2011

Botnet lab creation with open source tools and usefulness of such a tool for researchers

Dimitris Vergos

Botnet Lab Creation with Open Source Tools and usefulness of such a tool for researchers

January 10, 2011

By

Dimitris Vergos

Thesis submitted in partial fulfillment of the requirements

for the degree of

Master of Science in

Computer Security and Information Assurance

Rochester Institute of Technology

B. Thomas Golisano College

Of

Computing and Information Sciences

Thesis Reproduction Permission Form

Rochester Institute of Technology

B. Thomas Golisano College

Of

Computing and Information Sciences

Master of Science in
Computer Security and Information Assurance

Botnet Lab Creation with Open Source Tools and usefulness of such a tool for researchers

I, Dimitris Vergos, hereby grant permission to the Wallace Library of the Rochester Institute of Technology to reproduce my thesis in whole or in part. Any reproduction must not be for commercial use or profit.

Date: _____ Signature of Author: _____

Rochester Institute of Technology

B. Thomas Golisano College

Of

Computing and Information Sciences

Master of Science in
Computing Security and Information Assurance

Thesis Approval Form

Student Name: Dimitis Vergos

Thesis Title: Botnet Lab Creation with Open Source Tools and usefulness of such a tool for researchers

Thesis Committee

Name	Signature	Date

Charles Border_____
Chair

Bill Stackpole_____
Committee Member

Harris Weisman_____
Committee Member

Acknowledgements

I would like to thank my thesis committee for their participation on my project, and their guidance and assistance that they have provided me with.

I would to thank my parents for their constant support and emotional guidance at times when I could not continue any further.

I would also like to thank Eri who has been the one person that has given me that extra push forward to finish what I started a long time ago, and who understood my bad mood swings during these past difficult months.

Abstract

Botnets are large scale networks, which can span across the internet and comprise of computers, which have been infected by malicious software and are centrally controlled from a remote location. Botnets pose a great security risk and their size has been rising drastically over the past few years. The use of botnets by the underground community as a medium for online crime, bundled with their use for profit has shined the spotlight on them. Numerous researchers have proposed and designed infrastructures and frameworks that identify newly formed botnets and their traffic patterns. In this research, the design of a unified modular open source laboratory is proposed, with the use of virtual machines and open source tools, which can be used in analyzing and dissecting newly found bots in the wild. Furthermore, the usefulness and flexibility of the open source laboratory is evaluated by infecting my test machines with the Zeus Bot.

Table of Contents

1 Introduction

1.1 Botnets

Botnets consist of a large number of bots (infected computers) under the control of a bot "herder", also known as the bot master, who provides instructions/commands on what types of attacks his "herd" should perform.

The bot "herder" controls the botnets from a single location called the Command & Control (C&C) center. The most widely used C&C technologies that have been recorded up to today have been IRC channels, web (HTTP), Peer to Peer (P2P) and DNS. Regardless, of what communication methods are used for contacting the C&C center, as soon as the victim is infected, the bot will attempt to contact its C&C and download updated configurations files, software upgrades, and commands issued by the bot master. The security field is a game of "cat and mouse", bot programmers will keep on figuring out new and more efficient methods for controlling and communicating with their "herd" and it is up to the security experts in finding new methods in stopping them.

Botnets are currently considered a great online threat, and have been identified with causing numerous attacks and major service disruptions all over the web, which in turn causes large corporations and businesses to lose a lot of money. Botnets are available for hire in the underground market for anyone willing to pay the right amount.

According to information that has been gathered by the Shadow Server Foundation [1], botnet activity has almost doubled over the past two years (as seen in figure 1 below).

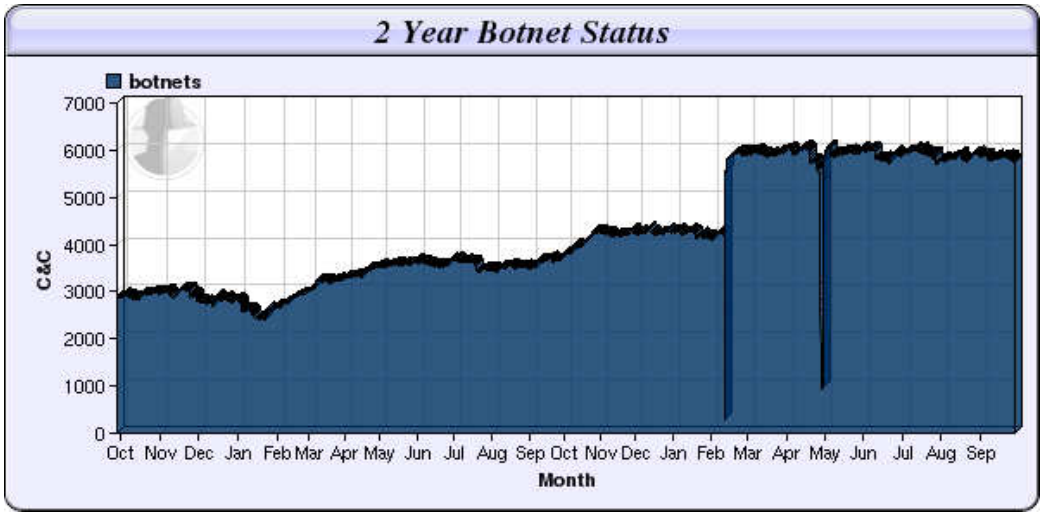


Figure 1. Botnet Activity over the past two years

In order to combat such a dangerous and expanding threat, a large number of scientific research has been performed, and innovative prototype systems / frameworks have been developed in order to give insight on the inner workings of botnets, and assist in the defense against an enemy that dominates today's attack landscape.

Due to the uprising of cheap and fast internet usage, more and more broadband users are constantly connected to the internet, which causes perfect targets for new bots. The unsuspecting security illiterate users partake without their knowledge in attacks, which need to be stopped with innovative ideas which help identify such attacks before they happen.

As mentioned above, botnets are merely the medium used by the underground community for uses that are triggered by money and the need for

destruction. According to the information gathered on botnets by the Honeynet Project [2], a research organization dedicated to improving the security on the internet, botnets are mainly used to conduct the following forms of attacks:

1. **Distributed Denial of Service Attacks (DDoS)**

   A DDos attack is an attack on a computer network to render their resources useless by fully consuming the targets bandwidth or computational resources.

2. **Spamming**

   With the assistance of a large botnet, the bot master is able to send thousands of spam email campaigns, as well as collect thousands of emails from infected workstations, which can also be used to send out spam/advertising emails

3. **Key logging**

   Bots have the ability to log all keystrokes performed on the host that they have infected, thus recording highly confidential information such as bank account information and passwords.

4. **Adware**

   Adware exists to advertise specific commercial entities, in which the user has no involvement in and he is not aware of these actions

5. **Spyware**

Spyware is software, in which information is sent to the infector of a particular system about what operations he has been performing (sites visited, passwords entered, and credit card numbers)

6. **Click Fraud**

   Click fraud, is where the bot directs the infected users to specific web-sites in order to generate traffic and gain personal or financial gain

7. **Fast Flux Attacks**

   A fast flux attack is where the bot hides different malicious sites behind a set of compromised hosts that act as proxies, in which unsuspected users visit

In order to combat this increasing plague in computer security, researchers need to track newly formed botnets, and obtain as much information as possible. The procedure required in analyzing a bot, necessitates an isolated lab environment equipped with specialized security tools, which are used to monitor and record all the activity generated by the infected hosts.

A publically available, unexplored path, for bot analysis is the creation of a unified laboratory, in which a researcher can use a plethora of tools that are available at his fingertips in order to conduct his investigation. Through the research that was performed about botnets, their C&C and their attack patterns, and by studying the methods used by security researchers to analyze the bots behavior, has prompted the design of a framework which, consists of all

the necessary open-source tools that are needed to perform both forensic and network analysis on specific bots.

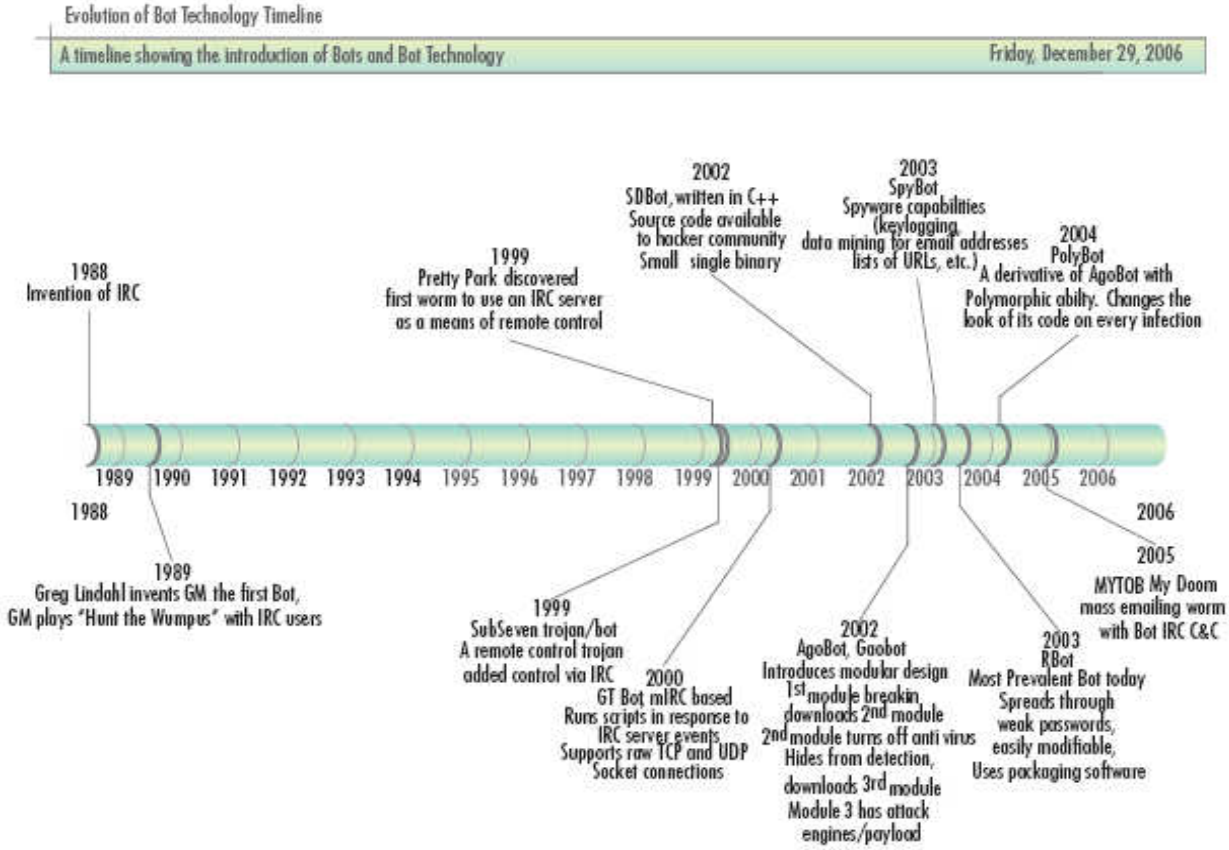
Furthermore, an important goal of this research was to design a centralized console in order to control the deployment of the bots and the management of the guest O.S's which are going to be used for the investigations. The research and design of the botnet laboratory, is based on the information and knowledge provided by a select group of security researchers.

1.2 History

After analyzing the functionalities of botnets, and what actions they are used for, it would be nice to include a few words about their history and where they originated from. As most tools on the World Wide Web, bots did not start as a malicious tool that were used by hackers to cause havoc and destruction, but as a tool that was able to monitor activity on IRC channels and perform actions for the channel operator while he was pre-occupied with other activities. The first IRC bot (GM), would play a game of "Hunt the Wampus" with IRC users.

The baseline is, that all of the original IRC bots that were created, were designed to assist the users in enjoying their IRC experience or in managing IRC connections and performing tedious operations for the IRC operator. Around this time, is when different IRC servers offered OS shell accounts, which allowed the users to run commands on IRC hosts. After 1999 multiple releases of different bots appeared, that would employ different methods to steal personal

information and spread to different workstations. The major change occurred in early 2002 when the SDbot appeared, which was written in C++, and the author uploaded the source code to the internet for anyone to use and alter. Following this revolution in bot creation, we have multiple malware writers creating new bots that would utilize existing vulnerabilities in software in order to propagate and collect information from different infected users. Below you will find the history of botnets [17].



1.3 Objective

The main objective of this thesis is to propose the creation of a unified botnet laboratory that will assist researchers in analyzing bots/malware and monitor their behaviour with open source tools. The need for such a laboratory

rose after the discovery that in all of the research papers that were read, the researchers never mentioned in detail how they set up their test-beds for testing their frameworks.

The main objective of the thesis is:

1. The creation of a laboratory with the use of virtual machines and a combination of open source tools in order to analyze bots that are discovered in the wild.
2. The creation of a management console to manage the infected hosts and the malware that will be analyzed.
3. The provision of transparency no matter what type of bot and command and control system is being used by the bot (currently focusing on HTML and IRC bots).
4. The botnet laboratory utilizes traffic monitoring tools and the darknet concept in combination with other forensic tools to discover bot traffic patterns and activity.
5. The creation of a central repository in which all of the bots will be stored and can be used for analysis.

1.4 Limitations

The proposed system has the following limitations:

1. The laboratory relies on systems like honeywall, mwcollect and nepenthes, in order to get hold of the bots that are going to be analyzed.
2. The laboratory needs to be set up according to each researcher's needs, thus a minimum amount of configuration and tweaking maybe needed according to the architecture that he wishes to deploy. Configuration functionality is not yet provided to the researchers in order to perform these modifications; they need to be done manually.
3. While the laboratory has been set up in order to capture all of the data that the bot transmits automatically, the investigation still needs to be done manually, as well as the reporting of the findings.

1.5 Contributions

This research makes the following contributions:

1. While numerous researchers have proposed in the past frameworks such as honeywall that assists researchers in catching bots in the wild, none of the researchers has proposed (commercially or publically available) a unified lab environment in which analysis of the bot captured can be performed on. The research conducted proposes and utilizes such a laboratory, which can be used by any security

analyst that does not have time to waste in setting up an environment from the beginning.

2. This research also utilizes the advantages of virtualization and open source tools in order to perform a runtime analysis and reverse engineering of the malware without the fear of contaminating the host machines, or the network he may be working on.
3. The proposed system provides a starting point for researchers to expand the functionalities of the system and provide advanced options that can be used by any researcher.

1.6 Chapter Summary

Chapter 2 analyzes the tools that will be used for the creation of the laboratory, and discusses related articles to this research. Some of the tools that will be discussed include snort, virtualization, honeysnap, IDA and Argus.

Chapter 3 details the architecture of the laboratory that has been developed for testing.

Chapter 4 discusses the methodology used for testing the laboratory. It details how specific bots were tested, and how they were analyzed, as well as the information that was gathered from the tests performed.

Chapter 5 summarizes the results as well as the contributions that are made by this research, and emphasis is given on the additional functionalities that could be added in the future to enhance the laboratory.

2 Literature Review

Research about new and innovative ways to discover and combat botnets are being published regularly, since the problem with botnets keeps on getting worse as time goes by. As mentioned above, the security field can be characterized as a game of "cat and mouse", where the malevolent users of the web, will keep on finding new ways of launching undetected attacks using botnets. The research conducted, will assist security experts, by providing a framework that will help them in deploying their own virtual laboratory with minimal hassle and maximum benefit.

In this section we will discuss and analyze the tools and information that were used for the completion of this thesis. These include virtualization tools, snort, ntop, IDA, honeysnap, Argus and general information about botnets. After analyzing these tools and identifying their usefulness, several articles will be analyzed, which are related to this work and have provided the necessary information and insight into the world of botnets that helped with the design of the botnet laboratory. While there has been significant research about the above mentioned tools, no researcher has proposed the combination of these tools, in order to create a unified testing environment that could help all researchers in the field. It is important to explain the individual use of each tool and how they have been used by security researchers in order to comprehend the combination of them under a single "roof". The design and research

presented in this thesis could not have been successful without their research and ideas.

2.1 Botnets

According to the research conducted by Nicholas Ianelli and Aaron Hackworth [3], great insight was gained into the world of botnets.

Their research paper provided a detailed description of how botnets are built, the capabilities that they posses, how they are operated, maintained and defended by the bot master. The article provides examples of such events and helped in getting a clear high-level picture about botnets architecture.

From the information provided in the above research article, general understanding was gained about botnets and their techniques, but it failed to provide information that would assist us in identifying the techniques and technologies that the researchers used in order to capture and present their findings.

This research and framework will assist such researchers in having a common and united framework where they will be able to deploy and do further bot analysis and testing on.

Another insightful research paper is about Botsniffer [4], which provided great information on how botnets operate on a technical level, and the different techniques that can be used to detect botnet commands and control channels in network traffic.

This above research assisted us, when developing the open laboratory framework, in order to set up a correct C&C center using two separate Linux servers for HTML and IRC based C&C.

Li, Z., Goyal, A., Chen, Y., and Paxson, V. investigated and analyzed the data from honeynets in order to create a framework for analyzing large-scale botnet probing events [5], so they could understand the significance of large-scale botnet probes. This research paper was very helpful in understanding the use of honeynets and different tools such as honeysnap, which are used in investigating IRC traffic generated by bots when communicating with their command-and-control.

Lastly, in "An inside Look at Botnets" [6], the researchers analyzed and evaluated different bots (malware) that have been caught in the wild, and they have documented their activity. The researchers fundamentally believe, that the only way to stay ahead of the game is by analyzing and figuring out how the mechanisms employed by malicious software work. The vast details provided on the various bots that were analyzed, provided a database of bots that could be used to test the functionality of the botnet laboratory. Another very important piece of information identified in this paper, was the suggestion of the use of online security laboratories such as WAIL [7] and DETER [8]. The former is mainly used for internet anomaly & intrusion detection, network traffic management, wide area network measurement and network & storage systems, and the later while more security based, focuses on a larger more decentralized group of researchers. The aim of this thesis is to provide the

researchers with a framework that they have total control over, and can add their own features and modifications if needed.

2.2 Virtualization tools

Virtualization provides us with the possibility of running multiple operating systems of different or similar architectures on top of our host machine which, shares its resources with the guests virtual machines. The virtual machines operate independently of its host machine. According to Peter M. Chen's and Brian D. Noble's research "When virtual is Better than Real" [9], they discuss the benefits of running virtual machines to provide specific services that maintain a separation with the host operating system, which in return gives us increased security and portability. According to their paper, with the use of virtual machines, you are able to manipulate their state with great ease, clone them, save them, restore them and dispose of them after their use, simply by deleting them. Based on the ideas by the above researchers, virtual machines can be used to create multiple isolated operating systems by cloning several templates that have been created specifically for our test environment. Additionally, virtual machines provide the ability to create restore points, which we can characterize as the guest machines clean state, before an investigation of a particular bot/malware is conducted. After the conclusion of the investigation, the researchers are able to restore the guest O.S. to their clean state (pre-defined restore point), or simply dispose of the virtual machines, and create new clones for a new bot analysis project.

The use of virtual machines, provide us with the ability to construct an isolated laboratory environment that was discussed in section 1. The use of virtual machines is the Alpha and the Omega of the botnet laboratory design.

2.3 Intrusion Detection

An Intrusion detection system (IDS) is a device (hardware or software), that is designed to monitor the network and identify malicious attacks against your infrastructure. When an attack is detected by the IDS, an alert is generated and the traffic is logged (which can be used for future analysis).

Depending on how IDSs gather their data, and how they check and identify for attacks, they can be categorized as host-based or network-based.

Network based IDSs identify intrusions by listening to network traffic and can monitor multiple hosts simultaneously. Furthermore, network based IDSs capture all traffic that passes through the network (that they are sitting on) and analyzes the content of it for malicious traffic. The traffic that is captured can also be logged as a network capture file (pcap), and can be examined locally by other tools as well (i.e. Honeysnap).

On the other hand, host based IDSs are enabled by installing an agent on the host machine that you wish to monitor, and analyzes system calls, application logs and file system modifications.

Both host-based and network-based IDSs can be either anomaly based or signature based. A signature-based IDS uses a predefined set of signatures of existing attack patterns, while this method is highly effective for existing and

known attacks, it is not able to detect new attacks that it does not have signatures for. Anomaly-based IDS creates a performance baseline on the network traffic, thus each time it detects activity that is outside of the parameters that have been defined, and it will alert us on the detection of hostile activity. The only drawback, is that as your network changes and increases in size a new baseline needs to be created in order to portray your networks current traffic, which if not performed can increase the false positives that you will receive.

In an article on Multi-Sensor Agent-Based Intrusion Detection Systems, Richard A. Wasniowski discusses the different types and uses of IDSs. He also emphasizes on the use of the popular snort IDS, which "looks for attack signatures, which are specific patterns of activity that has been defined to be of a suspicious or malicious intent" [10]. Based on the information gathered by the use of IDS's in identifying intrusion detection, Snort was used to monitor and alert us on botnet network activity, and assist in creating rules in order to defend against them.

2.4 Bot analysis and reverse engineering

According to Min Gyung Kang in his research paper "Botnets as a Vehicle for online crime" [3], he describes how reverse engineering and runtime analysis, are one of the main techniques used for botnet analysis. While both techniques may at particular circumstances reveal the same amount of info about a bot, a runtime analysis may be as simple as running a program to

capture the network traffic generated, while with reverse engineering you are able to dissect the executable and see all the actions that the malware writer indented the bot to perform. When you are done with reverse engineering all of the secrets of the bot, will have been revealed and can be defended against. The negative aspect of reverse engineering is that it is time consuming, and it requires a large amount of expertise. With the insight and information gathered from Min Gyuang Kang's research, as well as information provided by Defence Intelligence on the Mariposa botnet [11], it helped in identifying the necessary reverse engineering tools used by security experts in dissecting a bot.

2.5 Darknets and traffic monitoring tools

2.5.1 Darknets

Darknets have evolved from honeypots, they are really easy to implement and very useful in capturing information when examining a bot. A darknet is a segment of private or public network space that does not contain any production servers or workstations [12]. In the darknet we can have one or more servers called "catchers" which, as their name gives away catches all of our packets and inspects them. As discussed in Greynets: A definition and Evaluation of Sparsely Populated Darknets, internal darknets can provide an early warning system of hosts launching different types of attacks [13], and thus if used effectively you are able to collect the right amount of information that is needed to analyze a botnet. Below in figure two, you are able to see a darknet

setup, in which all blocked/illegal packets are forwarded to our darknet in which our scanners analyze the traffic.

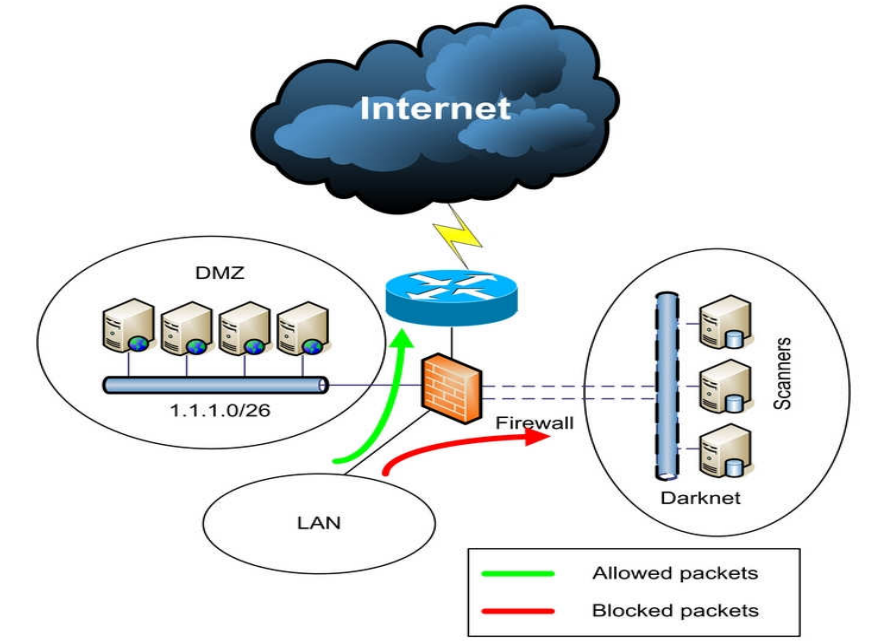


Figure 2. Darknet Architecture

With the above concept in mind, you are able to set up a single malicious network segment, which contains our bots. Thus, all of the traffic that is generated by the hosts will automatically be forwarded to the "catchers" and analyzed to identify botnet behavior.

2.5.2 Traffic monitoring tools

Lastly, when capturing malicious traffic, there is a need to gather statistics about the amount, and type (i.e. UDP/TCP) of data that is transmitted. With the use of a powerful tool called Ntop [14], you are able to gather information about the different packets that are being sent and the ports that are being used on each host. Bundle this information with the detailed network graphs that are generated you are able to identify the amount of traffic that a

malware is generating. A group of researchers, investigating new payload attribution methods for forensic investigations, utilize a modified version of Ntop to identify the source, destination and the times that the payload appeared, thus tracing the spread of the malware [15]. Following the information provided by their research, the botnet laboratory will also contain Ntop to track the botnets spread.

3 Architecture

Throughout all of the testing, regardless of the chosen bot or C&C center that was inspected, the system maintained a common architecture.

The design of the architecture was guided by the research that was conducted on botnets and their C&C centers. In order to set up a botnet laboratory, it was necessary to combine a variety of tools under a single framework, and design a centralized management console in which, the researcher would be able to interact with, and have an overview of the whole system. The centralized management console has a database backend that stores all of the data about the bots that are examined and the test machines that are deployed and used for analysis. Additionally, the console also provides the researcher with management capabilities over the virtual infrastructure that he has deployed. As seen in figure three below, the architecture required multiple isolated private network segments which, were set up using virtual machines deployed in VMWare Workstation. The architecture could easily be deployed with VMWare ESXi server, but workstation was chosen for portability reasons and also requires less demanding hardware.

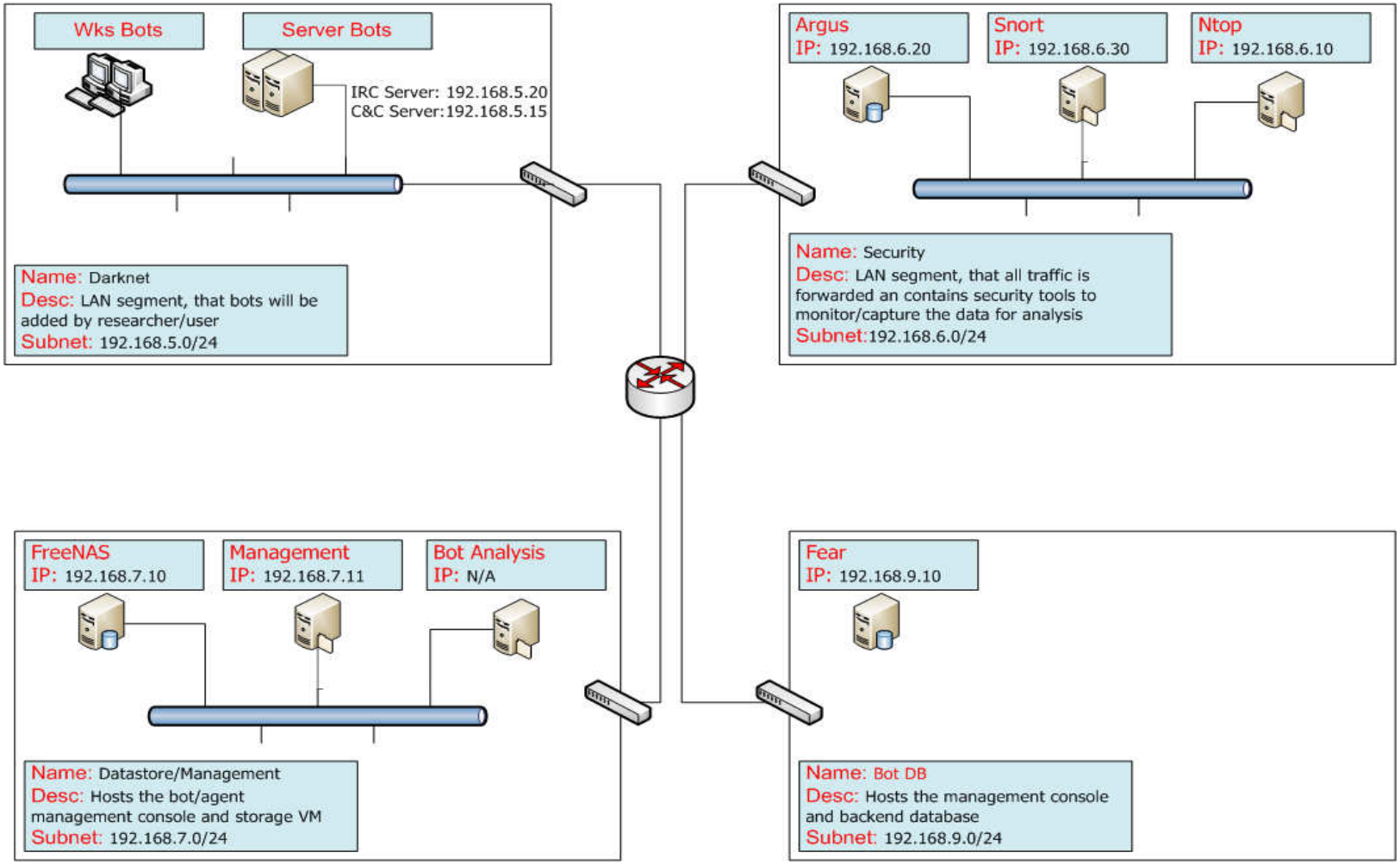


Figure 3. Botnet Architecture

From the architecture diagram, you can see that the focal point for all of the connections is the central router/firewall that is setup using Ubuntu Linux 10.04 LTS server edition which, contains interfaces to all of the networks, and forwards the packets to their final destination. The router has five interfaces one for each subnet within the architecture, and with the use of IP Tables, it isolates the "Darknet" subnet which contains malware.

The "Darknet" is deployed in the 192.168.5.0/24 subnet, and will contain the virtual machines, in which the bots will be deployed and our botnets created. All of the virtual machines will automatically get an IP address assigned from the DHCP/DNS server that is also located in this zone and serves this specific subnet only. Furthermore, in this zone, two additional preconfigured Ubuntu Linux servers exist; the first server hosts an IRC server that can be used to redirect specific bots to communicate and connect with (by using dynamic hosts) and allow for better investigation on that specific bot. The second Linux server can be used to set up any C&C center that the researcher is going to investigate, for example in our testing as you will read in section four, the Zeus C&C center is set up.

All of the traffic that is generated by this subnet which, is categorized as "darknet" is forwarded by the router, to the security network which is located in the 192.168.6.0/24 subnet. In the "Security" subnet, Argus is installed which, audits all of the incoming packets. Additionally, Ntop is configured, to collect statistics about network traffic that will assist us in identifying the communication patterns that are used by the bot that

are being investigated. Lastly, snort IDS is also installed in this network which has been configured with the use of BASE, in order to monitor all traffic generated in the "Darknet" and provides the ability to create and test rules that can be used to secure any network from attacks generated by a specific bot.

Another central point of the botnet laboratory architecture is the "Datastore/Management" network which, is located in the 192.168.7.0/24 subnet, and contains a virtual machine that has installed an operating system called FreeNAS (FreeBSD based) which, provides storage for the deployment of the cloned virtual machines, as well as a repository for storing the bot software that will be deployed and examined. FreeNAS contains a fully manageable web interface, in which the researchers are able to dynamically create additional space if needed, and set up file shares for the exchange of information. Additionally, part of this subnet, which will be used to conduct a detailed forensic analysis investigation, is a preconfigured windows XP workstation which includes all of the necessary tools that are needed for the analysis of a bot and in order to perform reverse engineering on it as well. This workstation, with the use of IP Tables configured on the router/firewall is completely isolated from the rest of the subnets and the machines that are located in its subnet as well.

Lastly, in the "Datastore/Management" subnet, a Linux Ubuntu 10.04 Desktop is deployed, in which the researcher is able to manage all of the servers, access the centralized console, the FreeNAS management

console, plus this workstation has access to the host machine in order to retrieve new tools, and upload bots to the bot repository. The management workstation has all of the links, documents and necessary information that will assist the researcher in getting started on his investigation.

At the heart of the laboratory, the Bot Data server exists which is located in the 192.168.9.0/24 subnet, also referred as the "Bot DB", which runs an apache web server that serves the consoles web pages used for deploying virtual machines and bot software (also uploading). The Bot DB server also has a MySQL Database instance as a backend, that stores all the necessary information about the bots that are located in the repository, the Virtual machines that have been deployed, their status (if they are alive or not), and what bots have been deployed to which cloned virtual machines. In figure four below, you are able to see the Database structure that was deployed.

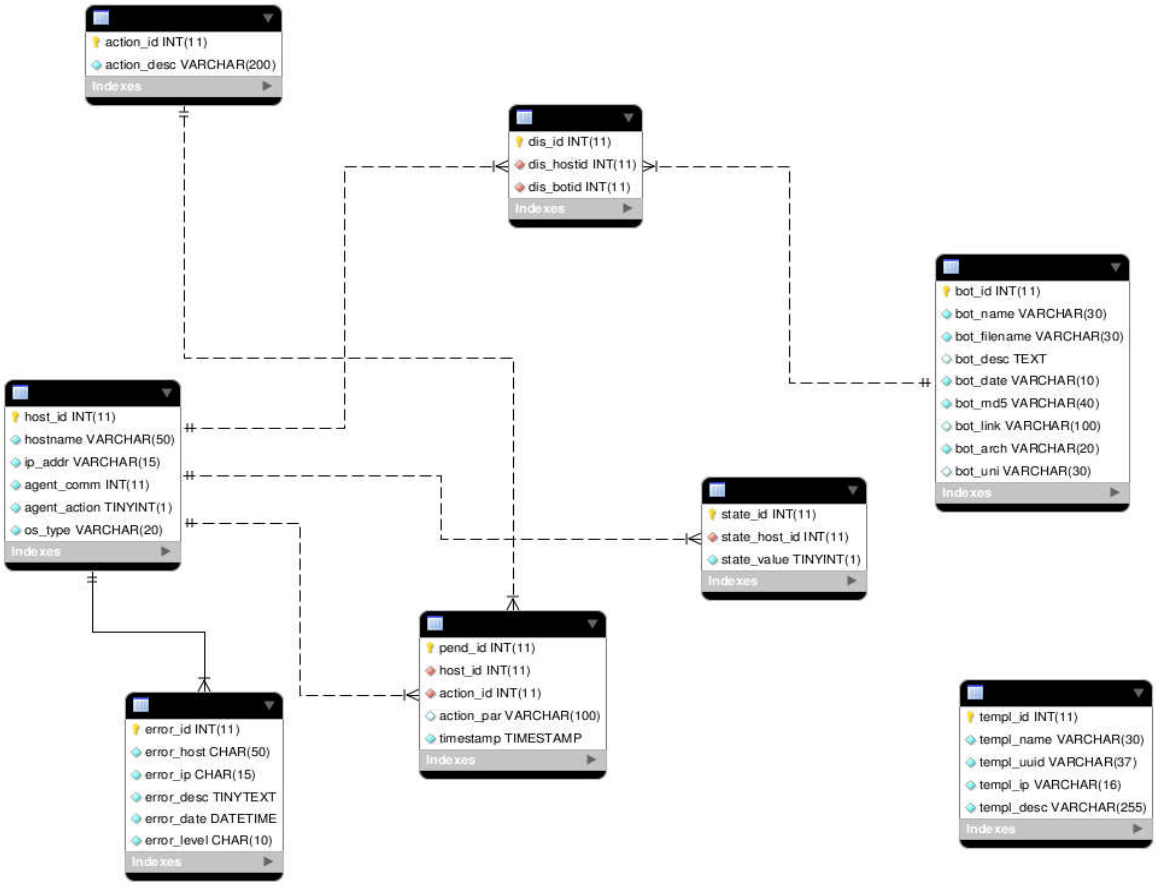


Figure 4. DB Architecture

4 Methodology

4.1 Botnet Setup

As discussed in the previous section, the framework deploys several network segments, which each serves a different purpose which will be explained in detail in the following sections.

4.1.1 Bot and VM Repository/Dynamic Hard Disks

In order to ease the creation of extra hard disk space that will be used to store a large number of virtual machines and bots (isolated), the botnet laboratory will facilitate a virtual machine which has installed a Linux distribution called FreeNAS. FreeNAS provides the user with the possibility of dynamically adding new hard drives and allows us to create file shares (using different protocols such as NFS of CIFS) which can be assigned for specific purposes. Combining FreeNAS with the functionality that is provided by virtual machines to add as much space as you want (limited by the hard disk space available on the host machine), and the web based configuration available on FreeNAS a researcher is able to configure and add as much space as needed for his testing on the fly. On the FreeNAS Virtual Machine we store both the bots that have been collected and are going to be examined are stored, as well as the virtual machines that will be used for the examination. Below in figure five you are able to see the FreeNAS interface and the share that has been created to store the cloned Virtual Machines.

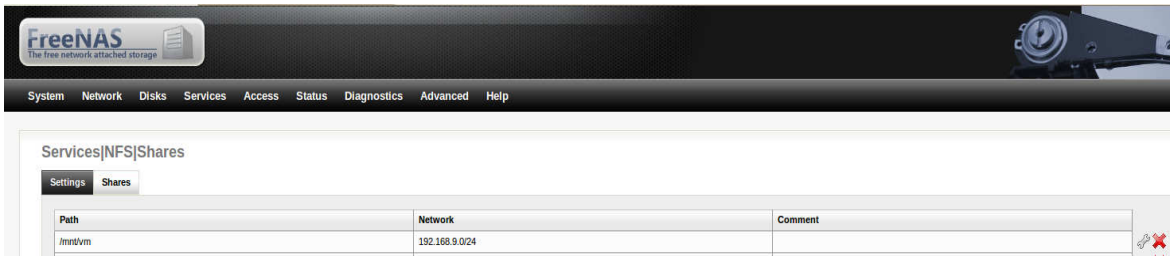


Figure 5. FreeNAS & NFS Share

4.1.2 Guest O.S. / Virtual Machine cloning & management

4.1.2.1 Templates

One of the main components that are provided to the researchers, are template Virtual Images for Windows XP and Ubuntu Linux, which are stored in our repository. These templates are pre-configured and can be cloned and deployed without any additional configuration needed from their part.

In case a researcher desires to use another operating system, for example OpenSuse 10 or Windows 2000, as part of the design, specific scripts have been created that configure both Linux and Windows architectures and make them ready to be used as templates and can be cloned by the researchers.

4.1.2.2 Virtual machine cloning & management

In order to replicate a botnet, a researcher will have to set up multiple virtual machines in order to infect them and monitor their behaviour. In order to assist this procedure, one of the functionalities that the management console (more information about the management console in the below section) provides, is the ability to allow the user to clone the templates that are predefined and available; below you will find

an image of the console's cloning section (figure 6). In order to save time and space, the use of linked cloning has been employed when replicating the templates.

Figure 6 – Cloning section from the management console

Furthermore, the user is able to view the clones that he has created and manage the virtual machines from the cloned section of the management console as seen below (Figure 7). By providing such functionality to the researcher, he does not have to go back and forth in managing the virtual machines, thus allowing him to manage everything from a centralized console.

Figure 7 – Cloned machines section of the management console

4.1.3 Management Console

At the heart of the botnet laboratory is a custom made console with the use of PHP and a MySQL database backend. The Management

console will be called from now on "bot control". Each template when cloned has an agent installed on the guest O.S. (either Linux or Windows based), which is also PHP based. When a user launches a cloned virtual machine the agent sends a signal to the database and checks if it is a clone or not, if the image is a clone it changes the hostname of the virtual image (and SID if a windows machine) and the IP address. Once the virtual image is updated, the agent reports back to the database and keeps on sending keep-alive signals in order to identify the last communication time of the agent. The console lists all the agents and their status as seen below in figure eight below.

Figure 8 – Bot Control Centre

The above image depicts basic information about the agent (virtual image – clone), such as hostname, IP address, OS type active/inactive status and a box that enables you to check it and perform additional functionalities. On this tab you can also view any errors that the virtual machine may generate during its operations, such as installations gone

wrong, or communication errors as well. From the console and once you have selected an active agent, you are able to perform the following basic operations:

1. **Bot Installation**

   Via this option the user depending on the architecture that he has chosen, is able to push and install the desired bot that he was chosen from his repository (either a windows or Linux binary). When a bot is installed on a host, it is registered with the database backend and is shown up in the bot control center as a little bug icon under the status section (as seen in Figure 8 above).

2. **Bot deletion**

   Additionally each bot that is installed can also be uninstalled. The disadvantage of this option is because each bot has its own method of un-installation you will have to create a un-installation package. Due to the difficulty of the task, it is recommended to either delete the virtual machine or clone a new one (which will only take a few minutes), or restore the virtual machine to its base snapshot where it is bot free.

3. **Agent Upgrade**

   Since this is a modular laboratory and it is designed on open source tools and the source code is free to alter, the user is able to perform changes and add-ons to the PHP agent and send a

major upgrade to all hosts, which will receive the functionality when they communicate with the server.

4. **Dynamic Routes (DDNS)**

   The user may desire to force a bot to communicate with his own IRC Server (which is provided as well – more seen in section C&C centers), may have to add a dynamic route to the virtual machine and trick the machine in connecting to our own IRC server for further investigation; This functionality has not yet been implemented.

5. **Agent Start/Stop**

   The user if he desires, is able to shutdown and start the agent at will, thus enabling the features mentioned above or disabling them, depending on the situation.

   NOTE: All communication is initiated by the agent towards the server at a specific time interval (default 5 minutes). You are able to push the communication if needed for testing purposes from the virtual machine.

4.1.4 Command & Control Centres

Each bot tries to connect to its command & control centre in order to get updates and instructions about the actions they are going to perform. Following this format, two servers have been dedicated for this purpose.

One of the servers has an IRC Server installed and with the use of dynamic DNS, you are able to add the server that the bot is supposed to communicate with and talk with your own IRC server, thus giving the researcher when possible the extra edge in gathering more info on the behaviour of the bots. Furthermore, the second server can be used to install any command & control centre that the researcher is currently investigating (for the current tests we have deployed the Zeus bot HTML based command centre).

4.1.5 Security monitoring tools

Since one of the basic parts of the botnet laboratory will be monitoring the traffic and data that the bots are generating and trying to send towards their command and control centres, a set of monitoring tools have been installed, that will "supervise" the network traffic that is generated by the botnet network segment. As mentioned in section 2, all packets that are generated in our darknet (our botnet network segment - with exception the DHCP/DNS), are considered malicious, and are forwarded by the router to the IDS catcher Argus which as stated in their website, "it can processes packets (either capture files or live packet data) and generate detailed status reports of the 'flows' that it detects in the packet stream" [16].

Also in the security monitoring tools zone, BASE has been deployed, which is a frontend to query and analyze alerts that are generated by the SNORT IDS system. Also with the use of SNORT you are able to create rules

based on the information that you have gathered and test them. Finally, NTOP is utilized for sniffing traffic and gathering statistics about the network traffic that is generated by the bots.

4.1.6 Data analysis tools

One of the most important systems of the laboratory is the virtual machine that will be used to analyze the network traffic and the bots that have been collected. This virtual machine will contain two disassembly tools, IDA Pro and ollybg. It will contain all the Sysinternal tools by Microsoft that can be used for analyzing a system. Additional tools have been added such as honeysnap that are used for identifying IRC communication from pcap files, as well as wireshark a packet sniffing program. Lastly, registry tools have been provided and nmap for analysis of targeted systems.

4.1.7 Connecting everything together

Furthermore, all of the above mentioned services that are provided are controlled from a central management virtual server that is able to connect everything together and has all of the necessary data in one place, which is the starting point for every researcher.

Lastly, each separate network segment is connected by a software router which is deployed using Ubuntu server version 10.04 LTS, which also serves as a firewall using IP tables and isolates specific environments that may contain malicious code from the rest of the production servers.

4.2 Testing

The proposed system was tested with the Zeus Bot, which its C&C is HTTP based. The goal of these tests, were to examine the flexibility and usefulness of the laboratory. For the tests that were performed, the diagram below was used, which depicts the flow of how the investigation was conducted.

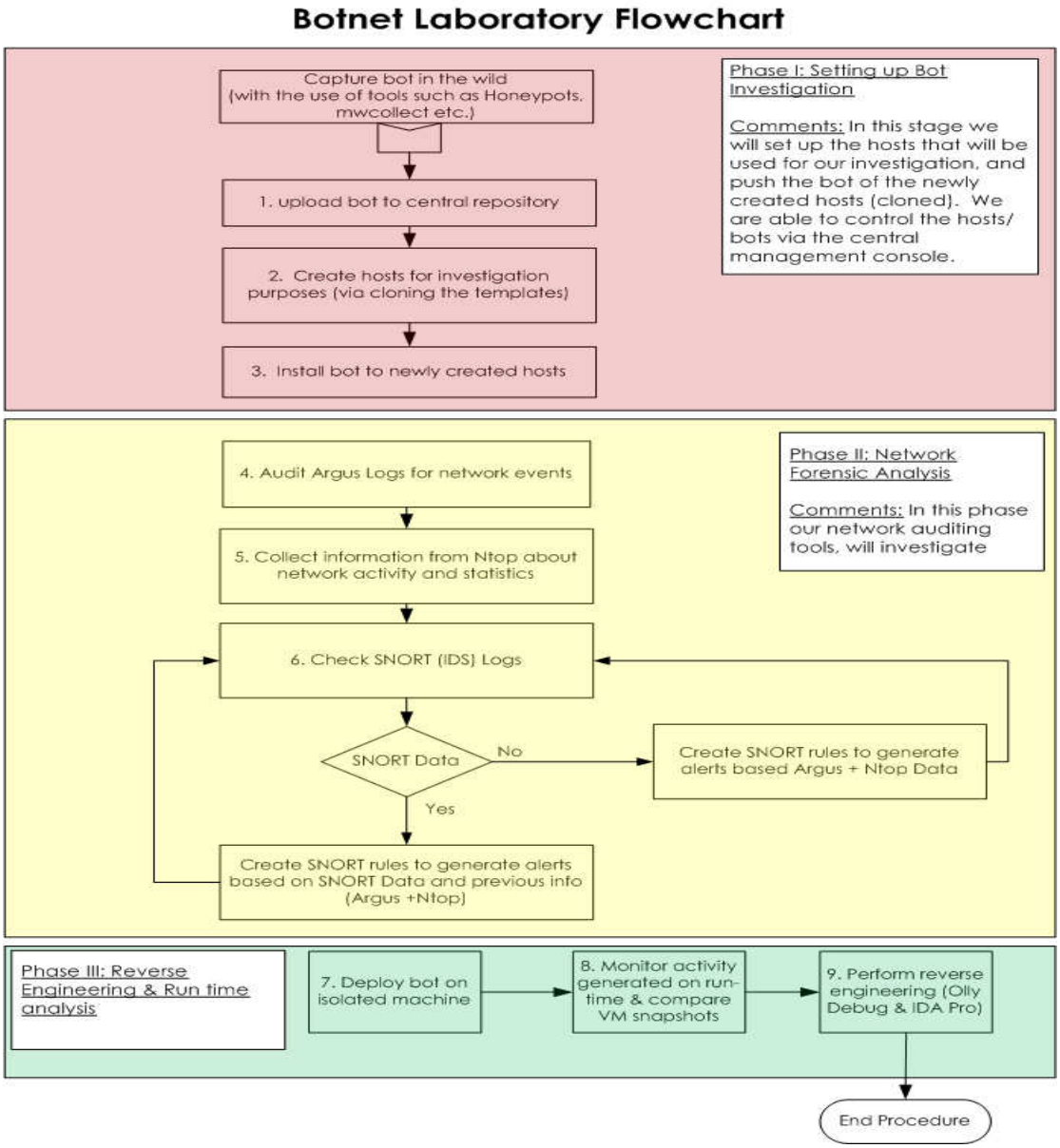


Figure 9. Investigation Flowchart 1

4.2.1 Zeus bot investigation

The analysis of the Zeus bot is divided into two parts, in the first phase the bot itself was analyzed, and in the second phase, the C&C center that was HTML based was investigated.

In the first part of the investigation, the executable was obtained from the wild; the Zeus bot is known in the underground community as the "Do It Yourself Botnet", you are able to compile the bot, set up the communication parameters that the bot will have (C&C address), URLs to log (banking based, email based), and what web pages to spoof and inject code into in order to gather data (see images below).

```
;version:      1.2.4.2

entry "StaticConfig"
  ;botnet "btn1"
  timer_config 60 1
  timer_logs 1 1
  timer_stats 20 1
  url_config "http://192.168.5.15/zeus/config.bin"
  url_compip "http://192.168.5.15/zeus/ip.php" 1024
  encryption_key "star13rvk"
  ;blacklist_languages 1049
end

entry "DynamicConfig"
  url_loader "http://192.168.5.15/zeus/zeus.exe"
  url_server "http://192.168.5.15/zeus/gate.php"
  file_webinjects "webinjects.txt"
  entry "AdvancedConfigs"
    ;"http://advdomain/cfg1.bin"
  end
```

Figure 10. Zeus Bot configuration Sample

```
set_url *.ebay.com/*eBayISAPI.dll?* GL
data_before
(<a href="http://feedback.ebay.com/ws/eBayISAPI.dll?viewFeedback&*">
data_end
data_inject
Feedback:
data_end
data_after
</a>
data_end

set_url https://www.us.hsbc.com/* GL
data_before
<table cellspacing="0" summary="page layout">
data_end
data_inject
data_end
data_after
</table>
data_end
```

Figure 11. Zeus Bot Web Inject configuration Sample

After the bot parameters were set up, and the Zeus executable was built, it was uploaded to the central repository which is located in the FreeNAS server (set in read-only mode). The next step of the investigation process would be to create a set of virtual machines that would be used to infect them with the previously uploaded bot executable. For this specific experiment, we cloned 5 Windows XP virtual machines. With the use of linked cloning, the process of populating the botnet with hosts is a task that lasted a few seconds. After the hosts were powered on, the agent checked if the machines had registered before with the Bot control console, and if they had not, they change their SID and their Hostname/IP so each machine could be unique, identifiable and easy to manage.

When the creation process had been finalized, it allowed us to view the hosts on the main console, in which the Zeus bot was pushed for installation on. After each successful installation, a bot icon appeared next to each host (as shown in figure 8).

The first part of the investigation process, was to check the network devices that were set up, and the network activity that the bots had generated. Due to the fact that all of the hosts had been deployed in the darknet, any activity that was generated, with exception of our legitimate traffic i.e. DHCP & DNS was considered malicious. In the Result and Future Work section, there will be an analysis of the results from the testing and on how this prototype system can be used in the future and how it can be improved upon.

4.3 Results

As mentioned above, In order to test the effectiveness of the system, the testing had to be separated into four different phases: (1) Uploading the newly found bot (2) Setting up the laboratory environment (3) Managing the hosts & Deploying the bots (4) Gathering forensic data.

From the tests that we performed, all four steps of the testing were completed with success. The Zeus bot was uploaded to the laboratory's central repository.

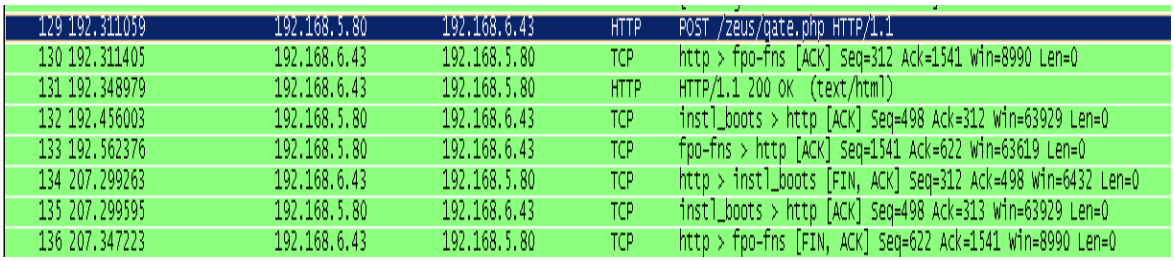
After the Virtual machines has been cloned and launched, they were managed via the central console and it allowed us to deploy the Zeus bot to all of the virtual clones. All of this was accomplished with success via the central web console that has been designed.

The last part of the testing included the investigation of the Zeus bot and the recording of the network activity that it produced. From the sensors that have been set up, and the darknet that is in place, we were able to gather adequate information about the network activity that the Zeus bot is generating with the use of Argus, Snort and Ntop.

After the information had been gathered, we performed a run-time analysis and gathered more information about how the Zeus botnet is working, and with the use of wireshark and VMware snapshots (in order to find the different files that were infected), we were able to gather the necessary information in order to conduct a full analysis.

The second that the Zeus bot was installed, it started to generate traffic and provided us with the ability to see the hosts on ntop, and view the packets, that the bots had generated. The second discovery that was identified and captured was when the Zeus bot tried to contact its C&C server on port 80, in order to retrieve new configuration files and commands, thus allowing Argus and Snort to capture the specific network activity.

Lastly, in order to get all of the necessary information that was needed, a run-time analysis was performed in the analyzer virtual machine. After the Zeus bot was installed, we were able to monitor a lot of activity towards the designated C&C center and captured a lot of connections towards that specific IP address (figure 12), as well as a TCP connection via netstat towards the C&C IP address (figure 13). With the use of various tools, we were able to identify that the bot had included three hidden files under the c:\windows\system32 directory which had to be deleted in order to remove the bot completely.



129 192.311059	192.168.5.80	192.168.6.43	HTTP	POST /zeus/gate.php HTTP/1.1
130 192.311405	192.168.6.43	192.168.5.80	TCP	http > fpo-fns [ACK] Seq=312 Ack=1541 Win=8990 Len=0
131 192.348979	192.168.6.43	192.168.5.80	HTTP	HTTP/1.1 200 OK (text/html)
132 192.456003	192.168.5.80	192.168.6.43	TCP	instl_boots > http [ACK] Seq=498 Ack=312 Win=63929 Len=0
133 192.562376	192.168.5.80	192.168.6.43	TCP	fpo-fns > http [ACK] Seq=1541 Ack=622 Win=63619 Len=0
134 207.299263	192.168.6.43	192.168.5.80	TCP	http > instl_boots [FIN, ACK] Seq=312 Ack=498 Win=6432 Len=0
135 207.299595	192.168.5.80	192.168.6.43	TCP	instl_boots > http [ACK] Seq=498 Ack=313 Win=63929 Len=0
136 207.347223	192.168.6.43	192.168.5.80	TCP	http > fpo-fns [FIN, ACK] Seq=622 Ack=1541 Win=8990 Len=0

Figure 12. Wireshark Capture towards 192.168.6.43

	Proto	Local Address	Foreign Address	State
13	TCP	0.0.0.0:135	0.0.0.0:0	LISTENING
71	TCP	0.0.0.0:445	0.0.0.0:0	LISTENING
59	TCP	0.0.0.0:29450	0.0.0.0:0	LISTENING
05	TCP	192.168.5.80:139	0.0.0.0:0	LISTENING
79	TCP	192.168.5.80:1068	192.168.6.43:80	CLOSE_WAIT
03	UDP	0.0.0.0:445	*:*	
76	UDP	0.0.0.0:500	*:*	
63	UDP	0.0.0.0:1032	*:*	
95	UDP	0.0.0.0:4500	*:*	
23	UDP	127.0.0.1:123	*:*	
72	UDP	127.0.0.1:1033	*:*	
	UDP	127.0.0.1:1900	*:*	

Figure 13. Netstat Connection towards 192.168.6.43

The next step was to inspect the C&C center that had been set up in the bot server section. After carefully inspecting the different options that the C&C had, we were able to understand how the bot operated, and the different commands, that I we able to apply to the bot, as well as the data that the bot uploaded to the server about the host (websites viewed, passwords, cookies etc.). The Zeus bot, is a information gathering tool, so the reports that it gathered, were passwords, user information, websites visited, cookies and also it provided us with the ability to run specific commands on the infected machines (see figure 14).

Available commands	
reboot	Reboot computer.
kos	Kill OS.
shutdown	Shutdown computer.
bc_add [service] [ip] [port]	Add backconnect for [service] using server with address [ip]:[port].
bc_del [service] [ip] [port]	Remove backconnect for [service] (mask is allowed) that use connection to [ip]:[port] (mask is allowed).
block_url [url]	Disable access to [url] (mask is allowed).
unblock_url [url]	Enable access to [url] (mask is allowed).
block_fake [url]	Disable executing of HTTP-fake/inject with mask [url] (mask is allowed).
unblock_fake [url]	Enable executing of HTTP-fake/inject with mask [url] (mask is allowed).
rexec [url] [args]	Download and execute the file [url] with the arguments [args] (optional).
rexeci [url] [args]	Download and execute the file [url] with the arguments [args] (optional) using interactive user.
lexec [file] [args]	Execute the local file [file] with the arguments [args] (optional).
lexeci [file] [args]	Execute the local file [file] with the arguments [args] (optional) using interactive user.
addsf [file_mask...]	Add file masks [file_mask] for local search.
delsf [file_mask...]	Remove file masks [file_mask] from local search.
getfile [path]	Upload file or folder [path] to server.
getcerts	Upload certificates from all stores to server.
resetgrab	Upload to server the information from the protected storage, cookies, etc.
upcfg [url]	Update configuration file from url [url] (optional, by default used standard url)
rename_bot [name]	Rename bot to [name].
getmff	Upload Macromedia Flash files to server.
delmff	Remove Macromedia Flash files.
sethomepage [url]	Set homepage [url] for Internet Explorer.

Figure 14. Zeus Commands

With the information that was gathered, we are able to go back and create the necessary rules on Snort in order to create an effective defence on the Zeus botnet attack pattern. Additionally, we are able to set up firewall rule to block all connections coming from these hosts based on the information that we obtained from our network analysis.

4.4 Future Work

While the initial results were very promising there are several fields that need to be improved upon, in order to make the system more automated and more useful for researchers in order to deploy, configure and use for testing.

4.4.1 Further testing

While the laboratory was used to test the Zeus bot which uses a HTTP based C&C center, it was not tested for a different kind of bot which uses another type of C&C such as IRC. This can easily be done by discovering an IRC bot and deploying it on newly created test machines and monitoring the traffic that the bot generates.

4.4.2 Central Control Panel

The central control panel needs some extra improvements, in creating dynamic hosts, and managing the hosts that are created (VMware control). Additionally, since the configuration is currently fixed, a centralized configuration page would be necessary, in order to adapt the framework to a researchers requirements.

4.4.3 Monitoring Tools

The investigation that is being conducted on the bots, is currently done manually, this can be modified and with the use of scripting we could be able to bind these tools (Argus, Snort, Ntop), to report to a centralized database and create a report, about suspicious activity and provide us with details about the attacks that the bot is performing and the patterns that it is following. Additionally, the run-time analysis set-up will need to be improved upon, and add additional tools in order to provide greater analysis possibilities.

5 Conclusion

The research presented here, demonstrates the use of a laboratory that can be used to investigate bot behaviour. This system, can easily be modified and used for malware analysis, and does not have to be limited to bot/botnets.

Furthermore, we have demonstrated the ability of the system, to dissect a bot that is installed on specific hosts, and to manage the hosts that have been deployed, in a safe isolated environment. The proposed system is able to provide an abundance of tools and options for any researcher interesting in performing research on botnets. Due to the fact that the use of open-source tools has been utilized in the design of the laboratory, it is highly flexible, and you are able to modify it according to the requirements of each researcher. The advantage of this laboratory is that you are able to deploy multiple templates that you will be using for your investigation and you are able to deploy the bots from a centralized console, instead of having to have access to each separate virtual machine, which makes the job of a researcher a lot easier. If you bundle together the functionalities mentioned above, with the network and forensic tools, used by researchers in the field, the laboratory is able to provide a central point for each researchers/security analyst's investigation.

While the current prototype laboratory developed is good starting point, it may have some weaknesses, which need to be addressed in the future. As discusses, there is a need for a central configuration console that will allow the researcher to customize and make appropriate modification when deploying the laboratory. Due to the fact that the laboratory is developed on VMware Workstation, it provides limited remote control over the deployed VMware images (i.e. Start/Stop/Reset/Power off), which will be addressed in the future when the framework will be deployed using VMware Server or ESXi. Currently, the setup of the laboratory, can successfully detect HTML & IRC traffic without any modifications, but in the future more emphasis will be added in using appropriate tools to detect P2P, encrypted traffic, and legitimate traffic (i.e DNS) used by bots to hide their communication methods.

Lastly, a larger template base will be added for the user to have a default setup, and a larger collection of existing bots will be added. As future plans, also the ability to sync the laboratory with tools such as mwcollect to automatically add newly found bots into our repository and provides the researcher, with a function to set up a new investigation on the fly.

References

[1] ShadowServer Foundation. (2010, August 27). *What is a Botnet?* Retrieved October 7, 2010, from Shadow Server foundation Web site: http://www.shadowserver.org/wiki/pmwiki.php/Information/Botnets

[2] Honeynet Project. (2010, August 10). *Know your Enemy: Tracking Botnets*. Retrieved September 13, 2010, from Honeynet Project Web site: http://www.honeynet.org/papers/bots/

[3] Ianelli, N., & Hackworth, A. (2005, December 1). *Botnets as a vehicle for online crime*. Retrieved from http://www.cert.org/archive/pdf/Botnets.pdf

[4] Gu, G., Zhang, J., & Lee, W. (2008, February 10). Botsniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*.

[5] Li, Z., Goyal, A., Chen, Y., & Paxson, V. (2009, March 10). Automating analysis of large-scale botnet probing events. In *Proceedings of the 4th International Symposium on Information,*

Computer, and Communications Security conducted at Sydney, Australia.

[6] Barford, P. & Yegneswaran V. (n.d.). *An inside look at botnets* (Master's thesis, University of Wisconsin, Madison). Retrieved from http://pages.cs.wisc.edu/~pb/botnets_final.pdf

[7] DETERlab Testbed. (n.d.). Retrieved October 23, 2010, from http://www.shadowserver.org/wiki/pmwiki.php/Information/Botnets

[8] WAIL (n.d.). Retrieved October 24, 2010, from http://wail.cs.wisc.edu/desc.html

[9] Chen, P. M., & Noble, B. D. (2001, May). *When virtual is better than real*. Retrieved from http://portal.acm.org/citation.cfm?id=876409

[10] Wasniowski, R. A. (2005, September 23). *Multi-sensor agent-based Intrusion detection system*. Retrieved from http://portal.acm.org/citation.cfm?id=110764

[11] Defense Intelligence. (2009, October 8). *Mariposa Botnet Analysis*. Retrieved from http://defintel.com/docs/Mariposa_Analysis.pdf

[12] Landecki, G. (2001, January 1). *Detecting botnets*. Retrieved October 30, 2010, from http://www.linuxjournal.com/magazine/detecting-botnets?page=0,0

[13] Harrop, W., & Armitage, G. (2005). *Greynets: A definition and evaluation of sparsely populated darknets*. Retrieved from http://portal.acm.org/citation.cfm?id=1080177

[14] *Ntop*. (n.d.). Retrieved November 2, 2010, from http://www.ntop.org

[15] Ponec, M., Giura, P., Wein, J., & Bronnimann, H. (2007). *New payload attribution methods for network forensic investigations*. doi:10.1145/1698750.1698755

[16] *Argus-Auditing network activity*. (n.d.). Retrieved November 10, 2010, from http://www.qosient.com/argus/

[17] Mahathi Kiran Kola (2008). *Botnets: Overview and Case Study*. Retrieved from https://www.mercy.edu/ias/kola.pdf