. *The Guardian.*

[14] Acohido, B., & Swartz, J. (2004, November 29). Unprotected PCs can be hijacked in minutes. *USA Today*, p. 3B.

[15] Bryan-Low, C. (2004, November 30). Virus for Hire: Growing Number of Hackers Attack Web Sites for Cash. *Wall Street Journal*, p. A1.

[16] Poulsen, K. (2004). FBI busts alleged DDos Mafia. Security Focus IDS News, Retrieved on October 4, 2004 from http://www.securityfocus.com/news/9411.

[17] Prolexic. (2004). Distributed Denial of Service Attacks. *Prolexic Technologies White Paper,* 1-36.

[18] SwatIt. (2003). Bots, Drones, Zombies, Worms and other things that go bump in the night. BOTS, Retrieved November 9, 2004 from http://swatit.org/bots/index.html.

[19] McCarty, B. (2003). Botnets: Big and Bigger. IEEE Security & Privacy, 87-90.

[20] Houle, K.J., Weaver, G.M., Long, N., & Thomas, R. (2001). Trends in Denial of Service Attack Technology. *CERT Coordination Center*, 1-20.

[21] Oikarinen, & Reed, D. (1993).Internet Relay Chat Protocol. 1-62. Retrieved from http://www.irchelp.org/irchelp/rfc/

[22] Puri, R. (2003). Bots & Botnet: An Overview. SANS Institute, 1-16.

[23] Merchant, C. (2002). Detecting and Containing IRC-Controlled Trojans: When Firewalls, AV, and IDS Are Not Enough. *SecurityFocus*, Retrieved on December 5, 2004 from http://www.securityfocus.com/infocus/1605.

[24] Ranum, M. (2004). I, BOTNET. Information Security Magazine, Retrieved from http://infosecuritymag.techtarget.com/ss/0,295796,sid6_iss446_art925,00.html

[25] Sysinternals. (2004). Sysinternals Web Site, Hhttp://www.sysinternals.comH.

[26] Agobot3.0.2.1. (2004). NetworkPunk, Retrieved on November 30, 2004 from Hhttp://www.networkpunk.com/?q=node/view/265&PHPSESSID=a54b731884e346617d58fe701439ad15H

[27] Garber, L. (2000). Denial-of-Service Attacks Rip the Internet. Computer, 12-17.

[28] Geng, X., & Whiston, A.B. (2000). Defeating Distributed Denial of Service Attacks. IEEE IT Professional 2(4), 36-41.

[29] Hanna, C.W. (2004). Using Snort to Detect Rogue IRC Bot Programs. *SANS Institute*, 1-17.

[30] Bollinger, J., & Kaufmann, T. Detecting Bots in Internet Relay Chat Systems. [Thesis], Computer Science, *Institut für Technische Informatik und Kommunikationsnetze.*

[31] Baumann, R., & Plattner, C. (2002). Honeypots. [Dissertation] Computer Science, *Institut für Technische Informatik und Kommunikationsnetze,* 1-143.

[32] Sterne, D. Djahandar, K., Balupari, R., La Cholter, W., Babson, B., Wilson, B., Narasimhan, P. & Purtell, A. (2002). Active Network Based DDoS Defense. *Proceedings of the DARPA Active Networks Conference and Exposition (DANCE '02).*

[33] Papadopoulos,C., Lindell, R., Mehringer, J., Hussain, A., & Govindan, R. (2003). COSSACK: Coordinated Suppression of Simultaneous Attacks. *Proceedings of the DARPA Information Survivability Conference and Eposition (DICEX '03).*

[34] Grabowski, S. (July 2003). The Real Cost of "Free" Programs such as Instant Messaging and Peer-to-Peer File Sharing Applications. SANS Institute.

[35] Lau, F., Rubin, S.H., Smith, M.H., & Trajkovic, L. (2000). Distributed Denial of Service Attacks. *IEEE International Conference on Systems, Man, and Cybernetics*, Nashville, TN, 2275-2280s.