Botnet Detection and Analysis Using Honeynet

M .C. Sacchetin, A.R. A. Gregio, L. O. Duarte and A. Montes, *CenPRA*

***Abstract*—In this work we discuss some new techniques used by intruders to control a group of compromised machines (botnets). It is also shown how honeynets can be used to identify, monitor and understand current botnets behavior. We outline a real case of compromise, detailing analysis techniques specially developed for botnets study, including the tools, the topology and strategies adopted, as well as the results obtained in the use of honeynets to identify botnets.**

***Index Terms*—botnet, computer network security, honeynet, site security monitoring**

I. Introduction

THE continuous growth of Internet services and resources in last years has been motivating illegal activities through the cyberspace, leading to a substantial increase in attacks and computer intrusions. Initially, underground hacker groups were motivated by communication and information sharing needs among themselves, and the curiosity to learn more about computer systems. But, with the increase of financial transactions and electronic commerce through computer networks, criminals changed their goals to obtain financial resources illicitly using the Internet [6]. Those malicious activities demand that more sophisticated attacks are launched against computers in the Internet.

Thus, computers that have large bandwidth capacity and long uptimes turn into easy targets to attackers, such as servers located at universities, enterprises or even certain home users. The larger the number of systems controlled by an attacker, the greater his power to perform malicious activities, so the need aroused of mechanisms to control large groups of machines. A set of compromised computers under an attacker's control is called botnet.

The term botnet is a junction of the contracted words robot (bot) and network (net). Most botnets use the IRC protocol in order to exchange information between the controller and the botclients, which are also known as bots, IRC bots, drones or zombies [1]. There are several kinds of bots available, such as Agobot, SDBot, Spybot, GTBot and Eggdrop with different levels of sophistication related to command and control, but generally consisting of a client (the bot) that connects to an IRC server in a predetermined channel and waits for commands from a controller [3].

A typical botnet works as follows: once a system is compromised, the attacker downloads the bot from his malware repository and installs it in the machine. When the bot runs, it connects to one or more channels in an IRC server and waits for commands. Usually, the command launching process follow a logical sequence, presented below:

-- First, the controller sends a command to log in the bot that serves to "recognize" the bot among possible watchers;

-- Then, the controller launch scans against networks, passing IP ranges, scanning interval and other parameters;

-- These scans help to identify vulnerable systems that will be targets of future attacks;

-- The controller launches attacks against vulnerable systems sending commands to the chat channel or channel topic;

-- If these attacks are successful, the bots propagate to other computers;

-- The recently installed bot tries to connect to a channel in a server, closing the botnet cycle.

With this sequence, a botnet controller can obtain, keep and control a reasonable number of machines to send SPAM, launch scans and attacks, capture keystrokes, maintain a malware repository, etc. This constant exploit cycle avoids substantial decreases in the number of botclients, even considering that some hosts lost communication with the IRC server, are turned off or the compromise is identified.

The motivation for the study of botnets and identification of the techniques used to build them resides on the curtailing of criminal activities committed using these architectures. The efforts expended aggregating zombies to build a botnet in general have the following objectives:

-- Information gathering: some bots have the ability to capture keystrokes, screenshots, files, network traffic and others. These functionalities are useful to identity theft, bank cards information collection, gathering of strategic and commercial documents from the competition, etc.

-- Increasing computer power for Distributed Denial of Service attacks (DDoS): it is very common the use of botnets to launch ICMP ECHO, TCP or UDP floods in order to consume resources and make machines unavailable. The

Manuscript received September 15, 2007

M. C. Sacchetin is with the Centro de Pesquisas Renato Archer, Campinas, SP CEP: 13069-901 Brasil (phone: 37466077; e-mail: marcelo.sacchetin@cenpra.gov.br).

A. R. A. Gregio is with the Centro de Pesquisas Renato Archer, Campinas, SP CEP: 13069-901 Brasil (phone: 37466077; e-mail: agregio@cenpra.gov.br).

L. O. Duarte is with the Centro de Pesquisas Renato Archer, Campinas, SP CEP: 13069-901 Brasil (phone: 37466077; e-mail: loduarte@cenpra.gov.br).

A. Montes is with the Centro de Pesquisas Renato Archer, Campinas, SP CEP: 13069-901 Brasil (phone: 37466077; e-mail: antonio.montes@cenpra.gov.br).

use of a large the number of compromised computers, makes flooding much more effective.

-- SPAM forwarding: the distribution of unsolicited e-mail (spam) is another activity that can be done using botnets, since a large number of machines sending SPAM at the same time allows greater distribution coverage and is harder to block.

-- Malware repository: botnets controllers need resources to keep their tools easily available throughout the Internet. In this case, some of the compromised machines can be used as repositories, as have been observed botclients accessing such machines and downloading malware from FTP or HTTP servers. The distribution of these tools among several machines guarantees the availability of these malware even if some bots are lost.

-- Illegal content hosting: botnets are also used to store illegal contents, such as phishing sites, stolen information (files, documents, and credit card numbers), porn and child porn.

-- Anonymity: the use of several machines around the globe as stepping-stones to access compromised hosts rendering it difficult to perform a traceback to the real attacker or botnet controller.

All the uses above explain why botnets are one of the main worries of the information security community nowadays [9]. The first step to mitigate this kind of threat is to deeply understand the internal working of botnets and the techniques employed by them. Thus, this work presents an approach to the study and gathering of information about botnets.

II. Related Works

In the current literature, there has been a lot of discussion about honeynets and botnets as distinct subjects. Several works describe important features and differences among the most well known bots, as can be seen in [1], [3], [6] and [7]. However, new techniques currently observed in botnets, such as the use of encrypted control commands, are not usually seen in papers. In this work we discuss how honeynets can be used to identify and study these new techniques.

In [7] it is explained how botnets can be used to commit crimes and there is a general description of other types of control mechanisms differing from the usual IRC approach. In [3] there is a complete description of the four most used bots currently: Agobot, SDBot, SpyBot and GT Bot. Also, it describes the architectures, how the control mechanisms work, how they spread and the attacks employed by each of them.

In [1] it is explained how a honeynet can be used to examine the operation of botnets and a general description of some kinds of bots is presented. In this paper we use a similar approach, but in addition to describing an identification and analysis methodology adequate to the bots variety seen in [1], our approach is also able to analyze new kinds of bots that use encrypted commands in order to difficult their detection and identification.

From the innumerable possible ways to compromise a honeypot, such as exploiting web services vulnerabilities [6], in this work we show the compromise of honeypots through vulnerabilities in NETBIOS protocol-based services.

In Section III, the honeynet architecture deployed to facilitate botnets study is presented. The methodologies and tools used in the analysis of captured bots are described in Section IV. The isolated environment (sandbox) to study bots is presented in Section V. In Section VI, we present a real case study of a botnet analysis and the results obtained. Sections VII and VIII contain, respectively, other bots analysis results and the paper conclusion.

III. Using Honeynets to Study Botnets

The first step of this study is the identification of botnets. Deploying honeynets has been an effective way to research botnet malware by allowing it to compromise and install botclients.

One of the main advantages of using honeynets is that all the traffic related to them is monitored and logged [14]. Thus, all the intruder's actions, from exploiting the vulnerability to the attempts to perform illegal activities with the compromised machine, are collected and used later for analysis in an isolated environment.

Figure 1 shows the honeynet architecture and the isolated environment used for botnet analysis. All incoming traffic from the Internet passes through a firewall device before reaching the honeypots and vice-versa. In the firewall, containment mechanisms have been deployed to avoid the use of the honeynet as launching point of attacks against other networks.

In this architecture, all traffic is monitored in a transparent way by an Intrusion Detection System (IDS) and copied into a file server, called Log Host. The main goal is to gather passively the maximum of information, from which one can infer the techniques used to compromise the honeypots, capture malware, locate malware repository, bot controllers IP addresses, other bots IP addresses and so on.

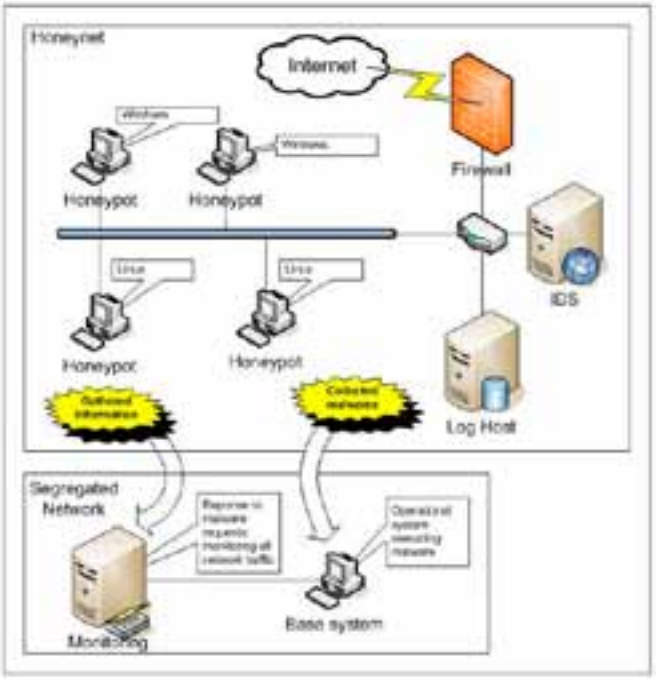

Fig. 1. Honeynet architecture. All information gathered is sent to the isolated analysis environment.

In the next sections the techniques used to obtain botnets information by means of the honeynet and the use of this information to perform the sandbox analysis is detailed.

IV. Analysis Methodology

Several events are registered by the Brazilian Honeynet on a daily basis. Different attacks from all over the globe are identified and analyzed, allowing us to report incidents to the authorities and competent organizations. Due to the increase use of botnets for illegal activities, the Brazilian Honeynet team has been paying special attention to events related to them. Aiming to facilitate the study of botnets, a methodology had been developed to analyze these events.

The first challenge we face was to identify among all attacks those that were related to botnet activities. Due to the great amount of IRC-based bots captured by the honeynet, we defined this kind of bot as our work focus. Thus, it was necessary to elaborate practical and effective ways to identify IRC-based bots activity inside the honeynet.

Using tools like honeysnap [19], ngrep [11], tcpdump [16], Smart [2] and Sebek [17] to handle honeynet's collected data, it is possible to identify the presence of a bot installed in a honeypot. There are some other techniques more sophisticated that can be used to identify a network traffic that is related to botnets activities, such as Machine Learning [8], however, applying such techniques demand the modeling of an environment where statistical methods are applied in order to accomplish satisfactory results. This kind of technique is out of the scope of this work.

Once the presence of a bot is confirmed through the use of previously cited tools, the next step is to discover all the features and actions performed by the intruder, which is accomplished via the analysis of the compromised honeypot. In this analysis, all commands executed by the intruder during the attack and the information exchanged between the botnet controller and the botclient – such as the IRC server address, the channel name and its password and the server ports – are collected. We also try to recover the malware used either by forensic analysis of the compromised machine, or by extracting it from the network traffic, or even by downloading it directly from the repository with the user and password collected, if it is still up and running. With the malware captured, the process of dynamical analysis can be done in a sandbox built for this purpose. An example of this process is presented in Section 5.

A. *Tools*

In the next subsections we are going to provide a general description of the tools and how to use them together to obtain a methodology to identify and analyze IRC-based botnets.

Norton Ghost

The Norton Ghost tool [15] is used to produce Windows operating system images, allowing the restoration of the system state when needed. This tool is often used to restore Windows systems from failures or compromises that make these systems unusable.

This tool can be used in malware dynamical analysis in the following way: a default Windows installation performed, the current state is saved using the tool and the malware is executed in the system. After finishing the analysis, the hard disk is zeroed and the tool can be used again to restore the system to the initial state in order to perform a new analysis.

However, as Norton Ghost is commercial software and supports only Windows systems, we choose to migrate to the Partimage tool.

Partimage

The Partimage tool [5] is very similar to Norton Ghost, with the difference that it is free and linux/unix-based. With this tool, one can generate either Linux or Windows systems images, as it supports the filesystems: ext2fs/ext3fs, Reiserfs, FAT16/32, NTFS (experimental), and others.

This tool served as a basis for operating systems restoring functions used in the sandbox. Due to the fact that it is linux-based, the deployment of automatic restoring mechanisms for the systems where the bots are executed was very easy. Besides the NTFS support is related as experimental by the developers, no problems occurred during the works and tests performed.

Honeysnap

It is a very useful tool to parse pcap-based network traces. Honeysnap generates summaries of activities in a time interval, listing packets, HTTP sessions, e-mails, etc. It is also capable of extracting FTP or HTTP downloaded files, which is very convenient in cases where intruders delete their malware

after use. Another feature is the summarization of IRC sessions and keyword search.

Tcpdump

Tcpdump is one of the most used traffic analyzers tool. It is capable of capturing all traffic in a determined collision domain in a TCP/IP network, storing all packet information for use by the security analyst. It is also possible to save all the traffic content in pcap-format files, which is the standard format for most of the network analysis tools.

The network traffic can be analyzed using logical filters known as BPF filters. These filters allow better understanding of the events, as one can separate them by ports, IP addresses, protocols and others.

Snort

Snort [13] is a free network-based Intrusion Detection System commonly used by the security community. It works with a known attack pattern's database, called signatures, which are compared with the packets in the network traffic being analyzed by the tool. In the case of a match, an alert is issued. These alerts are used as an initial step to analyze an event in the honeynet and, after the attack alert is confirmed, a deeper analysis is performed.

Ngrep

The ngrep tool allows the search for regular expressions or hexadecimal characters in network traces, supporting pcap-based files. It works with Ipv4/6, TCP, UDP, ICMPv4/6, IGMP and other protocols, and supports BPF filters like tcpdump.

This tool is very useful to find keywords related to IRC traffic in the honeynet network traces. Few commands are enough to identify the IP addresses and through which ports the bots are generating network traffic.

Sebek

Sebek is a tool designed to capture honeypots data in order to register all intruder's activities in a stealth manner. All data and keystrokes produced by the intruder in the compromised host are captured and sent to the log host.

This information constitutes a way to confirm the honeypot compromise, providing a general view about the actions performed by the intruder. Plenty of important information for bots analysis is captured using Sebek.

Smart

This tool works in a similar way to Sebek to capture data generated by intruders in compromised honeypots, but it also provides information about the executed commands responses. This tool was developed by the Brazilian Honeynet Project during the deployment of the Project and is described in [2].

Shell Scripts

There are also some shell scripts that run in the IDS to help the analyst in management tasks related to honeynet. Log rotation activities are performed daily, weekly and monthly to avoid disk space depletion in the IDS due to full time data collection.

The control of monitoring mechanisms, snort, tcpdump, Smart and Sebek is also done via scripts, guaranteeing that every alert generated by one of these tools is stored for analysis. There are also scripts to keep the honeynet working even after communication failures or power shortage.

B. Working with Tools

The first information source to be verified is the IDS alerts. All traffic gathered in honeynet is considered as malicious since honeynets do not provide any service.

Scripts executing on IDS send daily summaries via email to the honeynet manager. These summaries have important information as a list of all IP addresses that had accessed honeypots during the day. Also the honeynet manager receives by email activities description tables that:

-- Lists the top host access on specific ports;
-- Protocol (TCP, UDP, ICMP) access statistics;
-- Top attacker's source IP addresses (packet count);
-- Top attack source Operational System, and type of attack according to snort database;
-- List of suspicious backdoors and botnet activities on compromised honeypots.

Figure 2 shows a typical daily summary received by honeynet manager.

All these information, give honeynet manager a global view of what is going on the honeynet, and serve as a starting point for a deeper analysis considering all previous gathered data.

Fig. 2. Honeynet daily summary. Honeynet manager receives by email an overview of all suspicious activities on honeynet for the past 24 hours.

It is important to mention that the IDS also reports immediately attackers activities inside a honeypot. That is, if a honeypot is compromised, the manager receives immediately an email like that shown in Figure 3. A script running on the IDS crontab provides this feature, which checks every five minutes for received Sebek, Smart or modified bash data related to intruder activities in a honeypot.

We see in Figure 3 that in honeypot XX.XX.XX.37 commands are executed aiming to open an IRC channel (JOIN #aDDa), and connections attempts are made to FTP servers XX.XX.XX.178 and XX.XX.XX.98 probably to download malware.

The data generated by Smart confirms the intrusion [21]

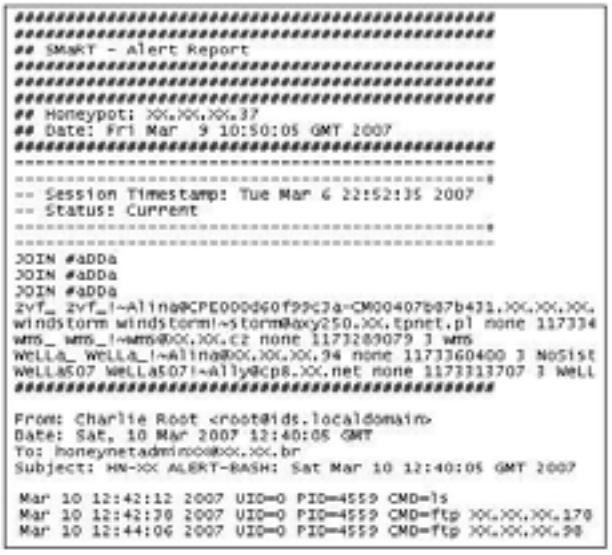

Fig. 3. Smart alert. Honeynet manager receives by email an on time alert when intruder gain access to honeypot.

suspicion alerted by daily summary on bottom of Figure 2. So the analyst who is now sure that honeypot XX.XX.37 has been compromised and that a botclient has been installed. One can now start the search for information related to Botnet control protocol witch is the focus of this paper.

The analyst may want to know more about the server that the honeypot is connected to. Up to now he knows just that the honeypot XX.XX.37 is trying to join the channel "#aDDa" available in a still unknown server. The server IP can be easily found using ngrep tool searching for string "aDDa" with the BPF filter "host XX.XX.37".

Assuming that the dump file containing the entire honeynet network traffic on march 10th has the name "/var/log/tcpdump/dump_file20070310", the analyst can use the following command:

```
#ngrep -I /var/log/tcpdump/dump_file20070310 aDDa
host XX.XX.XX.37
```

The previous ngrep command generates output lines similar to the one below:

```
T XX.XX.XX.37:1034 -> XX.XX.XX.80:10324 [AP]
JOIN #aDDa
```

Alternately, it is possible to search for other common IRC commands such as "PING", "PONG", "JOIN", "NICK", "TOPIC", etc. However the previous output of the ngrep command is enough to conclude that the honeypot has connected to server XX.XX.80 on port 10324.

At this point, the entire bot client and server traffic can be analyzed via tcpdump with the following command:

```
#tcpdump -X -s 1500 -nr
/var/log/tcpdump/dump_file20070310 host XX.XX.XX.80
and host XX.XX.XX.37 and port 10324
```

Analyzing the traffic between client and server, one can harvest lots of information related to the botnet, such as list of the channels that the bot tries to connect, the channels password, channel topic, and nicknames used by the bots.

Despite all the information provided by tcpdump, the honeysnap tool can also be used, since it generates a much more user-friendly output data format. Honeysnap can remount an entire IRC session based on the network dump file, so data is displayed like in a common IRC client program.

The following command tells honeysnap to interpret pcap file containing botnet traffic:

```
#honeysnap /var/log/tcpdump/dump_file20070310 -H
XX.XX.XX.37 --do-irc --irc-ports=10324
```

Honeysnap output is illustrated below when the bots analysis are presented. The section V illustrates how we use a sandbox to have a detailed understanding of the botclient behavior.

V. SANDBOX

A very common problem when analyzing a botnet comes from the use of static information, like the same source IP address with the same nickname always trying to join the same channels. This behavior pattern may alert the botnet controller that he is under some kind of surveillance. Another suspicious behavior is the presence of IP addresses coming from anonymous network, such as TOR [18].

If the botnet controller suspects he is being monitored, he can blacklist the source IP address and nickname, and might inform his associates about the incident. So, honeynet managers need to be careful when analyzing botnets, avoiding the identification of their activities by botnet controllers.

This can be done through the use of a sandbox that isolates the analysis environment from the botnet controller. Initially virtual machines were used as sandboxes, but it is fairly common for malware, such as botclients, to employ some kind of packaging technique protecting the binary content and avoiding file execution when it detects a virtual machines environment [10].

Another concern is the analysis response time. There are some techniques that allow botnet controller to modify their malware in short time intervals [20]. So if the analysis takes too long the results can be erroneous, since the botnet controller is able to modify his artifact before the analyst work

is completed.

Taking into account all the concerns above to avoid botnet analysis disclosure, obfuscation methods in many malware, and analysis time restrictions, a sandbox has been developed able to analyze a bot in a short period of time and avoid botnet controller suspicions.

Two machines compose the sandbox. In the monitoring machine an IRC server has been installed to be accessed by the botclient. It is a Linux system destined to monitor all network traffic and isolate the sandbox from the Internet. So, this machine also executes tcpdump, snort, honeysnap and ngrep. IRC server configuration is based on information previously gathered from the initial analysis, which provided IRC server name, channels, passwords, nickname, etc.

In another machine is installed a base operational system, Windows or Linux depending on the captured malware. The disc is zeroed with dd tool [4] before O.S. installation. After installation, the system disc image is copied to the monitoring system, and the image is restored every time the analyst wishes to do a new analysis. After restoring the base system, the botclient binary is copied to the machine and executed.

Figure 4 presents the sandbox environment.

This environment assures that all traffic is generated by the botclient being analysed (with the exception of some O.S. native traffic, such as Windows NBT packets). This technique reduces a lot the efforts and time expended during analysis.

Sandbox provides resources to investigate how bot server and client communicates, how botnet controller sends

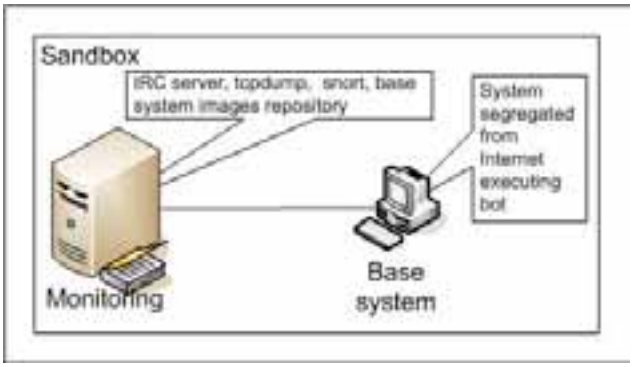

Fig. 4. Sandbox environment. The sandbox is composed by two machines isolated form Internet.

commands to bot, the meaning of those commands, and what are the botclient response for each command (such as start a network port scan, harvest base machine information, etc).

In the next section is presented a real analysis using all resources and techniques previously described.

VI. ZIP0 BOT

On January 17th honeynet manager received an email from IDS with information about a honeypot compromised by the Sasser worm [12]. The corresponding daily summary shows honeypot XX.XX.164 to present outgoing traffic in many ports, including port 80.

All packets exchanged have been listed with tcpdump and the contents have been verified and it was noticed that it was not a common HTTP communication, despite being directed to port 80.

Using ngrep tool, it was possible to conclude that the traffic was related to an IRC communication. Honeysnap provided rich visualization of these data:

```
Wed Jan 17 00:29:38 2007 XX.XX.XX.164:4251 -> XX.XX.13.91:80
pass None <zip0.compresspass>
Wed Jan 17 00:29:39 2007 XX.XX.XX.164:4251 -> XX.XX.13.91:80
nick None [0]USA|0309293
Wed Jan 17 00:29:39 2007 XX.XX.XX.164:4251 -> XX.XX.13.91:80
user None eexakewk 0 0 [0]USA|0309293
Wed Jan 17 00:29:40 2007 XX.XX.13.91:80 -> XX.XX.XX.164:4251
welcome zip0.rar0 [0]USA|0309293 Welcome to the zip0 IRC
Network [0]USA|0309293!eexakewk@XX.XX.XX.164
Wed Jan 17 00:29:40 2007 XX.XX.13.91:80 -> XX.XX.XX.164:4251
yourhost zip0.rar0 [0]USA|0309293 Your host is zip0.rar0,
running version Unreal3.2
Wed Jan 17 00:29:40 2007 XX.XX.13.91:80 -> XX.XX.XX.164:4251
created zip0.rar0 [0]USA|0309293 This server was created Thu Dec
28 2006 at 11:29:45 PST
Wed Jan 17 00:29:40 2007 XX.XX.13.91:80 -> XX.XX.XX.164:4251
myinfo zip0.rar0 [0]USA|0309293 zip0.rar0 Unreal3.2
iowghraAsORTVSxNCWqBzvdHtGp
lvhopsmntikrRcaqOALQbSeKVfMGCuzNT
Wed Jan 17 00:29:40 2007 XX.XX.XX.164:4251 -> XX.XX.13.91:80
join None #zip0-s#,#zip0-d1#,#zip0-d2# compress
```

Valuable information found on previous honeysnap output is listed:

-- Bot server operation on port 80, localized on USA, California, using the name: compress.zip0.com.ar (XX.XX.13.91);

-- Server access information: password: <zip0.compresspass>, user: *[0]USA|0309293*, channels: #zip0-s#, #zip0-d1#, #zip0-d1#;

-- Channel topics:

Channel	*Users*	*Topic*
#zip0-d1#	*1646*	*[+smntMCu] adv5c4n napi_139 50 3 0 -r -t -s*
#zip0-d2#	*1646*	*[+smntMCu] adv5c4n napi_445 50 3 0 -r -t -s*
#zip0-s#	*1646*	*[+smntMCu] k3y pay -s*

As soon as the bot connects to the bot server, they start communicating generating some traffic like the one shown above.

Based on all data gathered up to now, it possible to configure an IRC server (in the sandbox) to act like the original bot server in a way that the bot connects to this fake server thinking it is connected to the real server.

Running the botclient on the sandbox it was possible to deduce that the commands were provided through channel topics. The analyzed botnet, named “zip0”, uses the binary executable “h.exe” as a botclient. This malware has been captured on the botnet controller FTP server.

```
* Welcome to the zip0 IRC Network
[0]USA|0309293!eexakewk@201.82.26.240
* Your host is zip0.rar0, running version Unreal3.2
* This server was created Thu Dec 28 2006 at 11:29:45 PST
* zip0.rar0 Unreal3.2 iowghraAsORTVSxNCWqBzvdHtGp
lvhopsmntikrRcaqOALQbSeKVfMGCuzNT
* MAP KNOCK SAFELIST HCN MAXCHANNELS=10 MAXBANS=60
NICKLEN=30 TOPICLEN=307 KICKLEN=307 MAXTARGETS=20
AWAYLEN=307 :are supported by this server
*WALLCHOPS WATCH=128 SILENCE=15 MODES=12
CHANTYPES=# PREFIX=(ohv)@%+
CHANMODES=beqa,kfL,l,psmntirRcOAQKVGCuzNSMT
NETWORK=zip0 CASEMAPPING=ascii EXTBAN=~,cqr :are supported
by this server
* There are 2 users and 1651 invisible on 1 servers
* 3 :channels formed
* I have 1653 clients and 0 servers
* Current Local Users: 1653  Max: 3630
* [0]USA|0309293 sets mode +i [0]USA|0309293
```

The FTP access was possible using access information (user, password and server address) gathered by Sebek and Smart tools. In the server we could found 3 identical executables, but with different names:

```
artefato: h.exe , j.exe , z.exe
md5: b2ef11a82e287e6f0bf3fe57274adf11
```

The observed behavior was: as soon as the malware executes, it installs itself on system using the name "MSSCF32.exe", it deletes the executables "h.exe" and add a registry entry to execute the botclient every time the machine initiated.

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\C
urrentVersion\Run]
"MS System Call Function"="MSSCF32.exe"
```

After a proper installation on system the bot on execution, "MSSCF32.exe" tries to connect to server "compress.zip0.com.ar" as it can be observed from the collected network traffic.

```
00:15:57.601377 IP 192.168.0.55.1038 > 192.168.192.254.53:
52974+ A? compress.zip0.com.ar. (38)
```

Our DNS server on sandbox responds the DNS request as if it was the host "compress.zip0.com.ar", so the bot start the connecting process to our monitoring machine. After connecting to the server, the botclient remains inactive for about 40 minutes, and then starts some scans as can be observed from the Sebek data.

```
[SCAN]: Random Port Scan started on XX.XX.x.x:445 with a
delay of 3 seconds for 0 minutes using 50 threads.
```

The previous command captured by Sebek is related to the channel "zip0-d2" topic: "adv5c4n napi_445 50 3 0 –r –t –s". The scan process can also be analyzed by the tcpdump

```
00:29:42.546634 192.168.0.55.4650 > XX.XX.164.124.445: S
388531280:388531280(0) win 16384 <mss
1460,nop,nop,sackOK> (DF)
00:29:42.604849 192.168.0.55.4653 > XX.XX.104.166.445: S
388684351:388684351(0) win 16384 <mss
1460,nop,nop,sackOK> (DF)
```

captured network traffic.

The relation between channel topic and botclient behavior can be used to understand how botnet controller operates and what are his intentions.

The honeynet is able to monitor every channel topic change, which can be provided to sandbox and the new bot behavior analyzed. This is very important because even if the botnet controller starts to use some kind of encryption, the honeynet analysts will still be able to identify the bot response to the new command.

VII. CONCLUSION

In this paper we presented some of the main botnets features and threats against computers connected to the Internet which has been concerning network security community recently. We showed how a honeynet could be helpful to understand botnet behavior.

Using a honeynet and a sandbox, it was possible to identify how botnet controllers provide commands to their bots, executing operation of their interest. We presented a real example where commands and parameters had been provided by IRC channel topic and their effect on bot analyzed, revealing intruder intentions.

The entire methodology presented was based on free software tools available on the Internet, providing flexibility to researchers that intent to reproduce this work according to their needs.

REFERENCES

[1] Bacher, P and Holz, T. and Kotter, M. and Wicherski, G. (2005) "Know Your Enemy: Tracking Botnets" In The Honeynet Project White Paper http://www.honeynet.org/papers/bots (verified in 03/2007).

[2] Barbato, L. G. C. and Montes, A. (2004) "Smart: Resultados da Monitoração de Atividades Hostis em uma Máquina Preparada para ser Comprometida" I WorkComp Sul - Unisul - Universidade do Sul de Santa Catarina, Florianópolis, maio 2004.

[3] Barford, P. and Yegneswaran, V. (2006) "An Inside Look at Botnets" University Of Wisconsin http://pages.cs.wisc.edu/~pb/botnets_final.pdf (verified in 09/2007)

[4] DD (2007) "dd" Linux/UNIX Command http://www.gnu.org/software

[5] Dupoux, F. and Ladurelle, F. (2007) "partimage" http://www.partimage.org/Main_Page

[6] Evron, G. and Damari, K. and Rathaus, N. (2007) "Web Server Botnets and Hosting Farms as Attack Plataforms" Virus Bulletin, February 2007, http://www.virusbtn.com (verified in 03/2007)

[7] Ianelli, N. and Hackworth, A. (2006) "Botnets as a Vehicle for Online Crime" Procedings of the First International Conference on Forensic Computer Science Investigation (ICoFCS '2006) / Departamento de Polícia Federal (ed.) - Brasília, Brazil, 2006, 124 pp. - ISSN 1980-1114

[8] Livada, C. and Walsh, B. and Lapsley, D. and Strayer, T. (2005) "Using Machine Learning Techniques to Identify Botnet Traffic" Internet Research Department BBN Technologies. http://www.ir.bbn.com/documents/articles/lcn-wns-06.pdf

[9] MIT REVIEW (2006) "Report: U.S. generates more cyber attacks than any other country" Technology Review - published by MIT Associated Press http://www.technologyreview.com/Wire/18405 (verified in 03/2007)

[10] Nazario, J. (2006) "Botnet Tracking: Tools, Techniques, and Lessons Learned" Virus Bulletin 2006 Conference, Montreal, Canada

[11] Ritter, J. (2007) "ngrep - network grep" http://ngrep.sourceforge.net

[12] Sasser (2004) "Worm Sasser LSASS" http://research.eeye.com/html/advisories/published/AD20040501.html

[13] SNORT (2007) "snort" http://www.snort.org (verified in 03/2007)

[14] Sptitzner, L. (2003), "Know Your Enemy", Addison Wesley, 2nd edition.

[15] Symantec (2007) “Norton Ghost” http://www.symantec.com (verified in 03/2007)
[16] TCPDUMP (2007) “tcpdump” http://www.tcpdump.org (verified in 03/2007)
[17] The Honeynet Project (2007) "Sebek" http://www.honeynet.org/tools/Sebek
[18] TOR (2007) “The Onion Router” http://tor.eff.org (verified 03/2007)
[19] UK Honeynet Project (2007) "Honeysnap" http://www.ukhoneynet.org/tools/honeysnap
[20] Vixie, P. and Dagon, D. (2006) "Malware Repository Requirements" Defcon, Las Vegas, September 2006
[21] Yegneswaran, V. and Barford, P. and Ullrich. J. (2003) "Internet intrusions: Global characteristics and prevalence." In Proceedings of ACM SIGMETRICS, San Diego, CA, June 2003.