2011 Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática

Rui Sousa

Análise Comportamental Multi-Nível de *Botnets*

(Multi-Level Analysis of Botnets behaviour)

Rui Sousa

Análise Comportamental Multi-Nível de *Botnets*

(Multi-Level Analysis of Botnets behaviour)

"When you want to succeed as bad
as you want to breathe, then you'll
be successful"

- Eric Thomas

2011 Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática

Rui Sousa

Análise Comportamental Multi-Nível de *Botnets*

(Multi-Level Analysis of Botnets behaviour)

Dissertation submitted to University of Aveiro to fulfill the requirements needed to obtain the Masters degree in *Engenharia de Computadores e Telemática*, held under the supervision of Paulo Salvador and António Nogueira, Assistant Professors at the Department of Electronics, Telecommunications and Informatics of University of Aveiro.

The Jury

President

Rui Luís Andrade Aguiar

Associate Professor at University of Aveiro

Examiners Committee

Paulo Jorge Salvador Serra Ferreira

Assistant Professor at University of Aveiro(Supervisor)

António Manuel Duarte Nogueira

Assistant Professor at University of Aveiro(Co-Supervisor)

Carlos Manuel da Silva Rabadão

Adjunct Professor at the Polytechnical Institute of Leiria

Acknowledgements

To my supervisor, Professor Paulo Salvador, as well as my co-supervisor, Professor António Nogueira, for all the support and motivation.

To Ivo Petiz, a good friend, for helping me along the way.

To my close friends, for the encouragement given and pushing me to be a better person and student.

To my family, for being such an inspiration, throughout my whole life.

And to Silvana, for all the caring and making me want to thrive.

Resumo

Hoje em dia, as redes de computadores têm sido, mais do que nunca, alvo de ataques de segurança. Estes ataques tornaram-se bastante complexos, e com diferentes tipos de motivações. Uma grande parte destes ataques está ligado a *Botnets*.

As *Botnets* podem ser descritas como um grupo de *bots* que executam *software* malicioso autonomamente. Infectam maioritariamente computadores pessoais, e começam a executar tarefas automáticamente, sem o conhecimento dos utilizadores. Os computadores tornam-se então "parte" da Botnet.

Nesta dissertação, são descritos e analisados diferentes tipos de Botnets dedicadas ao envio de spam. Após serem instaladas, o tráfego gerado é capturado, processado e analisado, por forma a identificar características que possam diferenciar cada um dos tipos de Botnets.

São efectuados diferentes níveis de análise, de forma a compreender todos os mecanismos de funcionamento destes tipos de redes.

Abstract

Nowadays, computer networks are, more than ever, major targets of security attacks. These attacks became very complex, and with different kinds of motivations. A major part of the network attacks is linked to Botnets.

Botnets can be described as a group of bots that run malicious software autonomously. They mainly infect personal computers, and start performing automatic tasks, without the awareness of the users. Computers then become “part” of the Botnet.

This dissertation will describe and analyse different types of spam Botnets, by installing them, capturing the generated traffic and characterizing it, in order to identify differentiating characteristics that can be used to detect their activity.

Different levels of analysis are conducted, in order to understand all the functioning mechanisms of these types of networks.

Contents

List of Figures

List of Tables

Nomenclature

AV	Antivirus
Bot	Robot
Botnet	Network of Bots
CC	Command and Control
CERT	Computer Emergency Response Team
DCC	Direct Client-to-Client
DDoS	Distributed Denial of Service Attacks
DNS	Domain Name System
DoS	Denial of Service
FTP	File Transfer Protocol
GPL	General Public License
GT	Global Threat
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IRC	Internet Relay Chat
ISP	Internet Service Provider
mIRC	multiple IRC
MX	Mail Exchange
NBNS	NetBIOS Name Service
NBSS	NetBIOS Session Service
P2P	Peer-to-Peer
RPC	Remote Procedure Call

SIP	Session Initiation Protocol
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOCKS	Sockets
SSH	Secure Shell
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VM	Virtual Machine

Chapter 1

Introduction

In the last decade, technology completely revolutionized our world. Internet became a priority need on peoples lives. What used to be somewhat of a luxury, to be constantly connected to the Internet, is now common and people connect using different types of terminals, like computers, phones or tablets.

With this growth, security dangers started to become more and more important, as happens in many other types of activities. Hackers started developing more complex viruses, taking advantage of the systems flaws.

What used to be a question of personal recognition, got to a point where hackers use Networks of Bots (Botnets) to rent the resources of infected machines in order to obtain higher profits.

Botnets are one of the main problems of the Internet nowadays. Largest Botnets have sizable proportions, and their main objective is to obtain private information from the infected machines, also referred to as robots (bots).

Botnets can then be described as a set of infected machines that remotely controlled by a main entity, called a Botmaster. The Botmaster is responsible for the actions performed by the infected machines, which usually means infecting more machines for the Botnet.

Botnets rely on Command and Control (C&C) servers. This communication is mandatory before remote attacks can be made.

In order to study Botnets, we have to decide which approach to follow. For instance, we can study the C&C servers and their activity, the source code of the Botnet or even analyse its behaviour. This is where the research for this dissertation was focused on. Using a machine with the sole purpose of getting infected, we were able to observe various attacks and realize how they behave.

Major efforts are being made everywhere, some of them successfully, to dismantle the most important Botnets that have been detected around the Internet. However, existing Botnets are continuously evolving and new threats are always appearing.

1.1 Objectives

The main objective of this dissertation is to develop a detection mechanism that can help prevent Botnets infections. In order to achieve this goal, it is necessary to have a complete knowledge of the Botnets activities and behaviours. This knowledge can only be obtained by performing an analysis of the traffic flows generated by the identified Botnets.

So, the main objectives of this dissertation are:

- Characterize the different types of existing Botnets
- Identify the most common behaviours of existing Botnets
- Describe in detail the Botnets that were selected for further study
- Install the selected Botnets
- Analyse the traffic generated by the installed Botnets in the infected machine
- Characterize the behaviour of the studied Botnets

1.2 Motivation

Currently, it is believed that around 25% of all computers worldwide are infected. Despite the efforts that have been made to dismantle all Botnets, the main goal in this area is to dismantle at least the biggest ones, although there are hundreds of them.

The main objective of this dissertation is to study, analyse and understand how spam Botnets work, in order to identify differentiating characteristics that can be used to detect their activity. Hopefully, each spam Botnet can present a characteristic behaviour pattern that can be used to differentiate it from other security threats. However, among the different options that can be used to study Botnets, installing appropriate bots and capturing the generated traffic seems to be the most reliable methodology, as it will be explained latter.

1.3 Dissertation Structure

This dissertation has the following structure:

- Chapter two presents the Botnets state of the art. It will explain the different types of Botnets, their evolution, their common behaviours, as well as the detection techniques that are currently used.
- Chapter three will discuss in detail the different Botnets that were chosen for analysis, explaining why they were chosen and how will be the analysis process.
- In chapter four, we will start by presenting the results taken from the captures of the different Botnets, making also high and low level analysis of the obtained results.
- The last chapter will present the main conclusions about the developed work. Besides, some guidelines for future research work in this subject will also be given, specially regarding possible Botnet detection mechanisms.

Chapter 2

State of the Art

2.1 Introduction

This chapter presents an overview of Botnets, their evolution and classification, as well as their behaviours and the techniques that are used to detect them.

The term Botnet is commonly used to specify a set of automated software robots that execute instructions without human intervention. Their intentions are malicious and endanger the security and safety of the Internet, so it is mandatory to build detection mechanisms that can help disrupt this threat.

The remaining sections of this chapter are organized as follows. Section 2.2, gives an overview of the different types of Botnets, also discussing their evolution. Section 2.3 presents the different Botnets classifications and their common behaviours. Finally, Section 2.4, discusses the main techniques that are used to detect Botnets.

2.2 Overview and evolution

As previously said, Botnets are a recurrent theme nowadays. They have become the biggest threat in the Internet at this moment.

Originally, Botnets were developed as a tool for Internet Relay Chat (IRC) channels. However, due to the vulnerabilities of the clients, exploits started to appear, somewhere around the year 2000. After some time, hackers realized the potential gain they could get by using Botnets.

As Internet users began to trade and carry on banking operations online the nature of malware shifted from disrupting service to exploiting these technologies for financial gain. Malware may be used to steal sensitive information, such as credit card numbers, social security numbers, and passwords. It then sends the harvested information to a botmaster, which may use the information for further attacks or may sell it to other criminals. In turn, these criminals may use the information for nefarious activities including identity theft[1].

And as we will see further ahead, nowadays the primary motivation for operating a Botnet is the income that can be earned from spam mail. Ferris Research[2] claims that email spam costs businesses over €95 milliards per year worldwide.

Another popular source of income for online criminals is the installation of advertising software, known as adware, on victim systems. Many adware software companies offer monetary incentives for installing their software[3]. Phishing schemes are also a major revenue generator for Botnet operators.

Thus, what used to be a question of reputation, soon became a question of financial profits. The money that can be obtained depends on the number of infected machines. Resources from the infected machines can be rented out to the highest bidder, for various purposes. So, it becomes easy to realize that the bigger the Botnet is, the more money is there to be made.

We can claim that Botnets are mainly used for spamming, Denial of Service(DoS) attacks, data theft, as well as other crimes that will be explained later in this dissertation.

Regarding their behaviour, at the beginning of the Botnets activity, infected machines generated massive traffic, but due to the appearance of detection mechanisms, that will be discussed later in this dissertation, they have adapted themselves and became more intelligent. Infected machines started to generate less traffic in order to minimize the probability of being detected.

A general overview of the Botnet propagation process can be seen in the picture below.

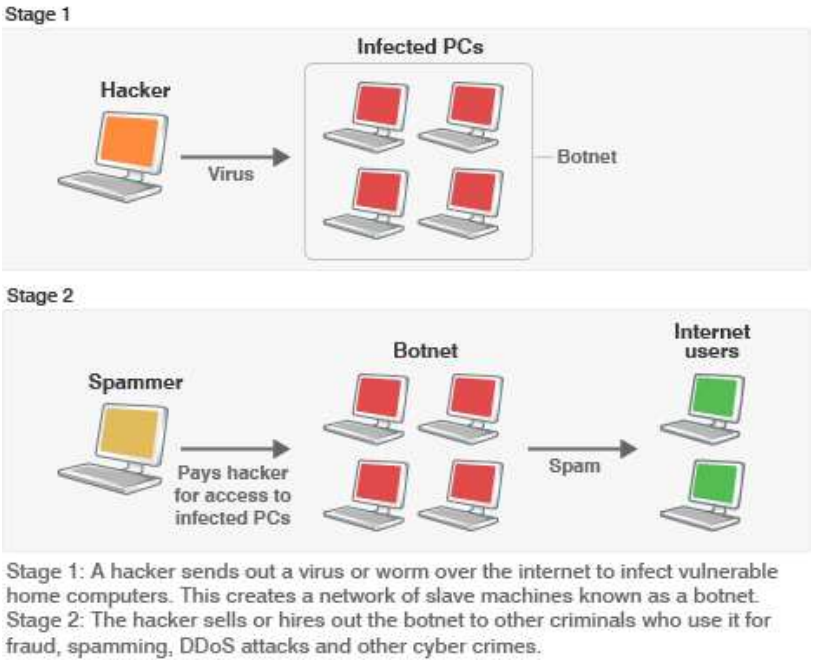

Figure 2.1: Classic example of the propagation of a Botnet[4]

Like it has been pointed out, Botnets send instructions to infected machines, which then send the same instructions to other infected machines, mainly through the IRC protocol. The recipe of a Botnet is usually a server and a client program, and the program that is installed in the infected machines. Usually, these three entities communicate between them and can even encrypt the communication in order to remain undetected or to avoid intrusion into the Botnet control network[5].

Botnets have a major upside, they are very effective performing tasks that would be impossible with a single computer, a single Internet Protocol (IP) address or even a single Internet connection. Botnets were intensively used to perform Distributed Denial of Service Attacks (DDoS) (Figure 2.2), but measures were taken to prevent networks from these kinds of attacks, making them less effective[6].

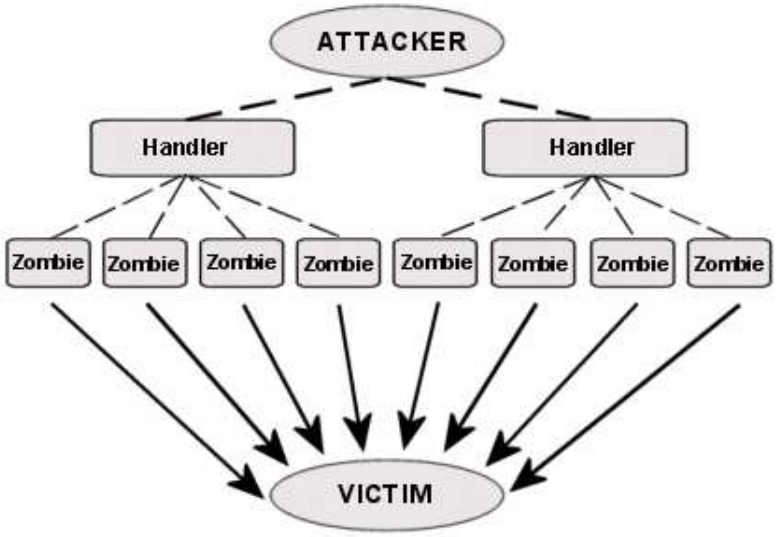

Figure 2.2: Architecture of a common DDoS attack[7]

In order to better understand Botnets, it is important to say something about the dimensions of Botnets nowadays. According to [8], the top Botnets at the end of 2010, regarding the percentage of spam, are Rustock, Grum and Cutwail. Rustock was responsible for 47.5% of spam sent in the Internet. Its size was estimated to be between 1.1 and 1.7 million of infected computers. The main infected country was the United States of America. However, in March 2011, in a joint task force between Pfizer, the University of Washington, FireEye, several Internet Service Providers(ISP's) and Computer Emergency Response Teams (CERT) around the world, Microsoft was able to take Rustock down. Grum and Cutwail numbers are relatively small, when compared to Rustock, with 8.5% and 6.3% respectively. Their sizes were estimated to be between 310 and 470 thousand infections, regarding Grum, and between 560 to 840 thousand infections for Cutwail. The main infected countries are Russia and India, respectively.

These top three Botnets were responsible for a total 57.9 milliards of spam mails per day.

It was expected, that with the take down of Rustock, these numbers would go down drastically, considering its size. Even acording to [9], despite the top Botnet(Cutwail) was only responsible for 16.1% of the spam reported, and with an estimated size from 800 thousand to 1.2 millions infected machines, constrasting Rustock's 47.5% and 1.1 million to 1.7 million users, the numbers did not change as expected. In detail, the overall percentage of spam only dropped from 77% to 76.6%, however, the number of spam mails per day went from approximately 71 milliards to 45 milliards. It is important to mention that on the 2010 report, from the grand total of 71 milliards spam mails per day, Rustock was responsible for 45 milliards.

It is also important to mention that the main operating system targeted by Botnets is Microsoft Windows, mainly because it is the most popular.

2.3 Botnets Classification and Behaviours

Botnets exhibit some characteristics in their communication mechanisms with the control server that gives us the opportunity to distinguish them, simply by analysing their behaviour.

However, Botnets present a general common behaviour that is shown in the following picture.

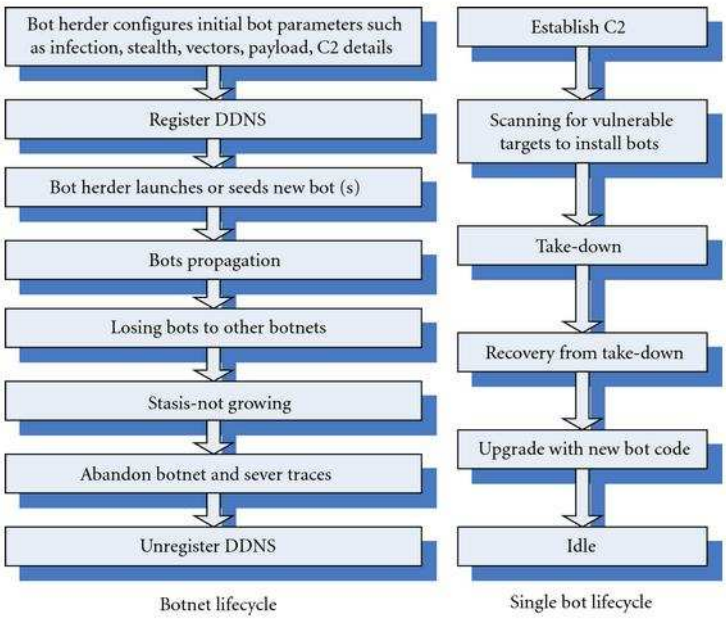

Figure 2.3: Botnet and Bot Life cycle[10]

There has been an evolution in the network structure topologies that are used, like the resort to P2P technologies. So, Botnets characterization should also consider the generated network traffic and their inherent topologies[11].

In [12], the different types of bots were identified and listed. Next table presents that list.

Types	Features
Agobot PhatBot Forbot Xtrembot	They are so prevalent that over 500 variants exist in the Internet today. Agobot is the only bot that can use other control protocols besides IRC[13]. It offers various approaches to hide bots on the compromised hosts, including NTFS Alternate Data Stream, Polymorphic Encryptor Engine and Antivirus Killer[14].
SDBot RBot UrBot UrXBot	SDBot is the basis of the other three bots and probably many more[13]. Different from Agobot, its code is unclear and only has limited functions. Even so, this group of bots is still widely used in the Internet[14].
SpyBot NetBIOS Kuang Netdevil KaZaa	There are hundreds of variants of SpyBot nowadays [15]. Most of their C2 frameworks appear to be shared with or evolved from SDBot [15]. But it does not provide accountability or conceal their malicious purpose in codebase[15].
mIRC-based GT-Bots	GT (Global Threat) bot is mIRC-based bot. It enables a mIRC chat-client based on a set of binaries (mainly DLLs) and scripts[14]. It often hides the application window in compromised hosts to make mIRC invisible to the user[13].
DSNX Bots	The DSNX (Data Spy Network X) bot has a convenient plug-in interface for adding a new function[14]. Albeit the default version does not meet the requirement of spreaders, plugins can help to address this problem [13].
Q8 Bots	It is designed for Unix/Linux OS with the common features of a bot, such as dynamic HTTP updating, various execution of arbitrary commands and so forth[13].
Kaiten	It is quite similar to Q8 Bots due to the same runtime environment and lacking of spreader as well. Kaiten has an easy remote shell, thus it is convenient to check further vulnerabilities via IRC[13].
Perl-based bots	Many variants written in Perl nowadays[13]. They are so small that only have a few hundred lines of the bots code[13]. Thus, limited fundamental commands are available for attacks, especially for DDoS-attacks in Unix-based systems[3].

Table 2.1: Types of bots

Considering the previous table, it is important to detail some of the most typical bots.

2.3.1 Most common bots

2.3.1.1 Agobot

This particular bot was programmed in C/C++[13]. According to [13], it is so far the only bot that resorts to a control protocol that uses the IRC channel[13]. Because of its implementation, attackers can very easily adapt Agobot to their purposes, due to the possibility of adding new functions in the related classes[13]. There are also various IRC commands that can be used to collect sensitive information[15]. As an example, a bot can be instructed to do some operations[15]. Besides that, Agobot is also able to secure the system, through the closure of NetBIOS shares[15]. Agobot has several commands to control the infected machine, which can be related for example to managing all the processes or managing autostart programs[15]. Agobot also has various features, as it has been pointed out in [12, 15]:

- It is IRC-based C/C++ framework,
- It can launch DoS attacks,
- It can attack a large number of targets,
- It allows the collection of sensitive information through traffic sniffing,
- Has the possibility to avoid detection by Antivirus (AV) software, by fixing vulnerabilities or even closing access to AV websites[13],
- Can detect virtual machines and avoid disassembly[13, 15].

To look for a new possible victim, Agobot just needs to scan a network range[15]. On the disadvantages side, it is not able to distribute targets among a group of bots as a whole effectively, due to its command set limitations[12, 15].

2.3.1.2 SDBot

SDBot's source code is small, around 2500 lines, but its command set resembles Agobot[13, 15] and it is published under General Public License (GPL)[13, 15]. Even though this bot does not have the capability to propagate and has only a few functions, it is still appealing due to their ability to implement new commands[15]. SDBot has, however, some very own IRC functions, such as spying and cloning[11]. The spying method basically records the activities of a channel on a log file[15]. The cloning method refers to the ability of the bot to repeat a connection to one channel[15]. In [13], it is believed that this bot may be the most active bot around the world.

This bot relies on an IRC implementation[15]. In order to establish contact with the IRC server, it sends identity information, and if it gets a PING message, it will acknowledge it with a PONG[15]. Assuming the connection has been established, it is possible to request a hostname by using the USERHOST message, and after that, join a channel resorting to the JOIN message[15]. Once the response code has been received, the Botmaster can control it through IRC commands[15].

SDBot has the ability to target new victims easily, due to its scanning tools[15]. As an example presented on [15], by using the NetBIOS scanner it can select a random target that is in an IP range[15]. The SDBot is capable to send Internet Control Message Protocol(ICMP) and User Datagram Protocol(UDP) packets, which can be used for flooding attacks [15].

2.3.1.3 SpyBot

SpyBot is a bot written in C, relatively small as well, with nearly 3000 lines, that is widely spread, with various versions nowadays[15]. In [15], the authors consider SpyBot an enhanced version of the SDBot. It has, like SDBot, the scanning ability, host controlling functions, as well as modules for DDoS attacks and flooding attacks[15]. The host controlling capabilities are very similar to those of Agobot's[15]. However, SpyBot does not provide the same possibilities as Agobot, which still make it a less used bot[15].

2.3.1.4 GT Bot

This last bot, Global Threat (GT) Bot, also known as Aristotles, supposedly stands for all mIRC-based bots that have several variants and are mainly used for the Windows operating system[13, 15]. It has some more general capabilities like IRC host control, DoS attacks, port scanning, or even NetBIOS/Remote Procedure Call (RPC) exploiting. GT Bot also has a few set of binaries and scripts for mIRC[13, 15]. It is important to refer the binary HideWindow program, which allows the mIRC instance to remain invisible from the user[13, 15]. The binaries are usually named as "mIRC.exe", but they can have different capabilities[15]. When compared to other bots, GT Bot has a limited set of commands for host control, only capable of getting local system information and running or deleting local files[15].

Botnets can also be classified according to the protocol that is used for their command and control operations: Hypertext Transfer Protocol(HTTP), Peer-to-Peer(P2P) or IRC. Although this dissertation is mainly focused on HTTP Botnets, more specifically on the ones dedicated to spamming, all of them will be presented and discussed in the next subsections.

2.3.2 Types of Botnets

2.3.2.1 IRC Botnets

IRC is a protocol used by Internet users for instant messaging. It is based on a Client/Server (C/S) model, but it is also suited for distributed environments[16]. Usually, IRC severs are interconnected and exchange messages between them[16]. It is possible that one machine connects with hundreds of clients through multiple servers. The multiple IRC (mIRC) is a more complex communication context, where communications between the clients and the server are based on several specific channels to which clients are connected. The available functions of IRC based bots include managing access lists, moving files, sharing clients, or even sharing channel information[16].

A classic example of a typical IRC Botnet can be seen in the following figure.

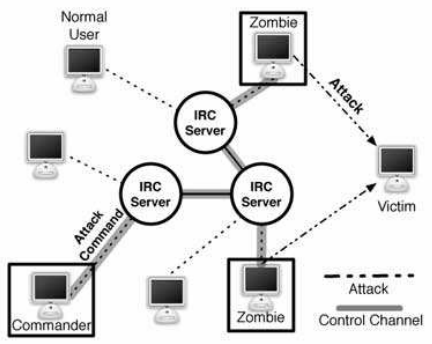

Figure 2.4: IRC Botnet propagation[17]

IRC Botnets were the first to appear, but they lost some relevance over the last years. However, they still exist and IRC is still the most common C&C mechanism used by bots. According to Microsoft, in the second quarter of 2010, the number of computers that reported Botnets detection follow the distribution that is shown in the next figure.

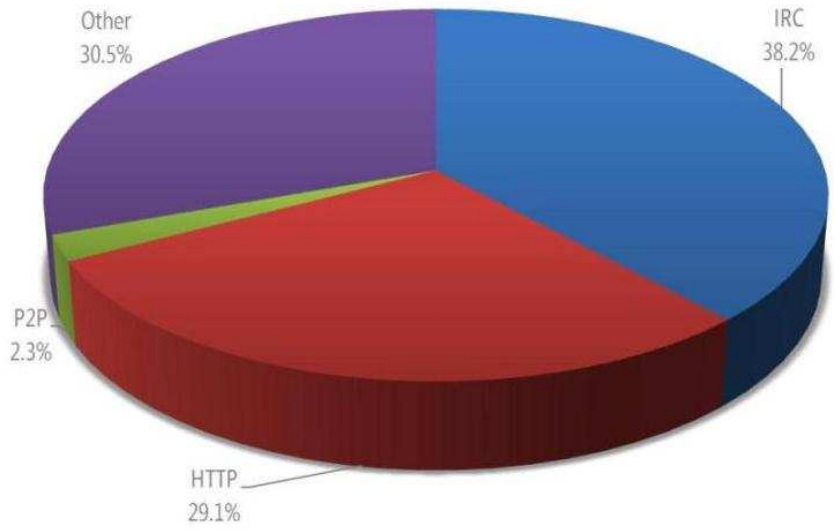

Figure 2.5: C&C mechanisms used by Botnet families according to the number of unique computers that report detections[18]

Machines infected by IRC Botnets are instructed to connect to an IRC server and a channel, where they wait for further instructions, usually in the form of personalized text messages. Obviously, some have become more sophisticated, to prevent detection, encrypting the bot commands in the channel topic. This can be quite complex, as they can instruct different subsets of bots to do different tasks. The criteria that is usually used relies on the Country, network location, the uptime of the bot, bandwidth available, among others.

On [14], the four stages of the attacker's operations are described, as follows :

1. The Creation Stage, where the hacker can add malicious code or modify an already existing one of the many configurable bots over the Internet[14].

2. The Configuration Stage is where the IRC server and channel information can be collected[14]. While the bot is in the victim's computer, it connects automatically to the selected host[14]. Then, the hacker can restrict the access and secure the channel to the bots for business or some other purpose[14]. As an example, the hacker can list its bots to authorized users, who in turn, can customize the bots as they like and use them for their own purpose.

3. In the Infection Stage, bots are propagated by various direct and indirect means[14]. Direct techniques take advantage of vulnerabilities on services or operating systems and are usually associated with the use of viruses[14]. While the systems are compromised, they keep running the infection process, saving the time of the hacker to add new victims[14]. The most vulnerable system is Microsoft Windows, more specifically Windows 2000 and XP, since the attacker can track, without much effort, unpatched or unsecured users[14]. Indirect approaches resort to the use of other programs as a proxy to spread the bots, or in other words, they use Direct Client-to-Client (DCC) file exchange to distribute malware on IRC networks to vulnerable hosts[14].

4. The last operation, the Control Stage, is where the attacker is able to send instructions to its bots through the IRC channel, usually to perform some malicious tasks.

2.3.2.2 P2P Botnets

P2P Botnets started somewhere around the year 2003. They are more popular now, but they are not the favourite mechanism used by bots. Mainly, they were created to avoid being shut down, like happened to other types of Botnets. Some P2P Botnets use mechanisms that derive from open source P2P implementations (e.g. Kademlia), while others (e.g. Waledac), have their own implementation.

A typical P2P Botnet can be observed in the next figure.

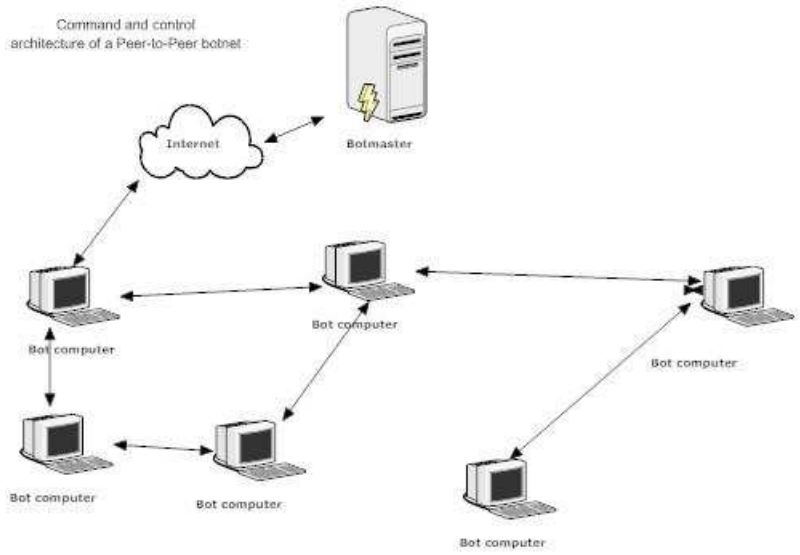

Figure 2.6: P2P Botnet propagation[19]

P2P Botnets are still hard to study, since there is much less information from these Botnets when compared to HTTP or IRC Botnets. The first worm related to P2P was Slapper[20], which infected the Linux operating system back in the year of 2002, resorting to DoS attacks. Using hypothetical clients, this worm sent commands to compromised machines, in order to receive responses from them[20]. Due to this, its location on the network could remain anonymous, so it was very hard to monitor[20]. In 2003, another P2P-based bot, Dubbed Sinit, was launched[21]. This bot used public key cryptography to update authentication. PhatBot, which has already been mentioned, appeared in 2004. Using a P2P System, it sent commands to other infected computers.

Storm Worm is probably, according to[22], the most spread P2P bot on the Internet. On [23], the authors analysed the bot with different detection mechanisms, and developed some techniques to disrupt P2P-based Botnets communications, like polluting the file or eclipsing contents.

P2P-based bots are not a major concern, since they are not well developed yet and still have many fragilities. Hybrid or Mixed P2P networks, the ones that have a central server or a list of peers that can be contacted to add a new peer, still have a single point of failure for this type of Botnets. However, in [24], an hybrid P2P Botnet was presented with the ability to overcome this fault.

This architecture allows an attacker to inject commands to any host of the Botnet. The hosts connect periodically to their neighbours in order to collect orders that have been given by his commander. When a command is issued, the host will forward it to nearby bots. The features of this architecture are described as follows[24],

1. It does not require a bootstrap procedure
2. Only some bots (near the captured one) can be exposed
3. Ease of management of the entire Botnet with a single command

Even though in [24] the authors propose various countermeasures to prevent this type of Botnet attack, this architecture still needs further research, as well as new prevention methods[10].

2.3.2.3 HTTP Botnets

HTTP Botnets are the perfect example of the Botnets evolution. IRC Botnets started being detected and shut down, so hackers felt the urge to adapt and evolve. Now, IRC Botnets are much harder to detect, because they are easily camouflaged under the HTTP protocol.

The following figure shows an example of an HTTP Botnet and its propagation procedure.

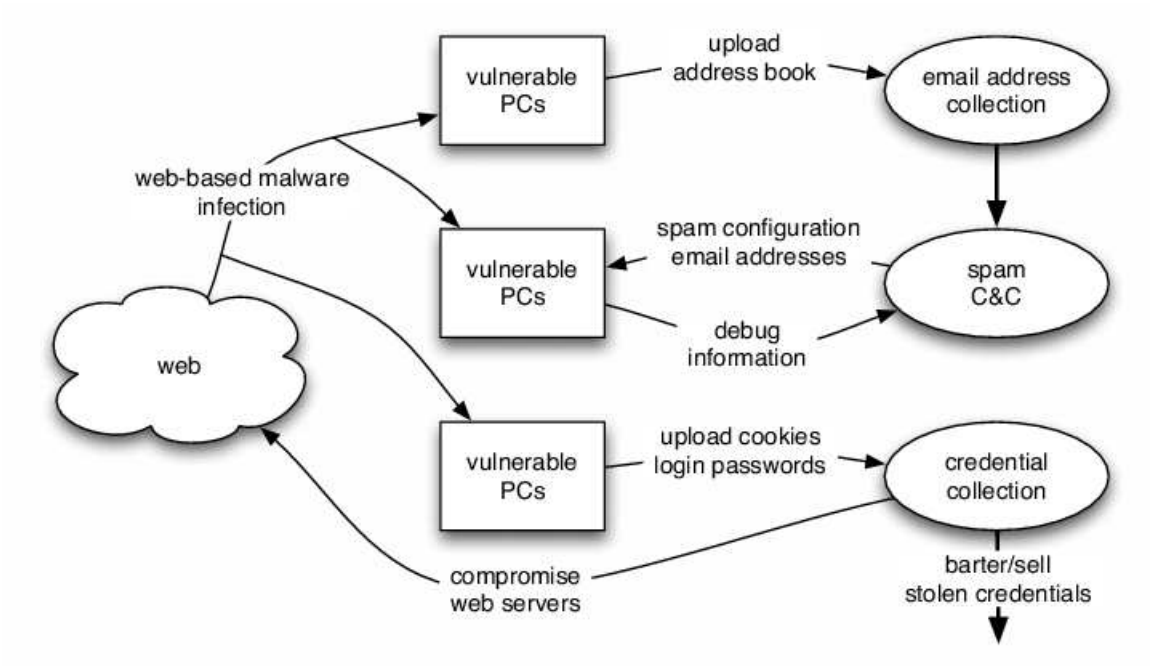

Figure 2.7: HTTP Botnet propagation[25]

HTTP Botnets are mainly dedicated to sending spam mail and data theft. The biggest Botnets dedicated to data theft are Zeus and SpyEye.

HTTP Botnets are a major threat for the Internet. The volume of spam mail has reached astronomic values, which endanger the Internet worldwide.

As previously discussed, hackers rent out their bots to third parties, but they also use it for themselves, to infect more machines, thus increasing the size of their Botnet.

It is important to mention that every infected machine is being monitored by the Botmaster. The Botnet is usually partitioned into subsections, so they can be controlled separately. This allows the bot owner to select which parts to rent or to prevent access to valuable parts of the Botnet without his permission. After a negotiation between the bot owner and the third party, the bot owner instructs machines to start a proxy server, and usually provides access to a webpage that has the IP addresses and ports of the bots that are part of the deal. These bots then become "property" of the spammer. And an unaware user may be in the Botnet indeterminately, since every time there is a change in the IP address or in the proxy server port, the infected machine informs its C&C server, which in turn alerts the spammer with the updates.

The biggest HTTP Botnets dedicated to spam are, according to [8], Rustock, Grum, and Cutwail. Rustock was shut down in March, so it was not used in this dissertation.

As it has been told before, hackers have several malicious intents. Mainly, Botnets are used for either financial and destruction purposes[26]. There are several types of common attacks, which include DDoS attacks, spamming, sniffing traffic, spreading new malware, installing advertisement Add-ons and Browser Helper Objects or even attacking IRC Chat Networks[12].

A brief explanation on each one of these subjects is given in the next paragraphs.

DDoS network attacks cause loss of service to users, which usually means losing network connectivity and services, consuming the available network bandwidth or even overloading the computational resources of the users systems[12].

Spamming relates to bots that have the capabilities to open Sockets(SOCKS) proxies on the infected machines. After a proxy is enabled, usually the machine starts relaying spam or phishing email[12].

Sniffing traffic is the ability to use a packet sniffer in order to observe data from an infected machine. It is usually used to steal private information, like usernames and passwords[12].

Spreading new malware refers to Botnets that are used to launch new bots and/or malware. This can be easily done, as all bots implement download mechanisms and execute files using HTTP or File Transfer Protocol (FTP). Some bots can even behave as servers for malware[12].

Installing advertisement Add-ons and Browser Helper Objects means the assembly of a malicious website, with some advertisements, signing up contracts with companies that offer money for clicks on advertisements. In this way, the creator of the website can get some income. This method can be automated, where the clicks are performed by several bots clicking on the advertisements[12]. It can be easily understood that these actions can provide significant financial profits. This is clearly a classic example used for financial interests.

Finally, attacks on IRC Chat Networks are the ones where the network of the infected machine is flooded by service requests from several bots or by several channel-join requests issued by various bots. This usually causes the network to go down, which is very similar to a DDoS attack[12].

In order to end this section, I think it is important to mention some numbers that can give a real idea of the problem that we have on our hands. The average global spam percentage was equal to 89.1% in the year of 2010, where 88.2% was generated by Botnets[8]. After Rustock take down, the numbers started to drop, and in June 2011, the global spam rate percentage was equal to 72.9%. Botnets were responsible for 76.6% of that spam[9].

As it will be better explained in the next section, it is urgent to develop better detection methods and reliable ways to bring Botnets down. Failing to do so will make the task of controlling the attacks that were explained before harder to achieve, due to their growing potential.

We already know the most common attacks that are performed by Botnets, so it is now time to better understand them, see how they behave, analyse them, try to find patterns and start to create effective techniques to shut Botnets down, in order to prevent further harm.

2.4 Detection Mechanisms

Since Botnets are playing a key role nowadays, their detection mechanisms are increasing in number and quality.

Reference [12] states that due to the escalation of the Botnets, computer security experts started to develop ways to detect and monitor their behaviour, in order to gather information that can be useful in future research. This tracking approach gives researchers the possibility to observe directly the malicious activity on the Internet. It also gives researchers insight into the hacker profile and motivations. It is believed that this research can mitigate the effects or even disrupt Botnets.

However, they also point out that this is not an easy task. These attacks can be very complex and can remain hidden until a certain criteria is met.

According to Matt Sergeant, senior anti-spam technologist for Message Labs Ltd., even a regular internet user can try to detect if his computer is infected. He defends that Botnets generate much more Domain Name System(DNS) queries than normal, so using a tool like Wireshark, this can be easily investigated. He also points out that it is important to look out for Mail Exchange (MX) lookups, as well as .ru, .cn and .info lookups. This usually means that there is an attempt to establish a connection between the bot and the C&C server. To conclude, this specialist claims that there is also a possibility to investigate the unusual volumes of traffic in a network, which usually are generated by the instructions given by the Botmaster.

There are however, more sophisticated detection mechanisms. In [1], some Botnet detection mechanisms are explored. These approaches can be classified as

- Host-based procedures
- Network-based procedures

2.4.1 Host-based procedures

Host-based procedures rely on the detection of possible anomalies on the file system.

One of the methods used to detect these anomalies is AV software. Basically, each malicious software running on the system has a signature, and this software can be used to detect it.

This same signature is, however, the weak point of the method, because hackers can modify the source code of the signature that is allocated to the bot. There is even the possibility that the bot does not have a signature at all, so the AV will not be able to detect it. So, it can be concluded that this method is not very effective.

Another approach that must be considered is the dynamic analysis of unknown software. This refers to the analysis of the behaviour of the software. On the disadvantages side of this approach, we can mention that the software needs to be installed on every system, which generates some overhead and makes the system degrade in performance.

Finally, it is important to mention the method that detects modifications in the Registry of Microsoft Windows. By analysing the malware binaries, it is possible to obtain relevant data regarding IRC, such as the username, the channel, the DNS or even the IP addresses.

All these are reactive approaches, as they can only be treated once the computer has been infected.

2.4.2 Network-based procedures

Network-based procedures focus on analysing the network traffic, based on detected anomalies, to perceive if it is caused by a Botnet.

The analysis can be performed online, as the traffic is generated, or the traffic can be stored in order to be analysed later. One of the procedures mentioned is the Vertical correlation, that focus the detection on individual or single bot infections. The software checks the network traffic and compares it with precast patterns between communications from the infected computer and the C&C server to see if there is a similarity, which would mean that the computer is infected. This approach, however, has the same limitations as the previously mentioned signature method, because the Botnet traffic cannot be detected.

Another procedure available is the Horizontal Correlation, which tries do detect some infected computers in the network. Network traffic is analysed, searching for similarities that can be, for example, the same C&C server. The disadvantage of this approach is that different bots in the same network don't share any relation between them, so they remain unnoticed in the network.

There is also an Anomaly detection procedure. This procedure tries to detect anomalies in the network traffic when compared to common traffic. In [27], Binkley and Singh state that "one infected host that performs a network scan is not an anomaly. However, if there are many hosts performing a network scan and they are in the same IRC channel, this phenomenon is an abnormality compared to the common network traffic".

Another comparison that can be made is to check the ratio between the number of e-mails sent and received. If this number shows that there are more e-mails sent, it usually means that there is a spambot infection taking place.

A procedure that can also be made, specifically for IRC Botnets, is to measure the IRC response time. As mentioned in [28], a human can't respond as quickly as malicious software. So, by comparing response times it is possible to detect the presence of malicious software.

Chapter 3

Study of Botnets

As it has been perceived by now, there is a lot of work that can still be done. There are 3 major types of Botnets, and thousands of variants inside them. This work could be directed to many different ways, but it will be focused on HTTP Botnets, more specifically on the variants dedicated to sending spam. So, in order to analyse them, some research is needed in order to find the most dangerous spam Botnets. After some research, we found that the top 10 Botnets of the year 2010 are described in [8]. So, according to this report, the top 10 Botnets of 2010, in terms of spam, spam per day, size and main countries of infection are the ones presented in the following Table.

Botnet name	**% of Spam**	**Spam/ day**	**Estimated Botnet size**	**Country of infection**
Rustock	47.5%	44.1 Milliards	1.1M to 1.7M	USA(17%), Brazil(7%)
Grum	8.5%	7.9 Milliards	310k to 470k	Russia(12%), India(8%)
Cutwail	6.3%	5.9 Milliards	560k to 840k	India(17%), Russia(16%)
Maazben	5.2%	4.8 Milliards	510k to 770k	Russia(11%), India(10%)
Mega-D	2.3%	2.1 Milliards	80k to 120k	Russia(15%), Ukraine(14%)
Cimbot	2.1%	1.9 Milliards	32k to 48k	Italy(27%), Spain(25%)
Bobax	1.2%	1.1 Milliard	250k to 370k	India(32%), Russia(25%)
Xarvester	0.5%	501 Million	17k to 25k	Italy(15%), UK(10%)
Festi	0.1%	96 Million	8k to 12k	Vietnam(24%), Indonesia(21%)
Gheg	0.1%	49.8 Million	8k to 12k	Spain(12%), Indonesia(10%)
Total Botnet Spam	**77%**	**±71 Milliards**	**3.5M to 5.4M**	**India(9%), Russia(9%)**

Table 3.1: Top Botnets of 2010

The Rustock Botnet was immediately discarded because it was already dismantled. Three of the chosen Botnets are from the top Botnets of 2010. The most difficult task was to find malware that could be installed on a machine in order to infect it. After some research, appropriate malware was discovered for various Botnets, which at the beginning were Grum, Cutwail and Bobax. After some time, It was decided to also install malware from the Lethic Botnet. Even though Lethic is not in the top Botnets of 2010, it still has a significant number of infected machines.

On June 2011, a new table with the top Botnets was released, confirming what was expected. Lethic, that did not appear on the top Botnets of 2010, jumped to number four in six months. The updated table can be seen below, according to [9].

Botnet name	% of Spam	Spam/ day	Estimated Botnet size	Country of infection
Cutwail	16.1%	9.6 Milliards	800k to 1.2M	India (10%), Russia (9%)
Xarvester	6.7%	4.0 Milliards	57k to 86k	United Kingdom (18%), France (13%)
Maazben	3.1%	1.9 Milliards	520k to 780k	Republic of Korea (14%), Russia (10%)
Lethic	3.1%	1.8 Milliards	230k to 340k	Republic of Korea (25%), Russia (15%)
Grum	3.0%	1.8 Milliards	200k to 290k	Russia (14%), India (14%)
Bagle	2.7%	1.6 Milliards	140k to 200k	India (15%), Argentina (8%)
Fivetoone	2.3%	1.4 Milliards	94k to 140k	Vietnam (20%), Brazil (12%)
Festi	1.2%	691 Millions	25k to 37k	India (10%), Vietnam (10%)
Bobax	0.4%	254 Millions	80k to 120k	Ukraine (27%), India (18%)
DarkMailer	0.5%	43 Millions	1k to 1.5k	France (27%), USA (16%)
Total Botnet Spam	**76.6%**	**±45 Milliards**		

Table 3.2: Top Botnets of June 2011

The difference between both tables, in only six months, is clear, reinforcing the idea that Botnets are constantly changing. The Botnets used in this dissertation are all in the top 10, according to June's report. Cutwail gained a lot of importance in the earlier months of this year, but none of the actual Botnets accomplish even half of what Rustock has accomplished. Meanwhile, it was expected to see a decline on the percentage of Botnet spam on the Internet, but the difference is only of 0.4%. What was more noticeable was the reduction of spam per day, which dropped from 71 Milliards to 45 Milliards.

Finally, a trojan was also installed and analysed. The trojan name is Kazy and it will help understand better the activity of a bot.

So, four Botnets were considered in this dissertation, as shown in the following Table.

Botnet name	Aliases
Grum	Tedroo
Cutwail	Pandex/Pushdo
Bobax	Kraken/Oderoor
Lethic	N/A

Table 3.3: Botnets analysed

Before going into the details of each Botnet, is it important to present some details about the Kazy trojan.

3.1 Trojan details

3.1.1 Kazy

The following sentence can be used to compare a trojan to a bot: "A bot is also known as a remote-access Trojan horse program"[29]. This specific trojan, Kazy, downloads malicious files from a remote server, installs and executes them.

Kazy is a backdoor trojan, well known for attacking online bank-related information[30]. This trojan can be distributed in many ways, such as malicious scripts on dubious sites, like advertisements, or as part of an installation package[30]. This trojan is usually included in the form of a link to an .exe

file. When installing itself, it usually resorts to a fake .gif format indicator and the name "iexplorer.exe", which is slightly different from the Internet Explorer process "(iexplore.exe")[30]. The major problems of this trojan that have been reported are the deletion of important files from anti-malware, anti-virus or anti-spyware programs. This makes it very hard to remove this trojan, turning the security of the infected machine into a very fragile element. Another reported problem is browser redirection to phishing sites. This Kazy behaviour is very well known, consisting usually in the use of phishing scams that are presented in the form of fake online bank login pages. However, these expedients can be easily detected by looking into the web address[30]. According to [30], Brazil based banks are privileged targets of the Kazy phishing attempts.

A couple of known aliases for Kazy are Trojan.Win32.Pakes.oya, Trojan.Fakealert.20587, Mal/FakeAV-IK, Generic22.YJ and Win32/Kryptik.MLF[30].

3.2 Botnets details

In order to better understand these Botnets, it is important to give a brief background on each of them.

3.2.1 Grum

Grum, also known as Tedroo, is an HTTP Spam Botnet, which mainly focus its spam on pharmaceutical products. It usually infects files referenced by the auto-run registries. This Botnet is able to hide component files as well as legitimate windows system files, making its detection and removal quite difficult[31]. It has five key features[31]:

- A Kernel-based root-kit
- Reports to a C&C server via HTTP on port 80
- Downloads plain text spam templates and address lists from a web-server
- Has multiple control servers
- Performs DNS MX lookups to send spam.

The behaviour of this Botnet has already been analysed, so it is known that this Botnet will try to establish a control server connection, using an email message, by sending an HTTP request message. Depending on the variant of the Botnet, Grum makes changes in the System Registry[31].

3.2.2 Cutwail

Cutwail, also referred as Pandex or Pushdo, among other names, is also an HTTP Spam Botnet. It has been working since 2007[32]. It focuses mainly in sending spam promoting pharmaceuticals, designer rip-offs or even software. It also distributes malware regularly, sending attachments in emails, usually a .zip file. It usually resorts to celebrity names to deceive users. Nowadays, these Botnets also send malicious campaigns, using social networking brands. They also distribute phishing emails, mainly targeting customers of several financial institutions[32]. The main Cutwail's features are[32]:

- Reports to a C&C server on port 80, resorting to encrypted HTTP
- Performs DNS lookups to send spam and uses templates.

In [32], the Cutwail behaviour is described. Bots connect to the control server using HTTP, through port 80, resorting to an encrypted tunnel and listen on a random UDP port for commands from the control server. The host is capable of downloading malware and, after installing it, it creates different processes[32], mainly with the purpose of notifying the Botmaster and running its commands.

3.2.3 Bobax

Bobax, which can also be found under the name of Kraken or Oderoor, is another HTTP Spam Botnet. Reportedly, it has been working since 2007 and had its peak in 2008[33]. During the last semester of this year, it was responsible for 5-10% of all generated spam. It started attracting a lot of attention, which lead to the disruption of its control servers in the end of 2008[33]. Bobax is still around as one of the top Botnets of 2010, but it is responsible for only 1.2% of spam nowadays, which places it in the 7th place. The main Bobax features are[33]:

- Reports to control server on UDP, through port 447
- Uses dynamic domain name providers for domains
- Performs DNS MX lookups to send spam
- Has multiple recipients per message
- Uses templates
- Has backdoor capabilities.

Bobax starts by checking for a Simple Mail Transfer Protocol(SMTP) connection to a server site, through port 25. Then it generates a pseudo-random domain name, and if the DNS query fails, it will append the domain name on the local network of the infected machine to perform a new DNS query. Once it successfully finds the C&C server, it sends an HTTP request[33]. Like in Cutwail, it creates processes to execute on Windows start-up, and hides its malware registering itself as a random service name. It also has the capability of searching for potential email addresses. After this process, it receives a template from the server to send to its targets.

3.2.4 Lethic

Lethic is an HTTP Spam Botnet that is suspected to exist for quite some time already. It is a proxy type spambot, that relays spam from a control server to its destination[34]. It mainly sends pharmaceutical and replica watches spam emails. Even though it is not present on the top Botnets of 2010[8], at the beginning of 2010 Lethic was responsible for about 8-10% of total spam[34]. However, this is due to the fact that Lethic has been dismantled somewhere near January 2010. Meanwhile, Lethic was revived in February of 2010.

In [34], its main features are described as follows:

- Acts as a proxy relay spam
- Process injection to Explorer.exe
- Fast, multi-threaded
- Anti-debugging and Anti-Virtual Machine detection.

After the infection, the compromised machine connects to different domains. The communication protocol is customised. When connected, the control server initiates the handshaking process and gives the bot an IP address and port to forward the data to, hence making it work as a proxy[33]. An the host, it starts by installing the malware on the Windows System directory. As happened in all other Botnets, it creates a registry entry to execute files on Windows start-up. Lethic also has the capability to inject its code into explorer.exe, and create random processes in the infected machine[33].

3.3 Analysis Methodology

The main objective of this dissertation is to analyse different Botnets by looking at the statistics of the generated traffic. This is a passive approach. We have connected a machine to the Internet with the only purpose of getting infected. Like it has explained before, some bots have the ability to detect Virtual Machines(VM). Thus, in order to get accurate results, no VM's were used in any case. The used machine was always formatted before each manipulated infection, in order to prevent interference between the traffic generated by different Botnets.

The operating system used was always Microsoft Windows XP Service Pack 2, since it is the most targeted operating system.

At the very beginning, a capture of one hour was made in order to observe the "typical" traffic generated by the computer immediately after being formatted. After this first step, we searched for different malware from the different Botnets. In [35], there is a fairly good database of malware. The next step was to set the captures to last for 48 hours. This time was chosen because it gives the possibility to better infer the behaviour of the Botnet and observe their patterns along a longer time line.

No other task was being performed on the infected computer while it was capturing, in order to reduce other generated traffic besides the Botnet traffic. So, apart from the traffic generated by the Botnets, there was very little traffic being generated. Obviously, any other traffic was discarded in the statistics calculated in Chapter 4.

At least two captures for each Botnet were made, in order to observe any differences that can occur in their behaviours.

Chapter 4

Experimental results

This chapter will analyse the traffic that was captured for the different Botnets selected in this dissertation. Traffic was stored in order to be analysed a posteriori. The conducted analysis is a multi-level one, looking at high and low level details. Regarding the high level analysis, the following statistics of the captured traffic were analysed:

- Protocols details
- Packets per hour and per minute
- Amount of traffic transferred per hour and per minute
- Unique peers per hour
- Transmission Control Protocol(TCP) Session Establishment attempts
- Geographical location of the contacted peers.

The protocols details will help understand some of the Botnet activities and objectives by looking at the protocols that are used in the different communications. Using the captures that were made, and creating a script for filtering them by Protocol, it was possible to get the different amount of packets over time per communication protocol. It is important to refer that in the following analysis the references to TCP (unknown) and UDP (unknown) refer to non identified packets and should not be confused with the sum of protocols from each layer. So, the selected filter was able to analyse the packet header and return the Protocol to which it belongs to.

The packets per hour and per minute allow us to verify the behavioural pattern of the Botnet over time. This was accomplished by creating a script that filtered the amount of packets generated per hour and per minute, plotting them conveniently. In order to better illustrate the Packets per minute statistic, a sample of the first two hours was also made by simply filtering the packets corresponding to the first 120 minutes.

The amount of traffic transferred per hour and per minute follow the same procedure of the previous statistic. The script was similar, and the only change that was made to the filter was using the amount of generated data instead of the amount of packets.

The unique peers will help us realize how spread and large the Botnet is. In order to observe this, another script was made, filtering the IP addresses per hour from the stored captures and plotting

them over time. In this procedure, it was decided to report only the number of peers per hour and distinguish them as Inbound and Outbound.

It was also important to analyse the TCP Session Establishment attempts, differentiating the connections that were established from the unsuccessful connection attempts. In order to obtain this statistic, we had to filter the SYN packets from the SYN/ACK and RST/ACK packets. TCP session establishment follows the three way-handshake scheme illustrated in the following figure.

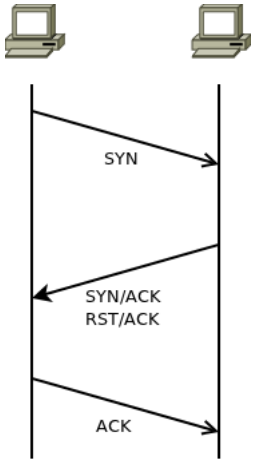

Figure 4.1: TCP Session Establishment diagram

The ACK packets were not considered for this analysis. In fact, the number of ACK packets can be easily manipulated by changing the flag bits in order to obtain a particular value. In our procedure, we split the TCP SYN packets sent from the infected machine from the TCP SYN packets received at the infected machine.

Finally, a World Map showing the geographical location of the contacted peers was built in order to observe the geographiccal relevance of each Botnet. This was possible by resorting to a tool in Python[36]that allows to know the peers geographical location based on their IP addresses. With a script, providing GnuPlot's world map and the tool for getting the IP address coordinates, it was possible to pin down these coordinates in the map.

In order to better understand the Botnet behaviour, it was decided to split the traffic originated in the infected machine from the traffic that is destined to the infected machine. Hence, for further reference, we will describe them as the Upload and Download traffic, respectively. On the World Map, this distinction was made by using black dots for the Outbound connections and red dots for the Inbound connections.

In the low level analysis, we decided to analyse the following static/metric:

- Scalograms

Scalograms are a visual method of displaying wavelet transforms. They have three axes, representing time, scale and the energy coefficients. This is one method of using wavelet analysis to obtain spectral information. Scalograms allows us to analyse wavelets in different scales of frequency and time. They also help detect the variance of a signal.

An analysis of the variance and mean of the energy of the scalograms was also made. Only the first ten hours of the captures were considered to build the scalograms.

All these analysis were possible due to the use of Tshark to capture the generated traffic, Matlab and GnuPlot to make the graphics of the different statistics and Python to write scripts that are used to extract all the necessary information from the generated traffic.

Next sections will present the analysis that was made to the traffic generated by each Botnet, together with a discussion of the results obtained.

4.1 Grum

As it has been said before, the malware from this Botnet was installed immediately after the computer was formatted. Traffic was captured during 48 consecutive hours, in order to make a deeper analysis. The malware used was downloaded from the site [35] on July 2011.

This malware had a particular characteristic: every time it was executed, it created a new process, disguised under the name of Internet Explorer ("iexplore.exe").

Although we have repeated the same process from scratch, the traffic obtained in the second capture was very similar to the first one, so only one capture will be presented and analysed.

4.1.1 General analysis

The first task when analysing the captures obtained from Grum was to see how traffic behave over time. By analysing the whole capture, it was possible to take the following conclusions.

Immediately after installing the malware, DNS queries started being made to one of Google's DNS (8.8.8.8) during 100 seconds, with a periodicity of 50 seconds. Then, most of the traffic that was filtered as TCP Unknown in the high-Level analysis used port 80, so we can conclude that it is HTTP traffic. The known HTTP traffic was generated during 150 seconds, also with a periodicity of 50 seconds, making various GET requests for different types of files (.exe, .gif, .png). After some time, Unknown TCP packets, directed through port 445, were exchanged. Therefore, we were seeing a communication of Server Message Block (SMB) over TCP/IP. Right after this exchange, the SMB communication began. A series of requests and responses were exchanged. The objective of this activity was to find shared files. A Session Initiation Protocol (SIP) packet was then received, including the information that is necessary to get options from an IP address. This packets continued to appear sporadically. Besides, several attempts for Secure Shell (SSH) and Telnet connections were also made. Recurrently, there were some packets being exchanged through port 6000, which has been reported as a port used by virus or trojans. Some SMTP packets were also detected over time, reinforcing the idea of spam intents.

The capture made followed a regular trend, with the vast majority of the packets belonging to HTTP and SMB. These packets continuously queried services through NetBIOS Name Service (NBNS) and tried to establish sessions through NetBIOS Session Service (NBSS). There were some exceptions that will be discussed later.

Before presenting the statistics that were extracted, we can conclude that this Botnet performed somewhat like we expected it to, considering that its traffic is mostly HTTP, it performs DNS lookups and the countries of infection are mainly located in the same continents that have been reported in the literature. The number of SMTP packets was expected to be higher, but the most important issue is that these packets are present.

4.1.2 High-Level Analysis

First of all, we analysed the protocols involved in the traffic that was generated by this Botnet.

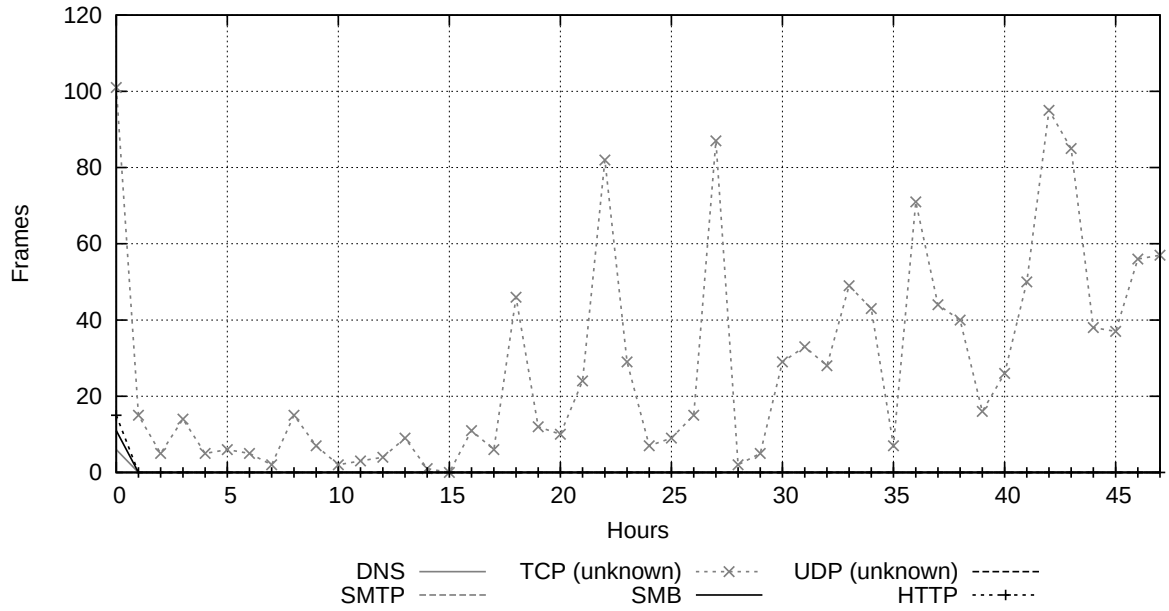

Figure 4.2: Protocols(Upload), Grum Botnet

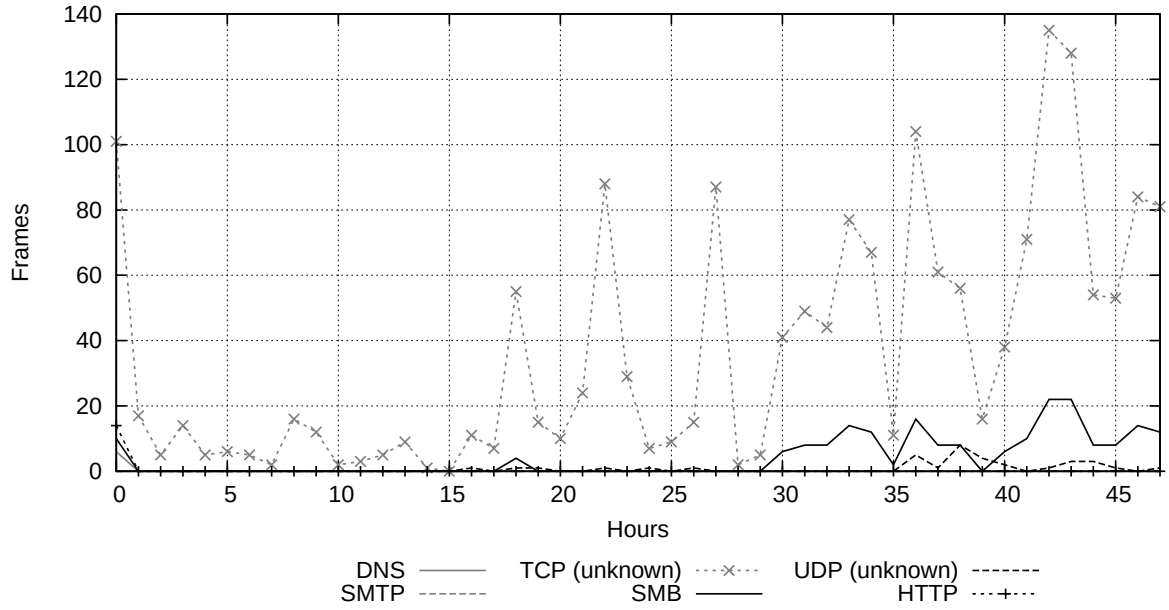

Figure 4.3: Protocols(Download), Grum Botnet

Regarding these two pictures, it is clearly visible that most of the generated traffic was filtered as unknown TCP. In the Upload direction, there are some HTTP and DNS packets in the first hour. Then, besides unknown TCP, there are also some SMB and unknown UDP packets.

In the Download direction, it is also possible to observe some few packets from three different protocols (HTTP, DNS and SMB), although this only happens in the first hour.

As it has been previously said, most of the unknown TCP packets are really HTTP or SMB packets.

The next procedure consisted in analysing the number of packets generated per hour and per minute.

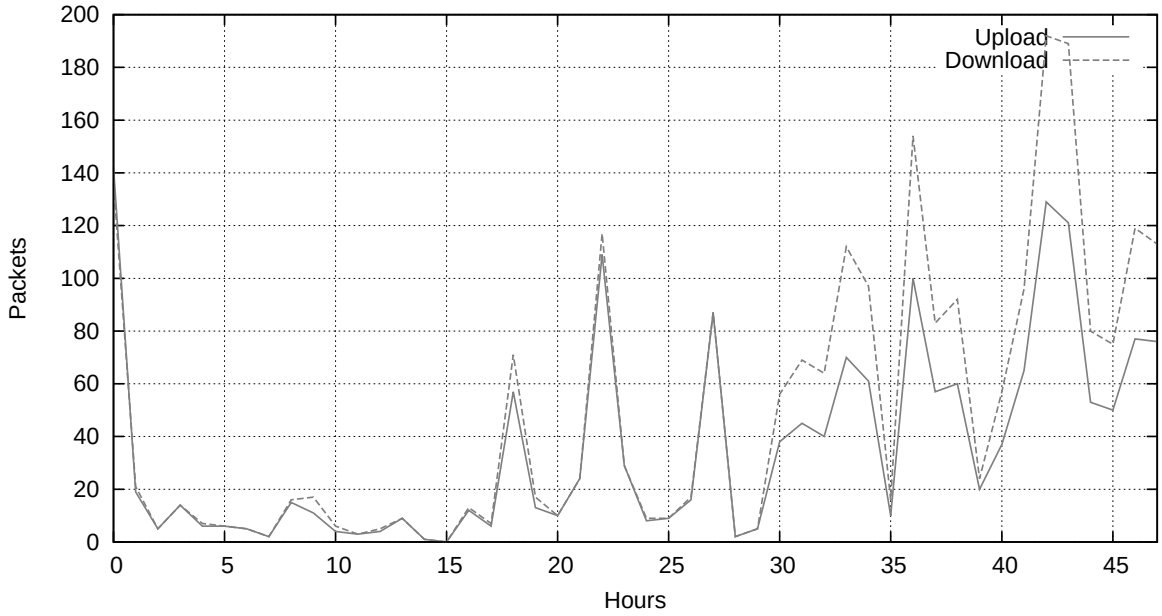

Figure 4.4: Packets per hour, Grum Botnet

In this picture, it is visible that the number of generated packets increased as time progressed, and there were always more Download than Upload packets. There are peaks in the amount of generated traffic around the 23th, 37th and 43nd hour.

In these peaks, it is possible to observe an increase of SMB session requests, as well as Remote Management requests.

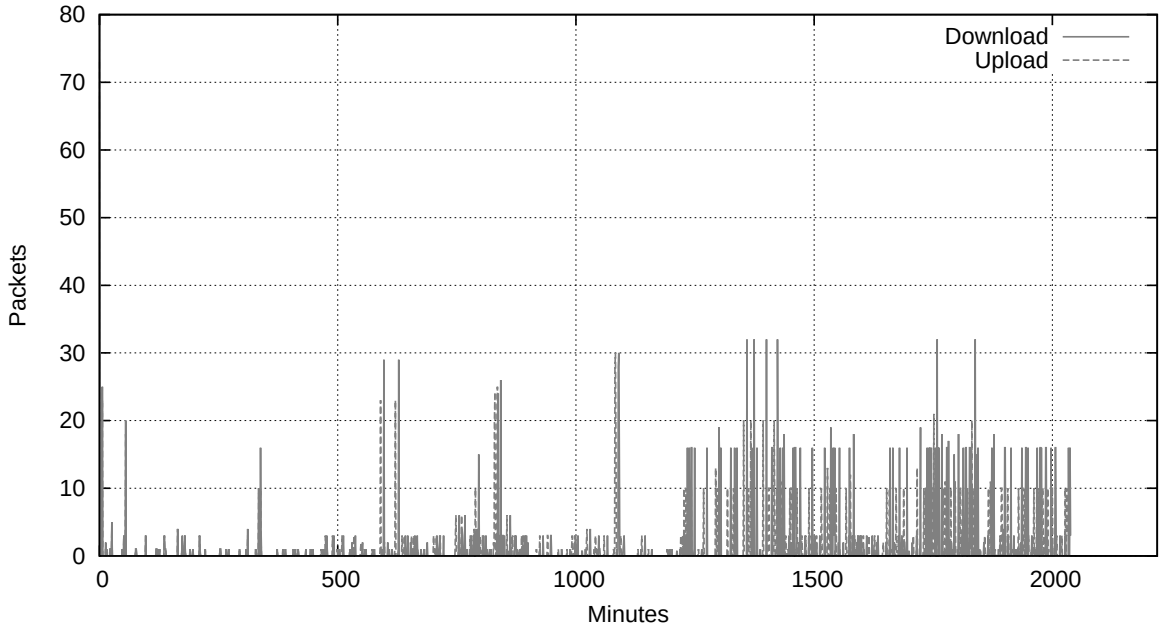

Figure 4.5: Packets per minute, Grum Botnet

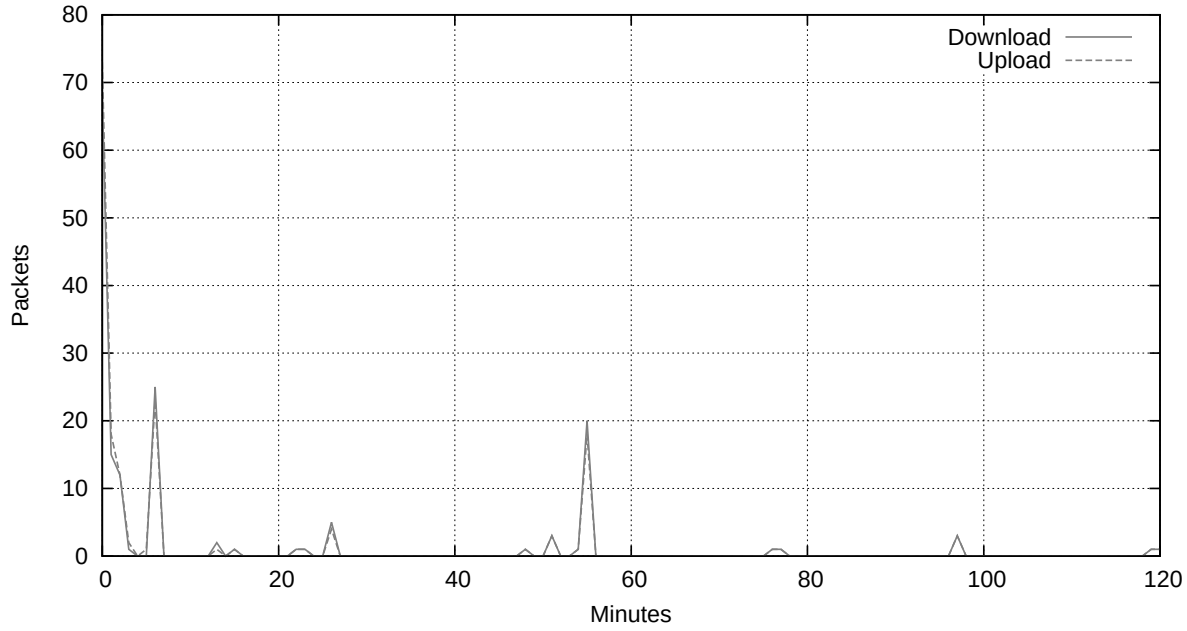

Figure 4.6: Sample of packets per minute, Grum Botnet

By observing these last two pictures, we can conclude that this Botnet does not generate a significant amount of traffic per minute, except on some peaks that occur over time, as previously explained.

From the captured packets, it is also important to observe the amount of generated data. The analysis made follow the same criteria of the previous procedure.

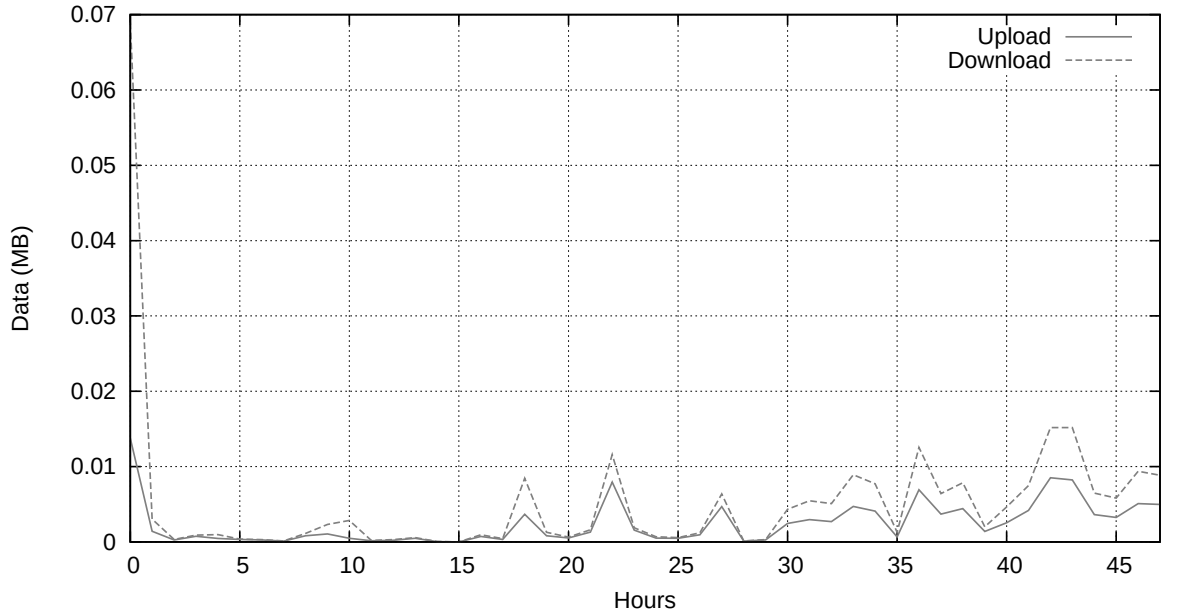

Figure 4.7: Amount of traffic per hour, Grum Botnet

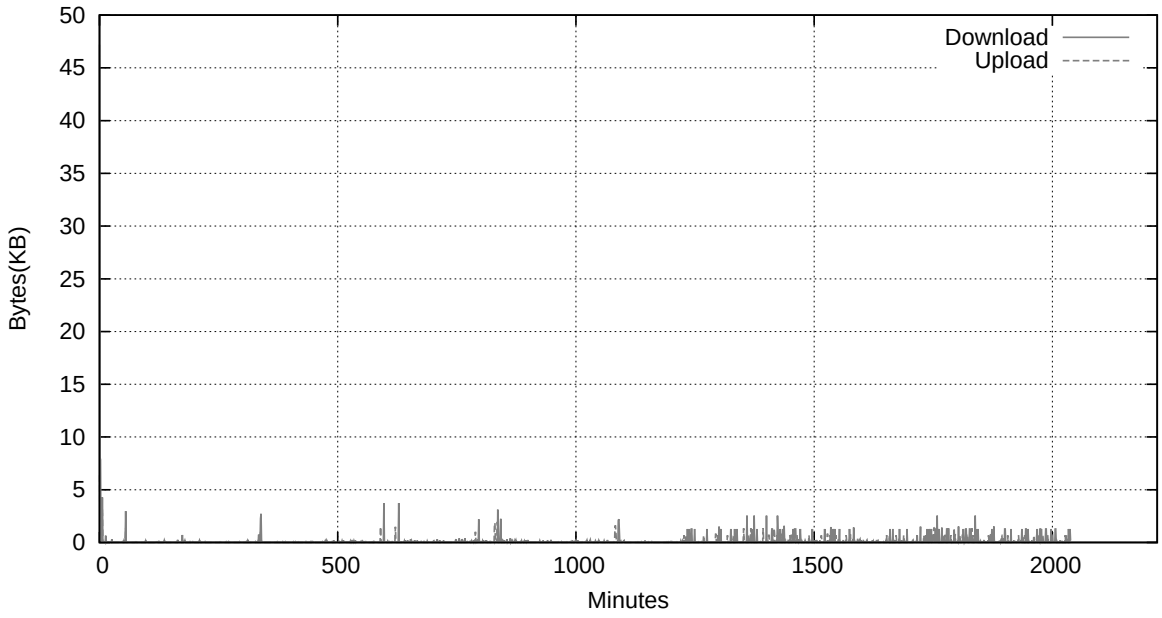

Figure 4.8: Amount of traffic per minute, Grum Botnet

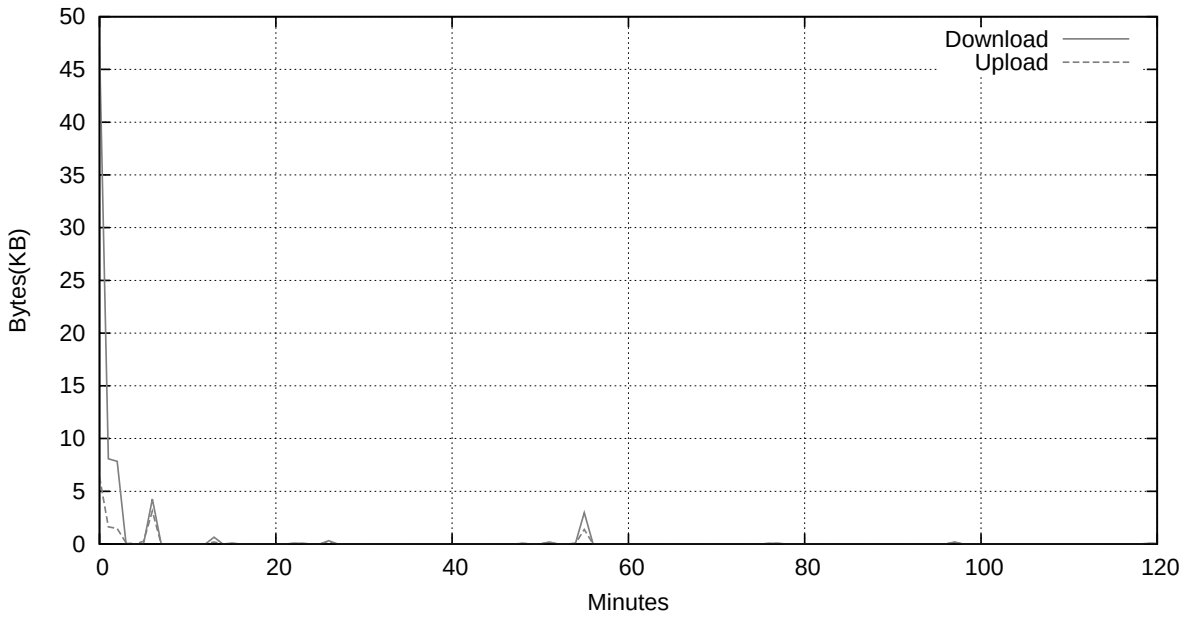

Figure 4.9: Sample of amount of traffic per minute, Grum Botnet

From these three pictures we can observe that Grum generated a very limited amount of traffic, around 10KB per hour. As expected from the amount of received packets, the amount of download traffic was always higher than the amount of upload traffic.

Next graph shows the amount of unique peers contacted over time.

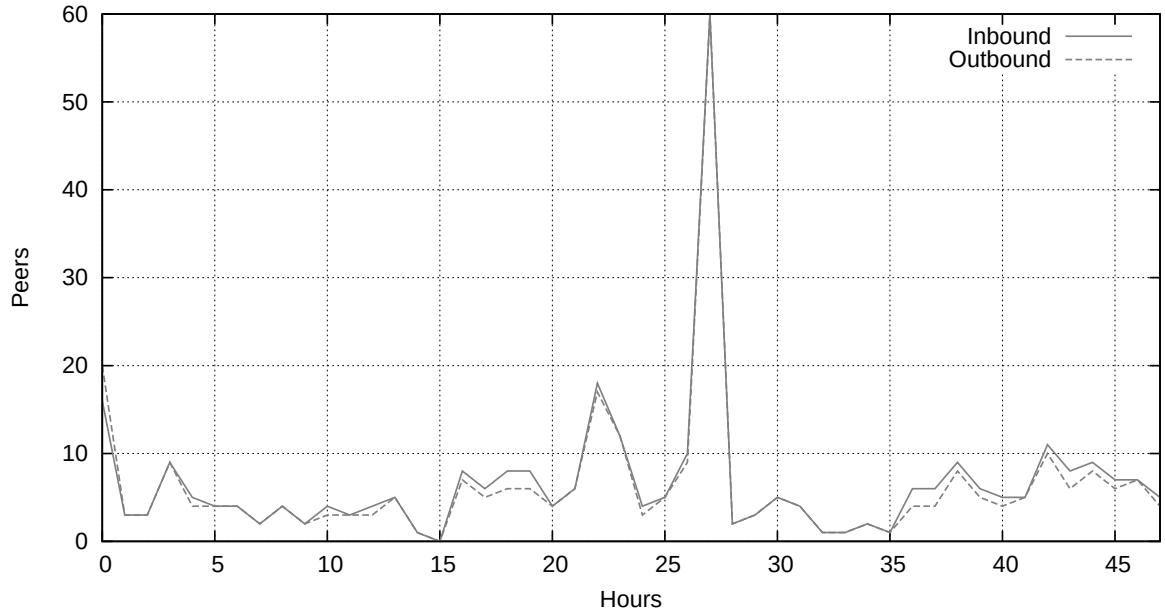

Figure 4.10: Unique peers per hour, Grum Botnet

We can see that there was a fairly regular amount of peers contacted per hour, except for the peak on the 28th hour, where six times more peers were contacted than usual. This peak was the result of the attempts of TCP Session Establishment that were not successful.

Another procedure that is useful to better understand the behaviour of this Botnet is to analyse the TCP Session Establishments. This analysis can be seen in the following pictures.

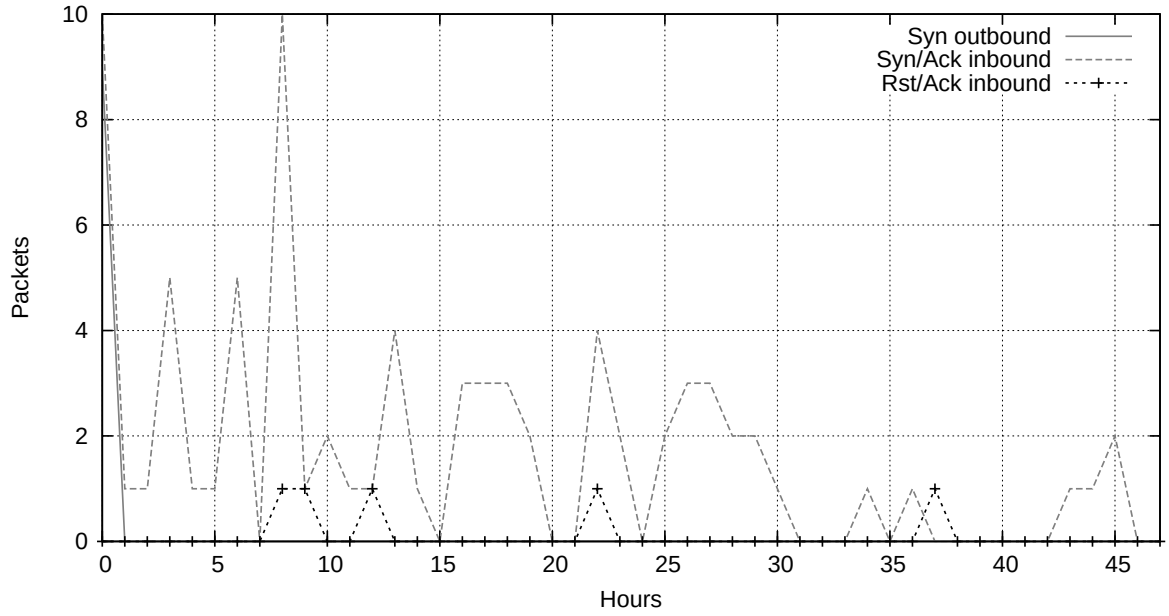

Figure 4.11: TCP Session Establishment(Outbound), Grum Botnet

In the previous picture, it is clear that most of the packets generated in response to SYN packets were SYN/ACK, so the session establishments followed the expected behaviour.

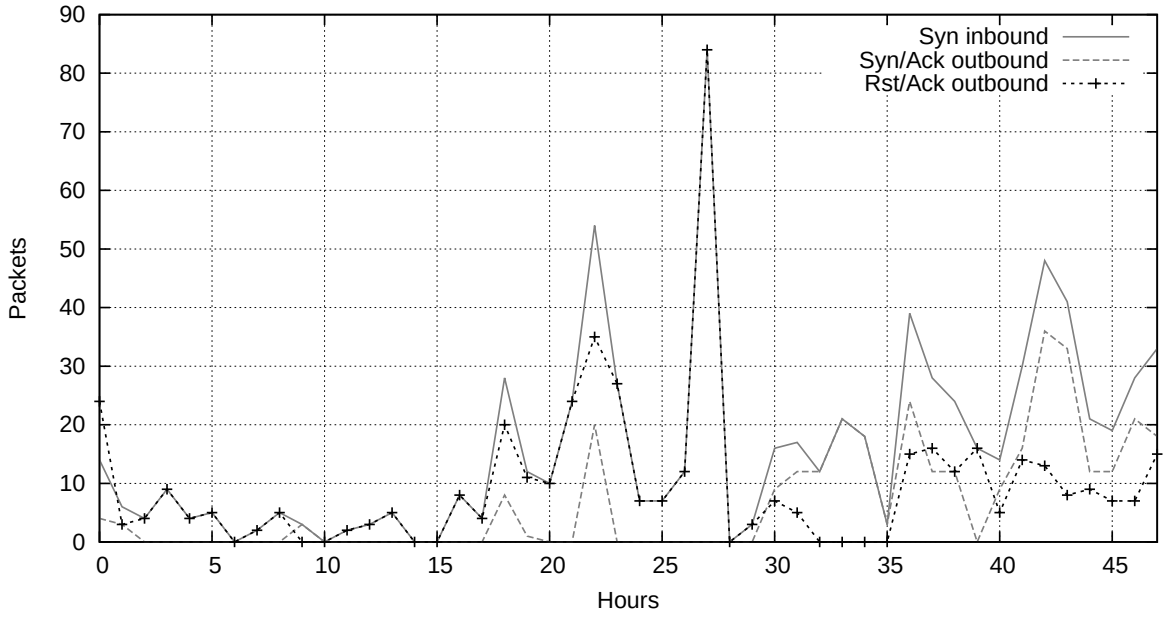

Figure 4.12: TCP Session Establishment(Inbound), Grum Botnet

In the Inbound picture, we see that most of the time there are more RST/ACK packets than SYN/ACK, which means that most of the session establishments attempts were not successful. Like we said before, there is a strange peak of 60 peers contacted per hour, which is due to these high number of RST/ACK packets. In the 28th hour of this capture, a total of 84 RST/ACK packets were sent from the infected machine.

To conclude the High-Level analysis, it is interesting to see a world map that shows the geographical location of the peers that established communication with the infected machine.

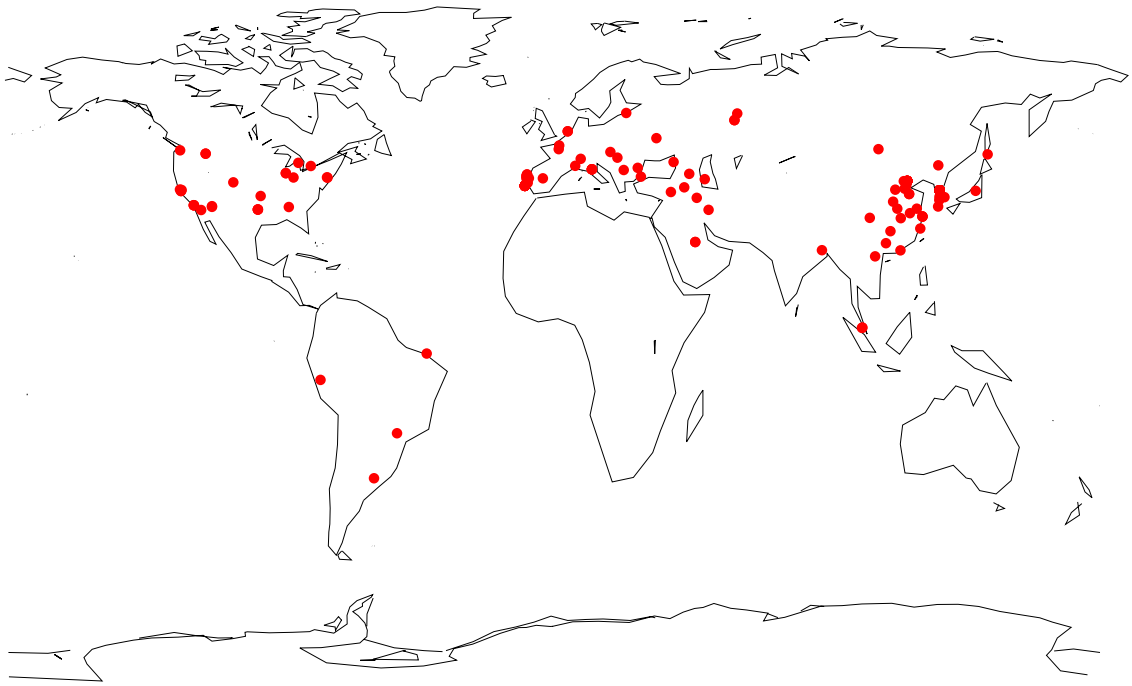

Figure 4.13: World Map, Grum Botnet

It is perceivable that Grum's infected machines are primarily located in Europe, Asia and America. The main infected countries reported in this capture are China and the United States of America.

The location of the Inbound and Outbound peers was very similar,so we decided to present only one of these statistics.

4.1.3 Low-Level Analysis

The scalogram for this capture is shown in the next picture.

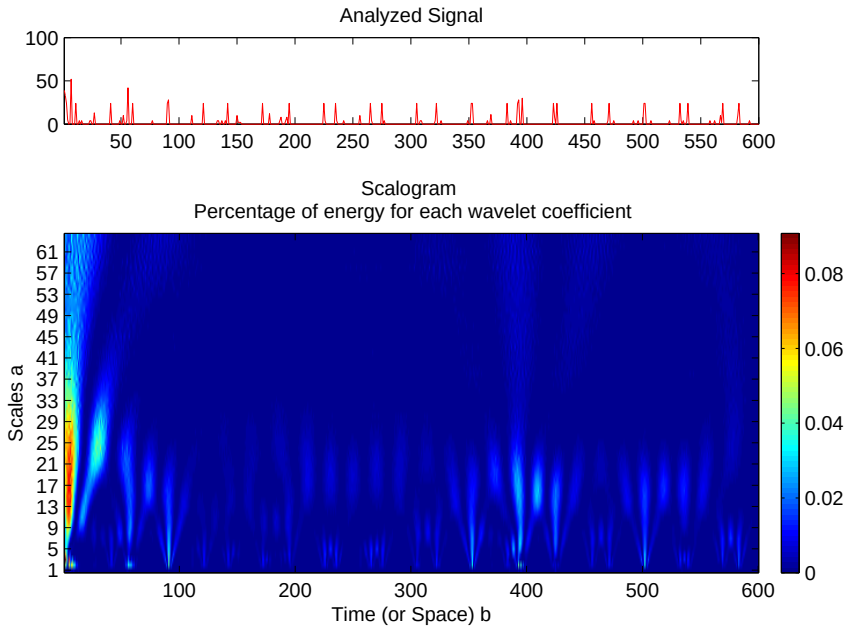

Figure 4.14: Scalogram, Grum Botnet

The increase of energy percentage around the first minutes of the sample is clearly visible, which indicates the high variance at the beginning of the capture. There are also some smaller variance peaks occurring at minutes one hundred, four hundred and five hundred. This scalogram allow us to clearly see the variance of the signal over time.

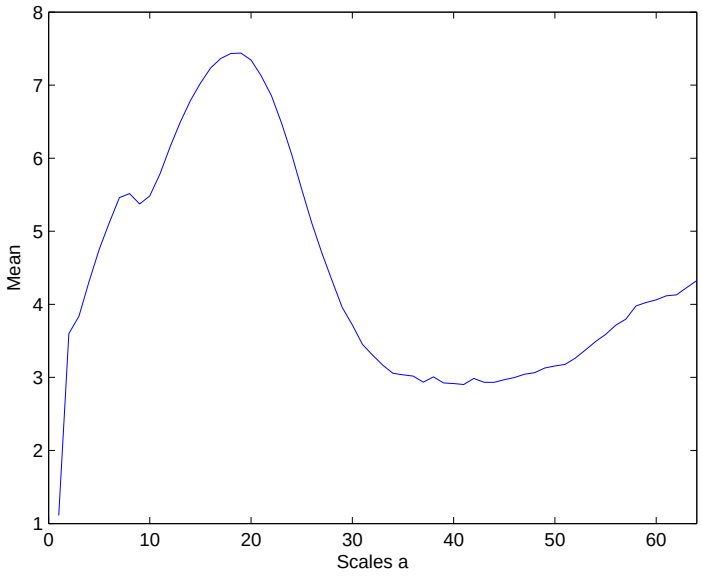

Figure 4.15: Energy Mean, Grum Botnet

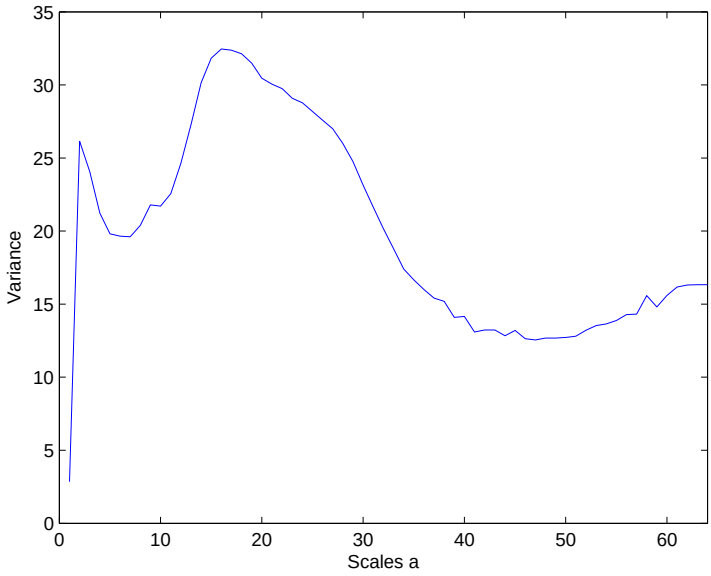

Figure 4.16: Energy Variance, Grum Botnet

The mean and variance of the energy were also analysed. We can see in the last two pictures that the mean is quite different for the various coefficients of energy. The variance of the energy has significant values.

4.2 Cutwail

Once again the computer was formatted and the malware corresponding to this Botnet was installed. Generated traffic was captured for 48 hours, in order to make a deeper analysis of this Botnet behaviour. The malware used was downloaded from [35] on July 2011.

The second capture that was made produced different values, so both captures will be presented in order to show their different behaviours. This proves that even though Botnets have a behavioural pattern, they have the capability to behave differently under similar circumstances.

4.2.1 General analysis

Analysing the captures obtained from Cutwail, the main objective was to see how it behaved over time. After the complete analysis, it was possible to take some important conclusions.

In the first capture, just after the malware has been installed, NBNS requests to the infected machine router started to be executed every 700 seconds, in order to obtain information from other machines in the network. It was possible to see that port 59022 was used for SSH communication, after opening the firewall and allowing SSH in that specific port. The Unknown UDP packets that will be shown in the statistics actually refer to DoS attacks. Many of the used ports are known to be frequently used for DoS attacks, so we believe that the main objective was to disrupt the services running on the infected machine.

At hour 11, there is a SIP packet including the information that is necessary to get options from an IP address. At hour 15, there are a couple of Simple Network Management Protocol (SNMP) packets to make requests. Starting in hour 29, most of the Unknown TCP packets are actually SSH and Telnet packets, attempting to make remote connections with the infected machine. Finally, at the 45th hour the Unknown TCP packets are not any more SSH and Telnet, but are now HTTP packets. There are only a couple of SMTP packets, which confirms the idea that the main purpose of this Botnet is to disrupt services.

In the second capture, what was more common when analysing traffic was the presence of HTTP packets. Actually, most of the Unknown TCP packets are in fact HTTP packets. After a couple of hours, some HTTP/XML Notify packets were spontaneously exchanged. There were a couple of SIP packets as well, with both the OPTIONS information and INVITE. A couple of Telnet packets were also seen in the first hours, but nothing too suspicious. There were some NBNS Query packets as well, also using port 445, and a significant amount of SMB over TCP/IP packets were also detected.

Regarding Unknown UDP traffic, most of it was actually being exchanged for DoS attacks. Many of the ports were recognized as the ones that are usually used for this type of attacks.

Around the third hour of the capture, a lot of Unknown TCP packets started being directed through port 50000, known for being used by a trojan named SubSARI. This activity lasted until hour nine, where this port stopped being used almost at all. By the end of the capture, there were some Remote Management packets. To conclude, once again SMTP packets were detected but now in a smaller quantity.

We can conclude that this Botnet did behave like it was expected, considering the amount of HTTP traffic, the DNS lookups, DoS attacks and also the countries of infection, which were mainly focused on the same continents there were reported in [8]. The only exception was that more SMTP traffic was expected, leading us to suspect that this Botnet did not perform exactly at it should.

4.2.2 High-Level Analysis

Following the same procedure of the previous Botnet, we first analysed the protocols involved in the generated traffic.

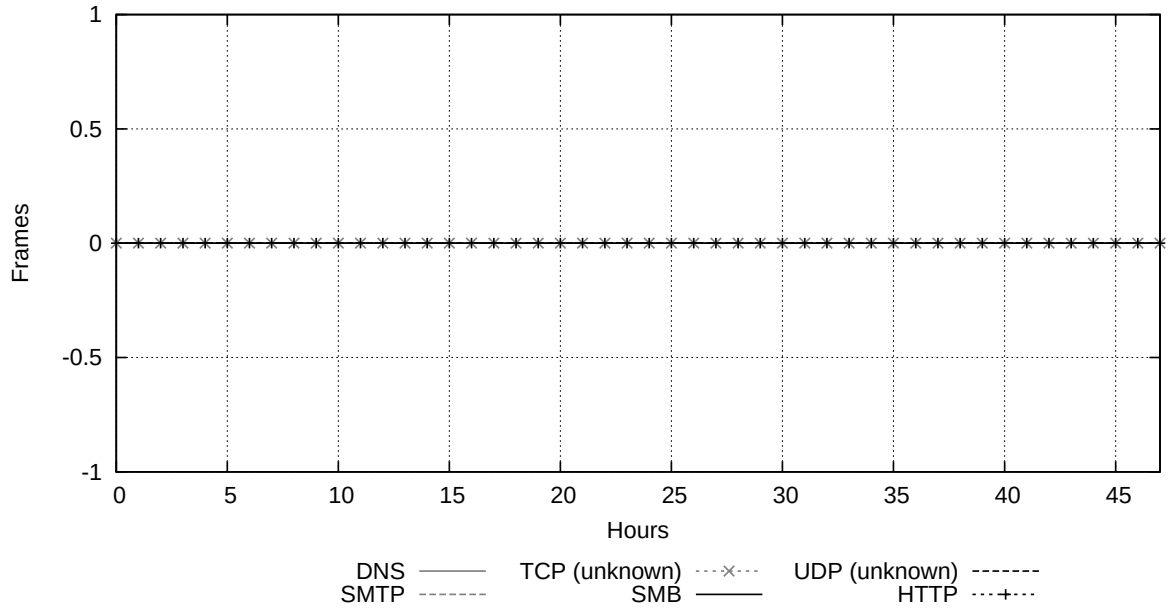

Figure 4.17: Protocols(Upload), Cutwail Botnet

As we can see in the first capture, the mostly used Protocol is Unknown TCP, which has an erratic behaviour. Then, in some time frames we have Unknown UDP packets as well as DNS packets. Like we already discussed, mainly this Unknown TCP packets are really SSH, Telnet and HTTP packets.

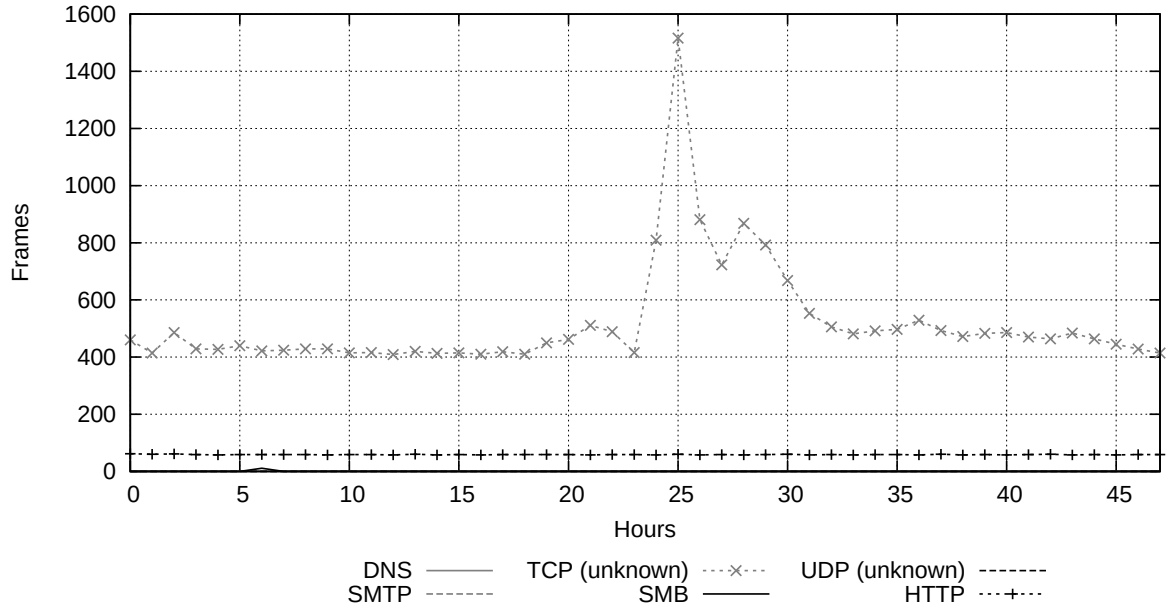

Figure 4.18: Protocols(Upload), Cutwail Botnet(2nd Capture)

Contrasting to the first capture, the second capture shows a perfect example of a simple behavioural pattern. It has a constant rate of sent HTTP packets and the Unknown TCP packets, despite having a peak near the 26th hour, also follow a simple pattern. There is a small rate of SMB and Unknown UDP packets. The latter has a peak around the 25th hour as well.

This peak is caused by TCP Session Establishments, which will be discussed below.

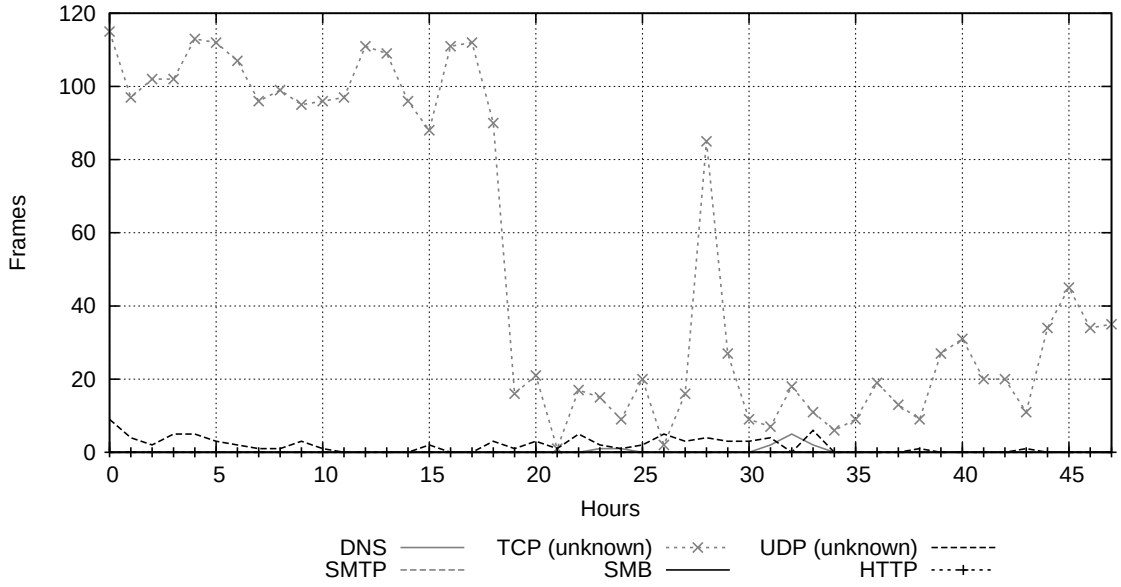

Figure 4.19: Protocols(Download), Cutwail Botnet

This is an interesting picture. Despite the received packets, not even a single packet from the Protocols shown in the figure was sent by the infected machine. The only packets sent from the infected machine were NBNS packets.

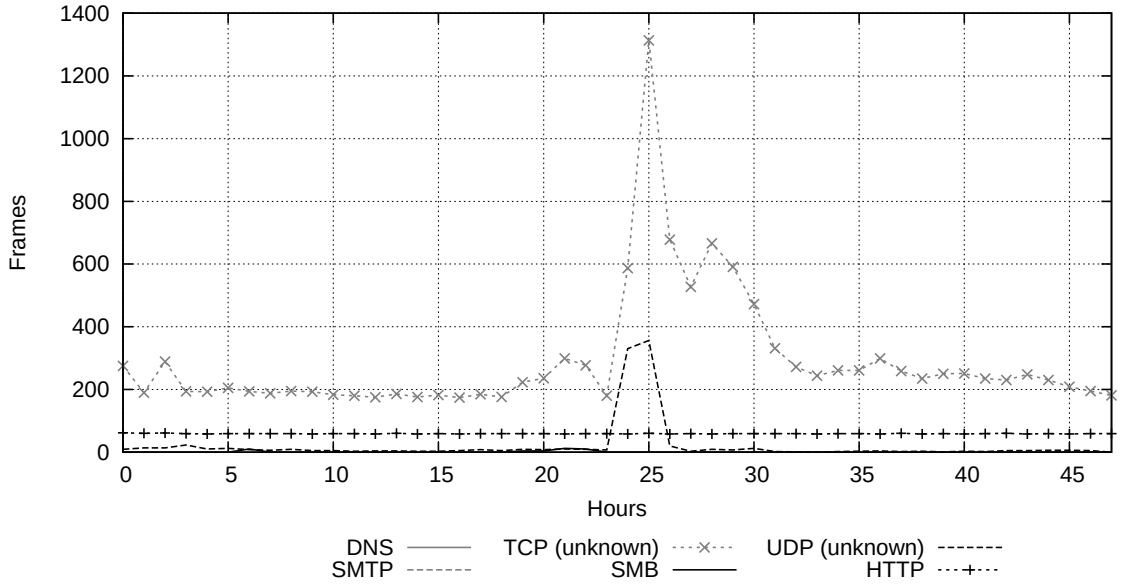

Figure 4.20: Protocols(Download), Cutwail Botnet(2nd Capture)

This last picture is very similar to the Upload one, except in the absence of the Unknown UDP packets peak.

In the last four pictures, it was possible to understand the importance of making several captures of the same Botnet. There were some significant differences between both captures, although the general behaviour remains the same. The most present Protocol is once again Unknown TCP.

Like we already discussed, most of the Unknown TCP traffic is actually HTTP, while Unknown UDP is mostly sent through different ports in order to perform DoS attacks.

Analysing now the number of packets generated per hour and per minute, we obtained the following statistics.

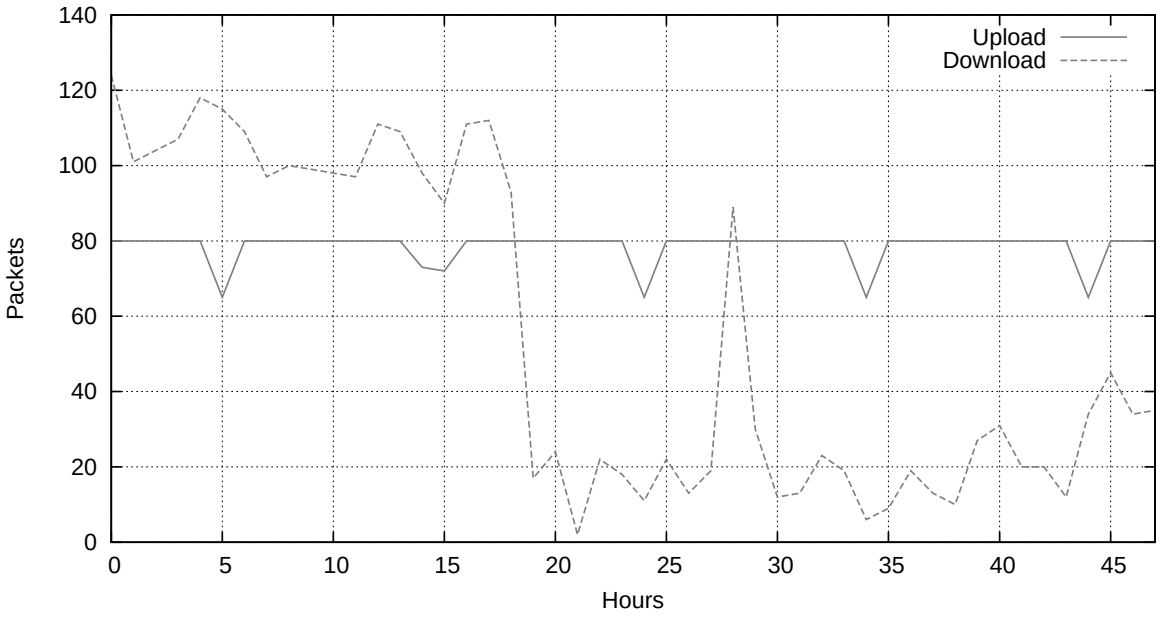

Figure 4.21: Packets per hour, Cutwail Botnet

In the first capture, we can observe a constant pattern of packets sent per hour. The number of packets received has, however, an unpredictable pattern.

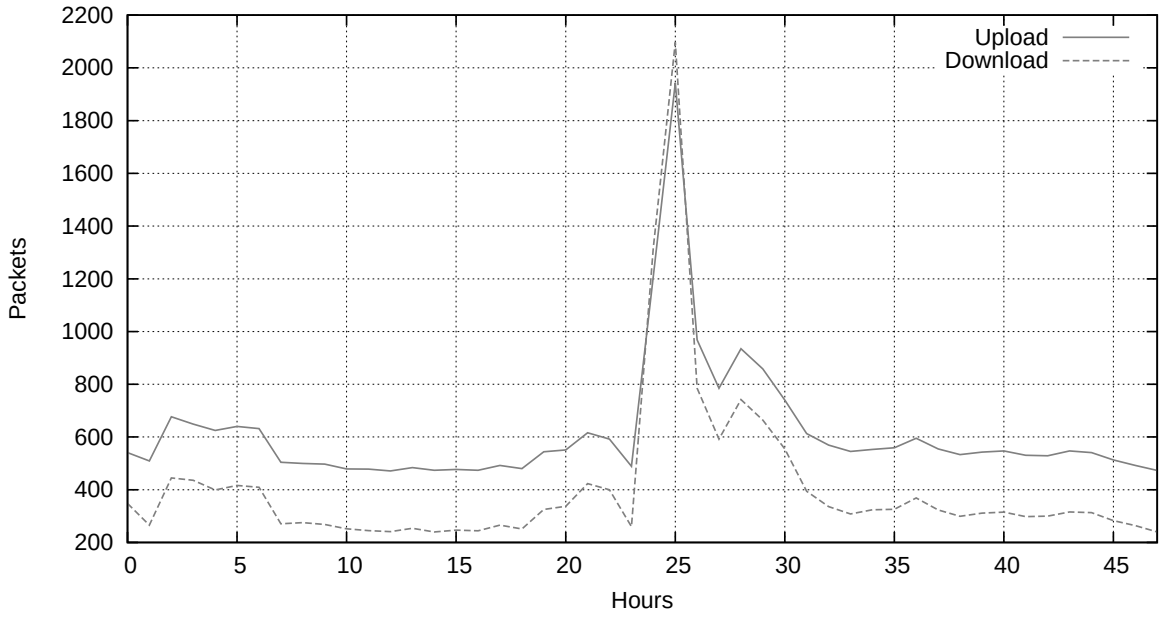

Figure 4.22: Packets per hour, Cutwail Botnet(2nd Capture)

In this picture we can observe that both sent and received packets follow the same pattern, and both have a peak in the 26th hour. Once again, this peak is originated by the TCP Session Establishment attempts. There was almost always more Upload than Download packets.

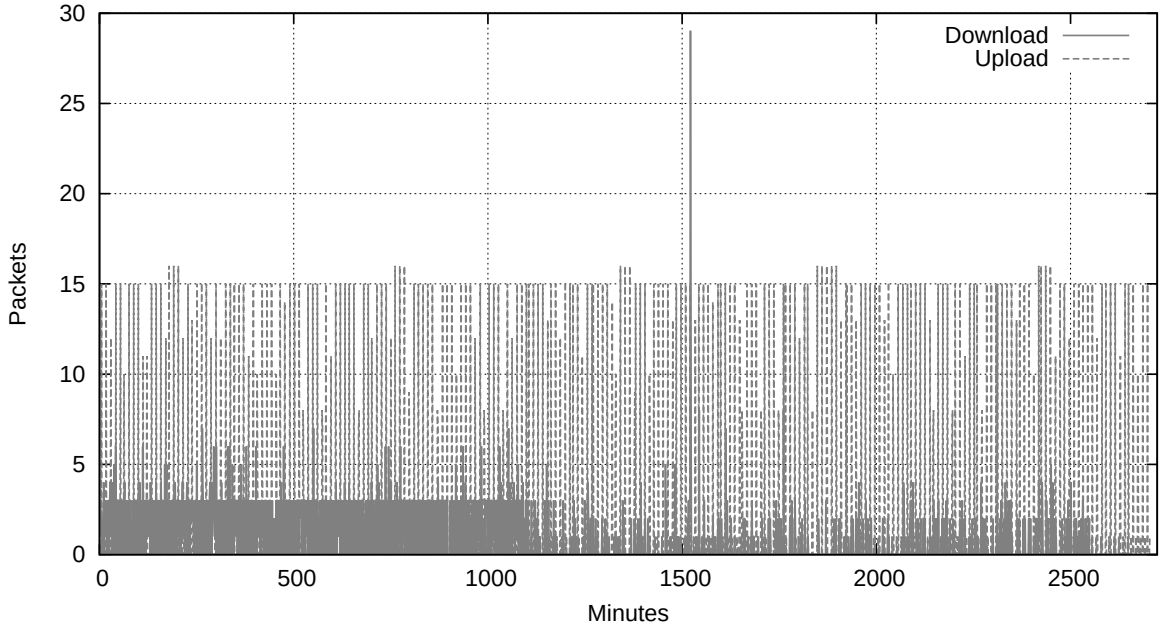

Figure 4.23: Packets per minute, Cutwail Botnet

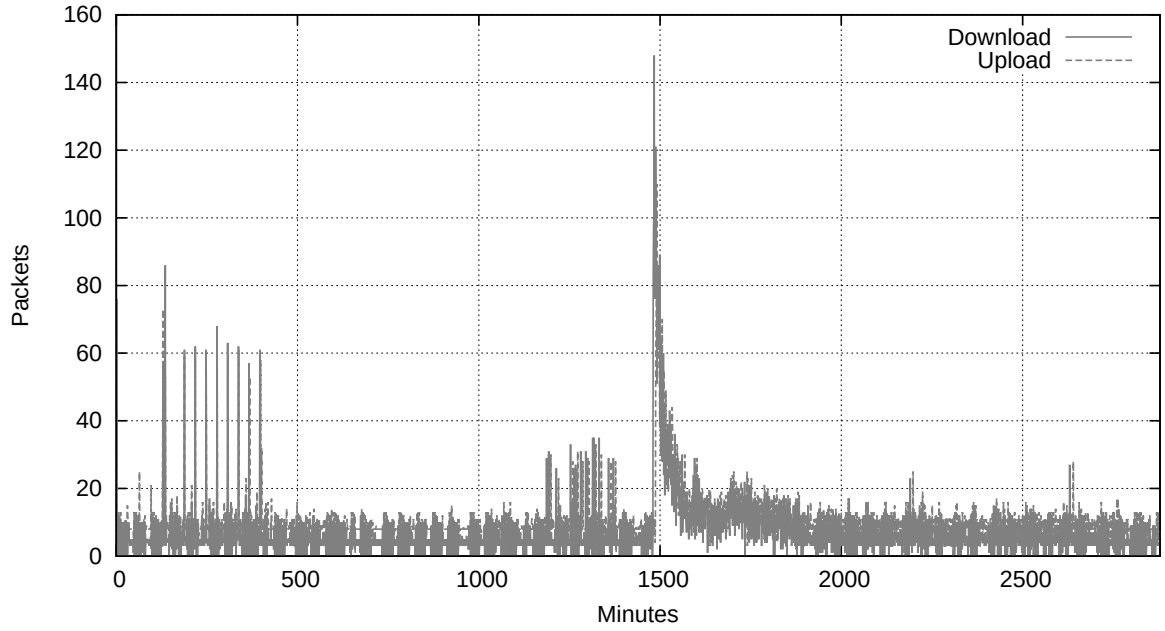

Figure 4.24: Packets per minute, Cutwail Botnet(2nd Capture)

These last two images better explicit the behaviour along the 48 hours of the capture, which was not very clear yet, specially in the first capture.

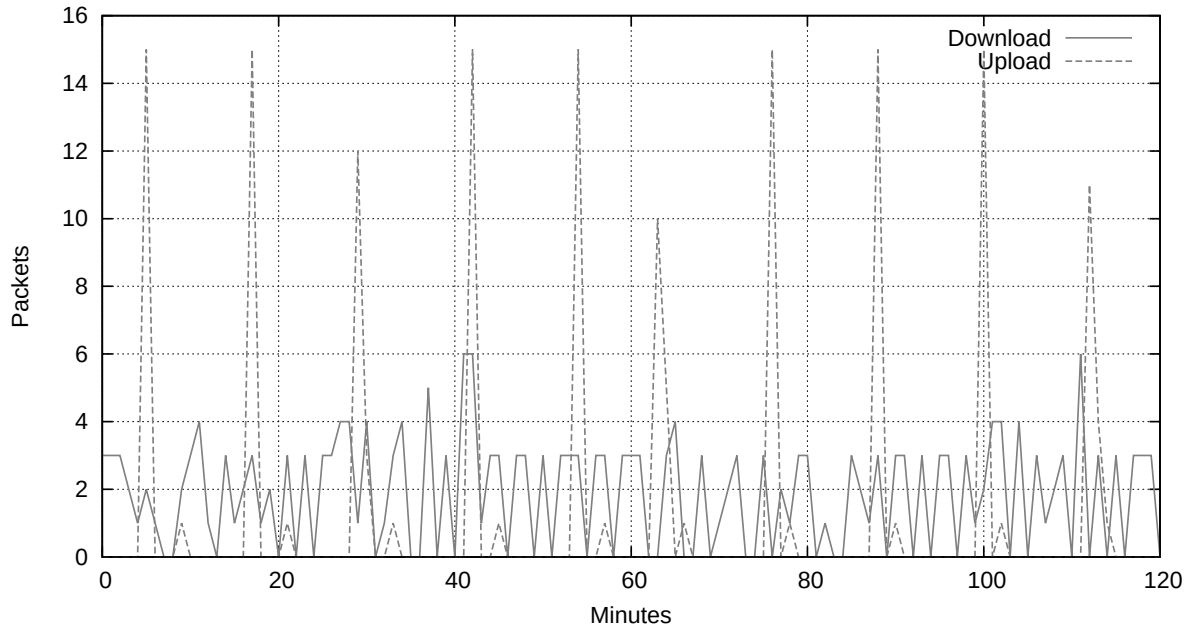

Figure 4.25: Sample of packets per minute, Cutwail Botnet

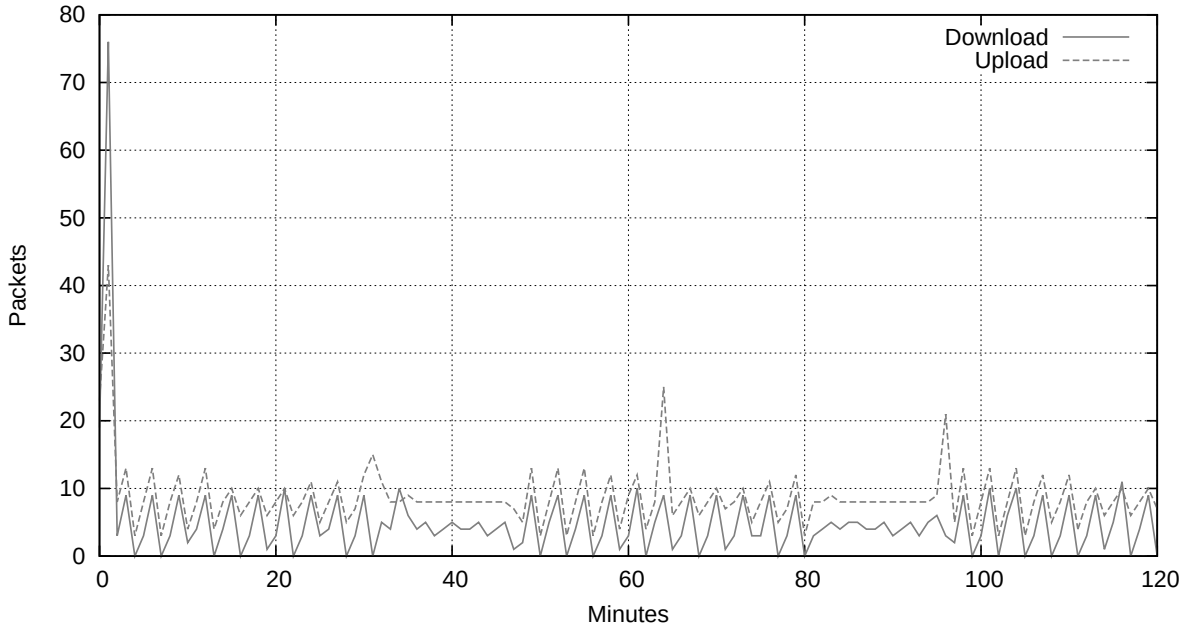

Figure 4.26: Sample of packets per minute, Cutwail Botnet(2nd Capture)

These two samples from the first two hours are useful to clearly see the behaviour of Cutwail minute by minute.

Moving now to the amount of generated data, we can see that these two images are very similar to the ones corresponding to the amount of packets per hour.

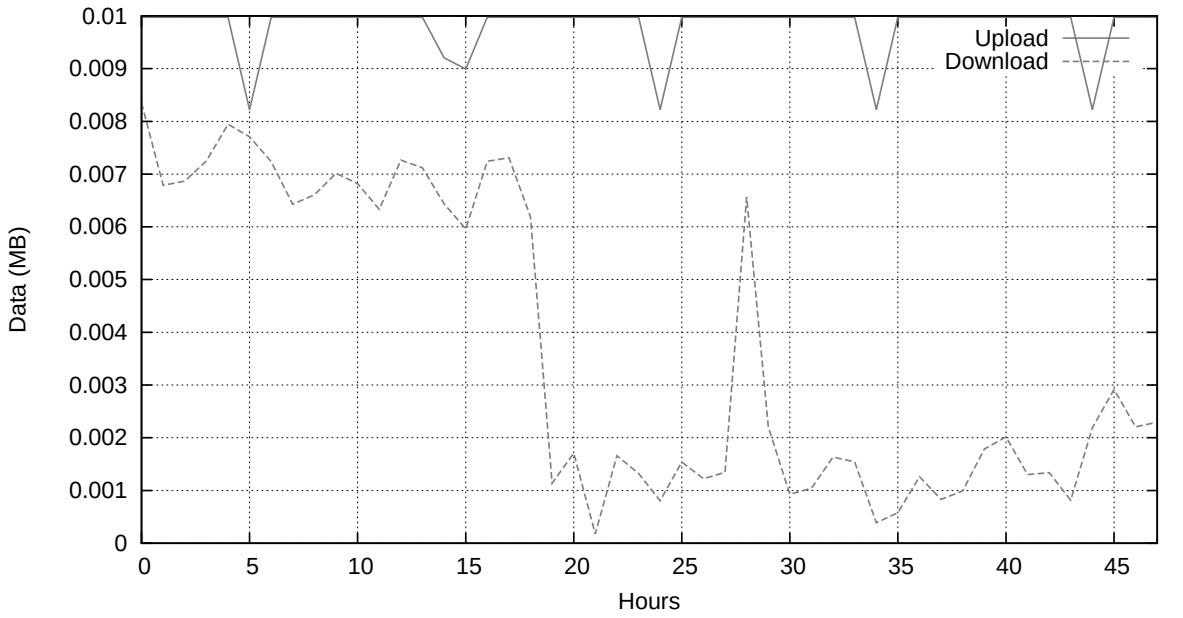

Figure 4.27: Amount of traffic per hour, Cutwail Botnet

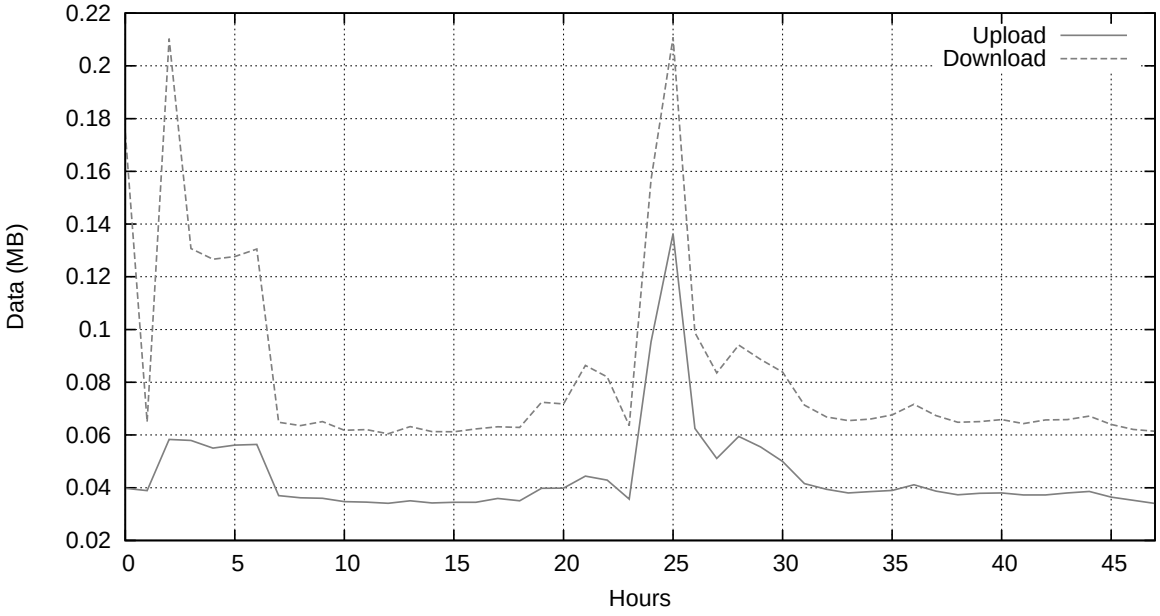

Figure 4.28: Amount of traffic per hour, Cutwail Botnet(2nd Capture)

The differences are that in the first capture there is always a high quantity of Uploaded traffic than Downloaded, while in the second capture it is exactly the opposite situation.

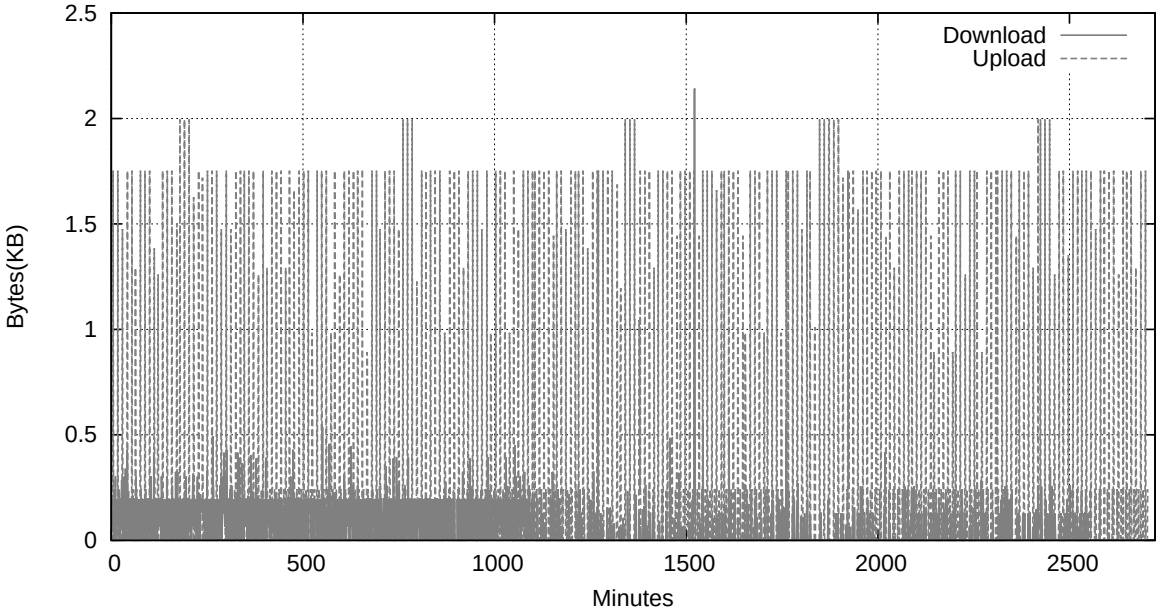

Figure 4.29: Amount of traffic per minute, Cutwail Botnet

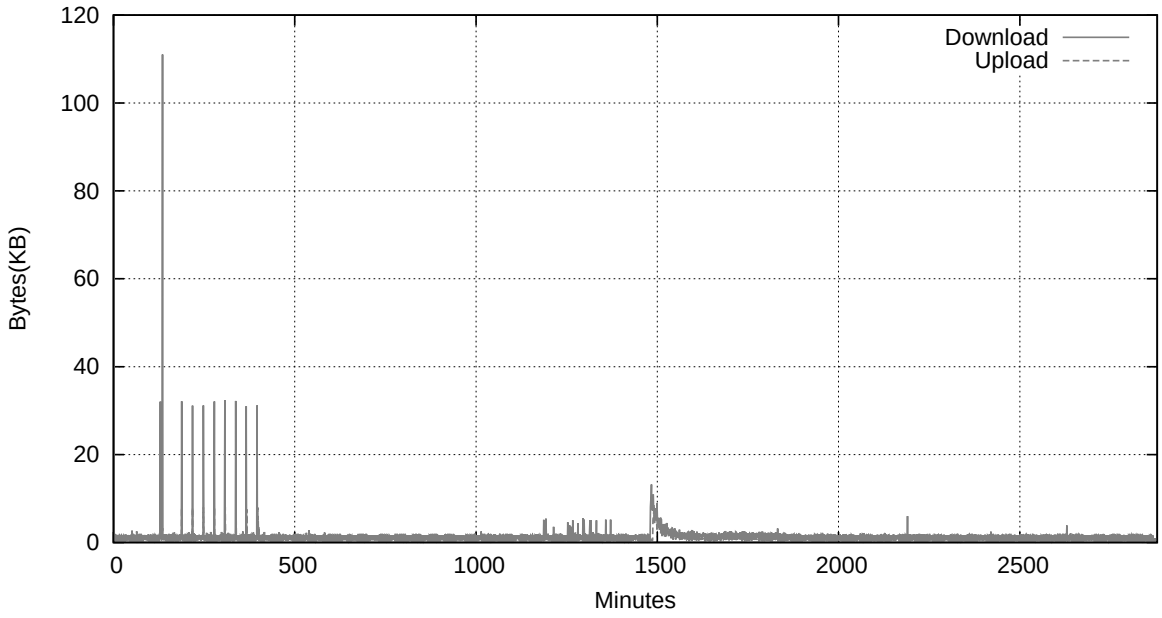

Figure 4.30: Amount of traffic per minute, Cutwail Botnet(2nd Capture)

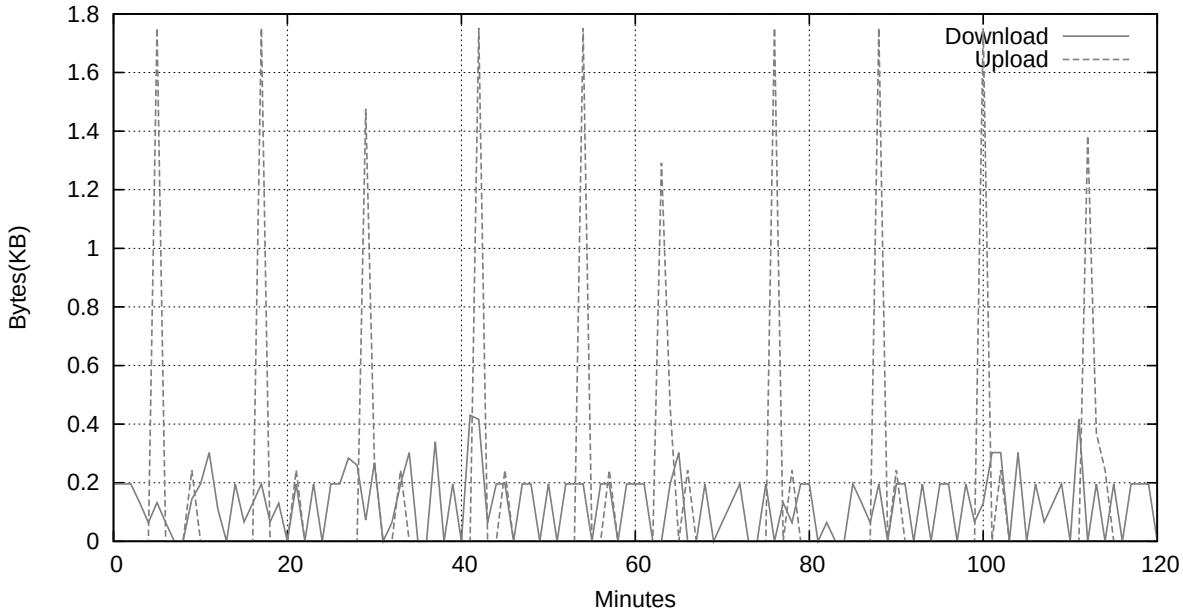

Figure 4.31: Sample of amount of traffic per minute, Cutwail Botnet

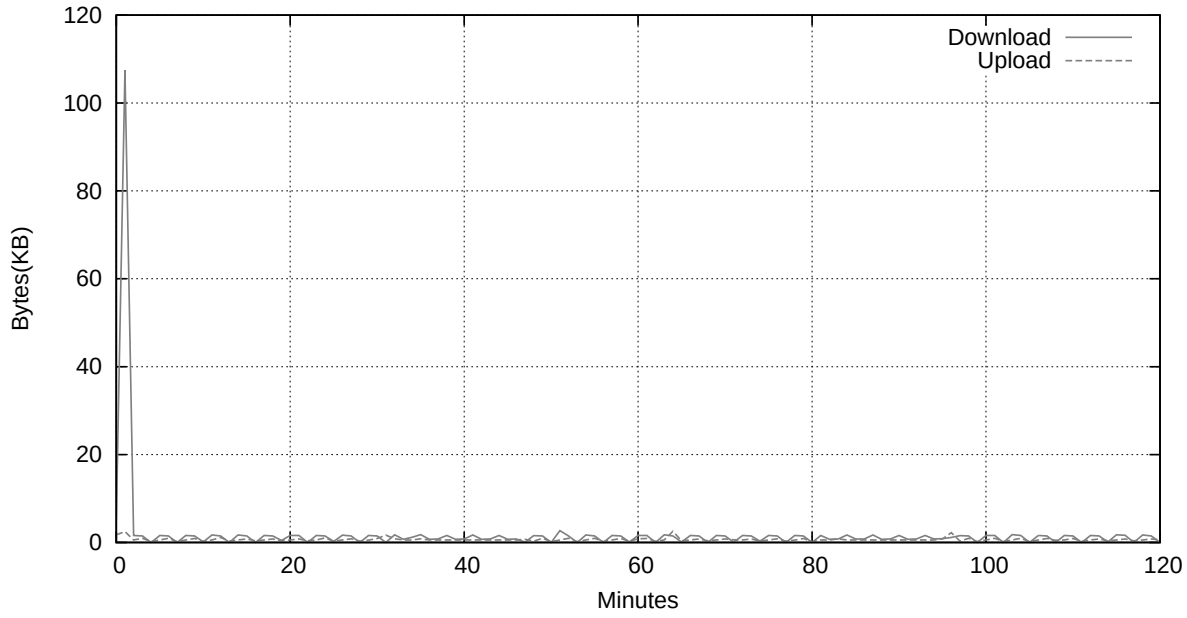

Figure 4.32: Sample of amount of traffic per minute, Cutwail Botnet(2nd Capture)

These last pictures have the same objective as the ones corresponding to the packets by minute and its corresponding samples. Cutwail generated a very small amount of traffic, which can be confirmed in the last six pictures.

The amount of unique peers contacted over time was also analysed and can be seen in the next two pictures.

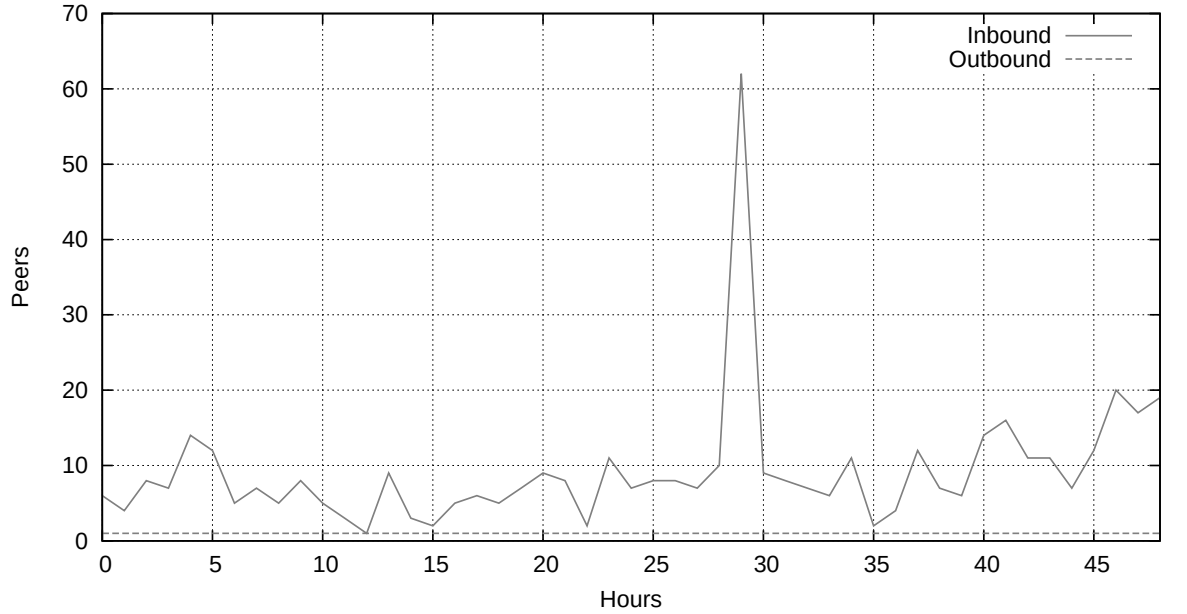

Figure 4.33: Unique peers per hour, Cutwail Botnet

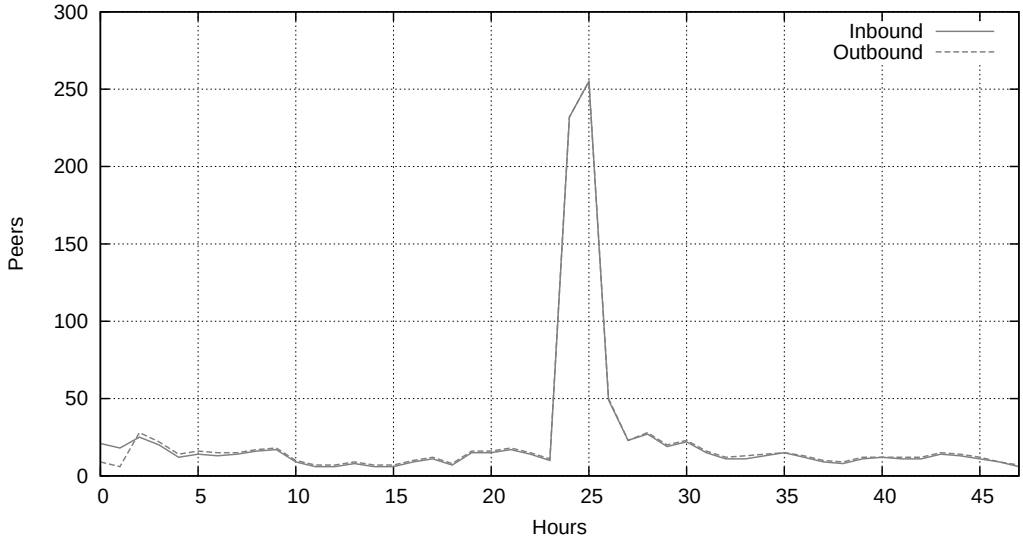

Figure 4.34: Unique peers per hour, Cutwail Botnet(2nd Capture)

We can observe that both captures have a regular number of peers contacted per hour. In the first capture, there was a peak around hour 29, which resulted in six times more peers than usual. In the second capture, there was also a peak around the 25th hour, increasing the amount of contacted peers by twenty times.

These peaks were a result of an increase of TCP Session Establishment attempts.

It is also important to state that both captures contacted around the same number of peers per hour, except at the moments when the peaks occurred.

Another analysis referred to the TCP Session Establishment phase. The results can be seen in the following pictures.

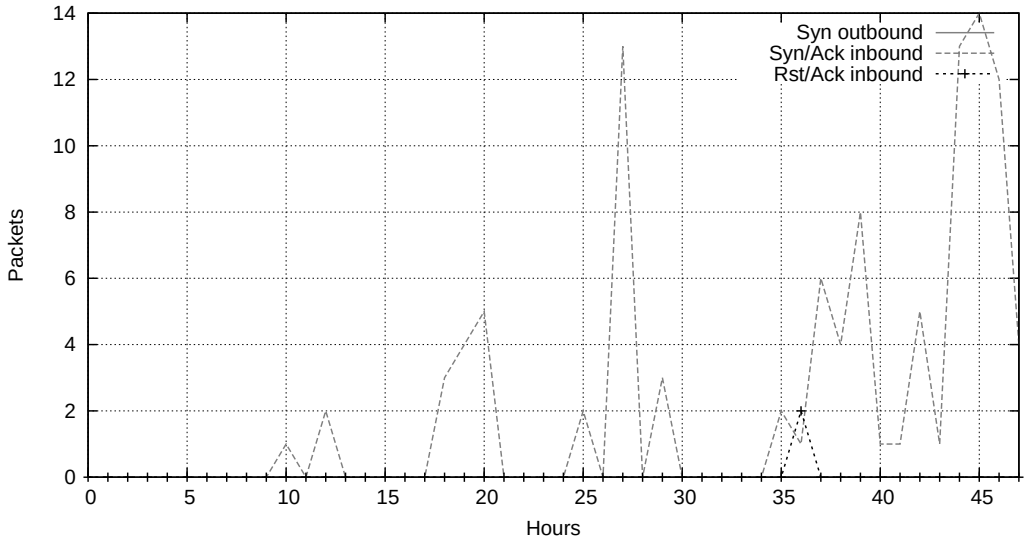

Figure 4.35: TCP Session Establishment(Outbound), Cutwail Botnet

In this first case, all the Session Establishment attempts were replied by a SYN/ACK packet, except for two packets.

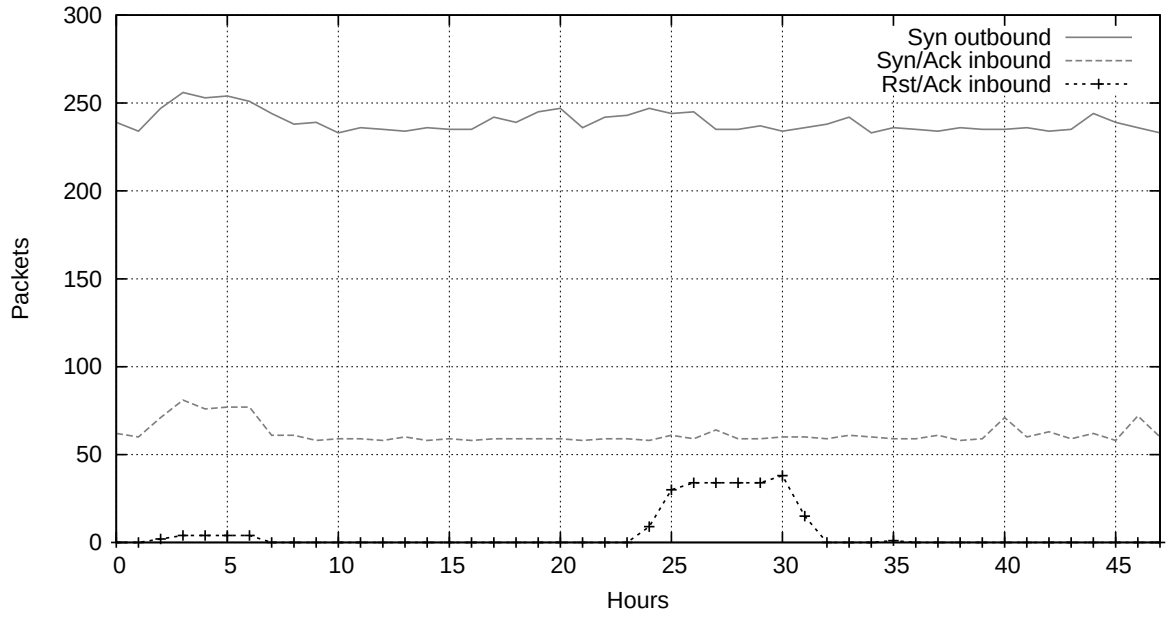

Figure 4.36: TCP Session Establishment(Outbound), Cutwail Botnet(2nd Capture)

In this picture it is clearly visible that most of the generated packets in response to SYN packets has the SYN/ACK flags active, although there were also RST/ACK packets, but in a very low number. There was however a large number of unanswered SYN packets. This is not a common behaviour and should be an alarm for Botnet activity.

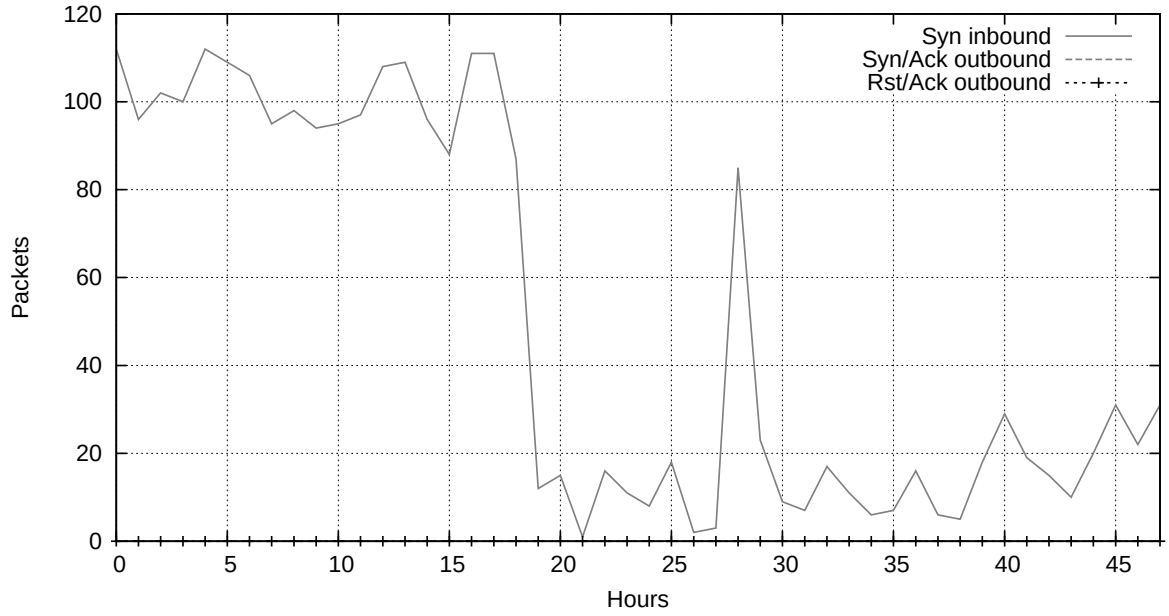

Figure 4.37: TCP Session Establishment(Inbound), Cutwail Botnet

In this case, all received SYN packets were not answered back. Like it was already explained, the infected machine in this capture only sent NBNS packets and nothing else.

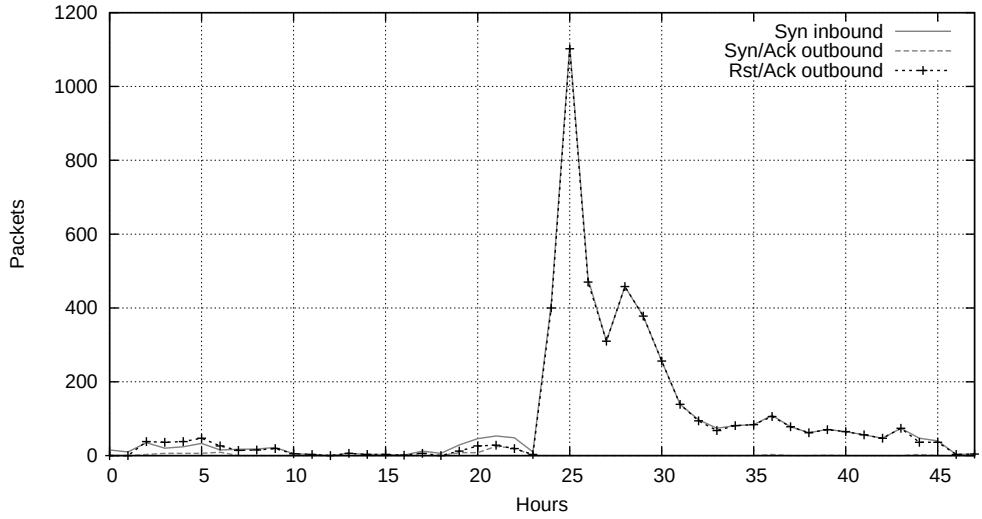

Figure 4.38: TCP Session Establishment(Inbound), Cutwail Botnet(2nd Capture)

In this last picture, we see that there are always more RST/ACK than SYN/ACK packets. Actually, almost all received SYN packets were replied with RST/ACK packets, which means that most of the session establishments attempts were not successful.

It was seen before a peak around hour 25 in the second capture, which conducted to this increase in the number of SYN and RST/ACK packets.

To finish this High-Level analysis, it is important to observe the world map that shows the location of the peers that communicated with the infected machine.

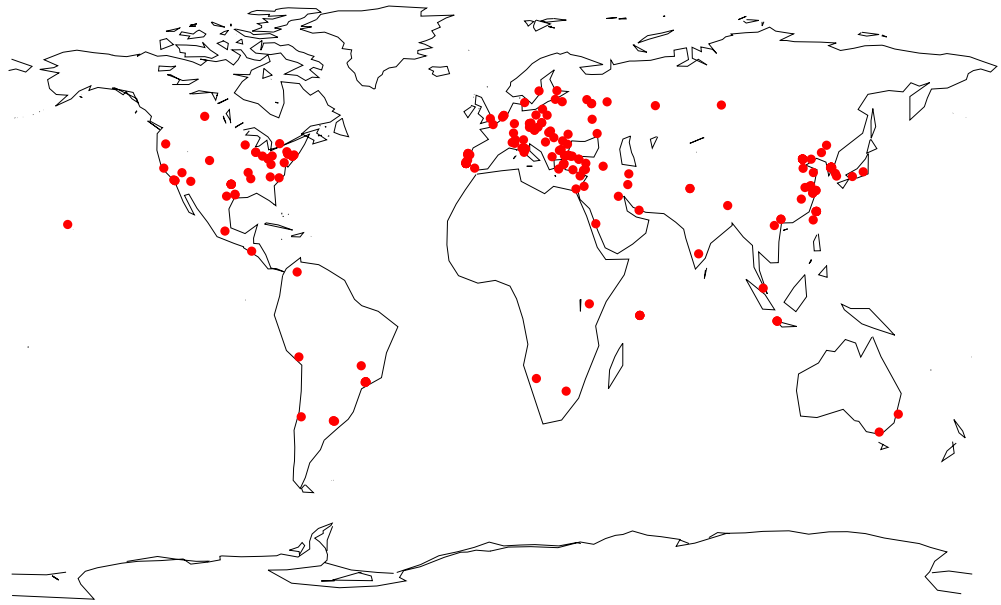

Figure 4.39: World Map, Cutwail Botnet

The Inbound and Outbound number of peers were very different, generating very different maps, so it was decided to only present the Inbound peers in the map.

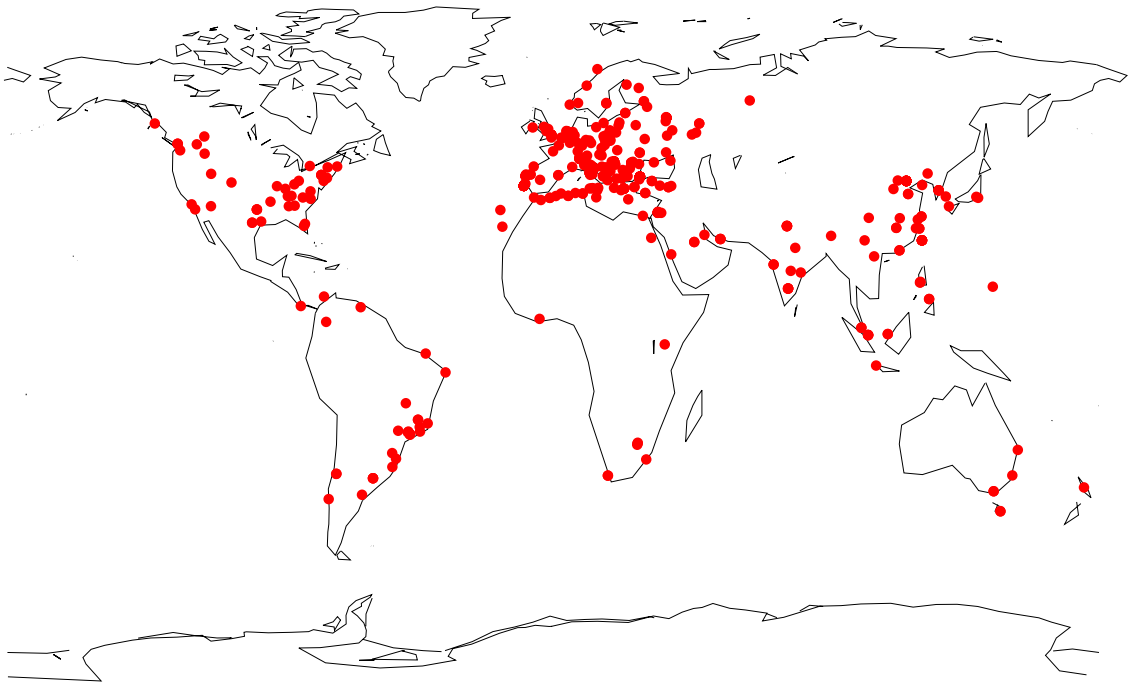

Figure 4.40: World Map, Cutwail Botnet(2nd Capture)

In these pictures, it is visible that the infected machine communicated with machines from all continents. The main infected ones are however Europe and Asia.

Unlike on the first capture, the amount of Inbound and Outbound peers is very similar, which originated very similar World Maps. So following the same criteria, we only considered Inbound peers.

4.2.3 Low-Level Analysis

Here are the Scalograms from the two captures:

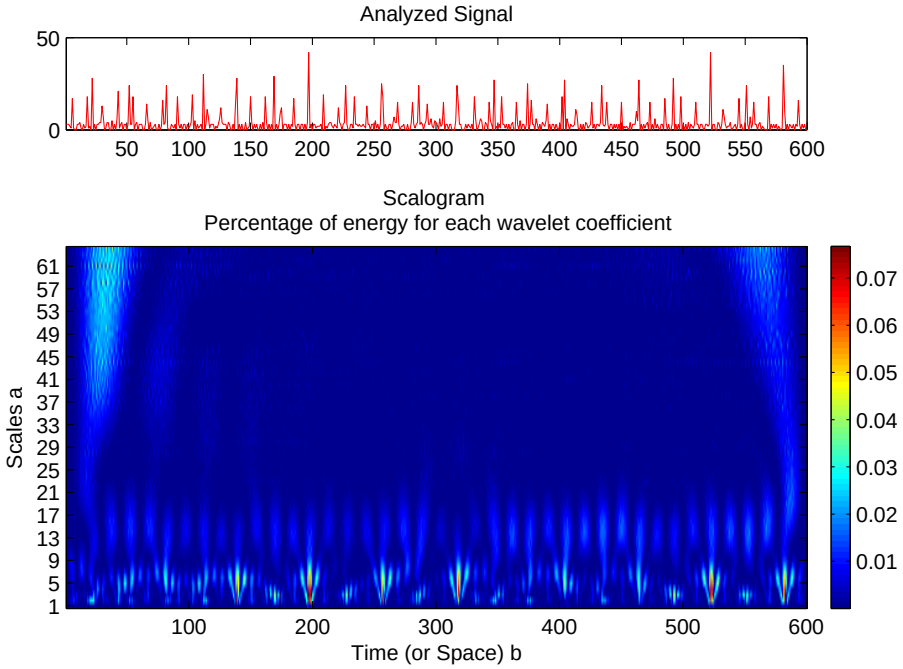

Figure 4.41: Scalogram, Cutwail Botnet

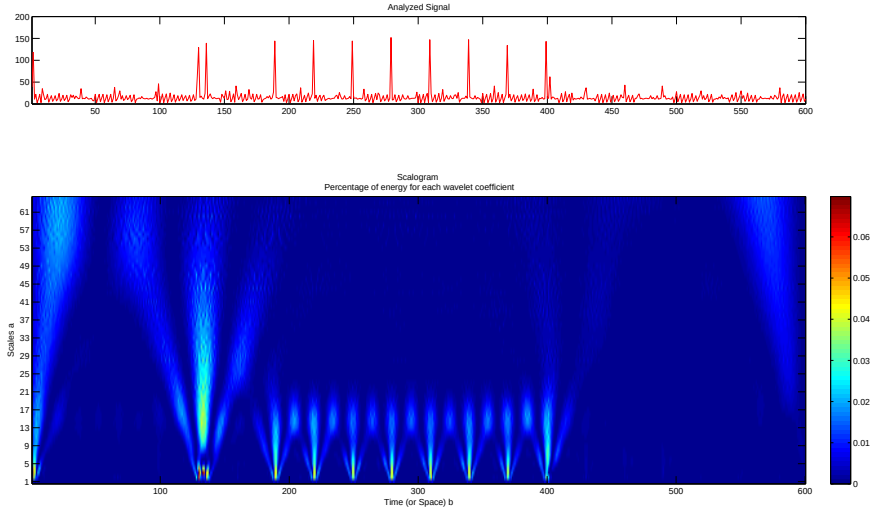

Figure 4.42: Scalogram, Cutwail Botnet(2nd Capture)

The last two scalograms are similar. Despite the fact that the first has more peaks, the second clearly has a more visible peak in the analysed signal around minute 130. This is easily explained by the high variance signal that is present in the second capture. However, both scalograms have a fairly regular behaviour, despite their different values.

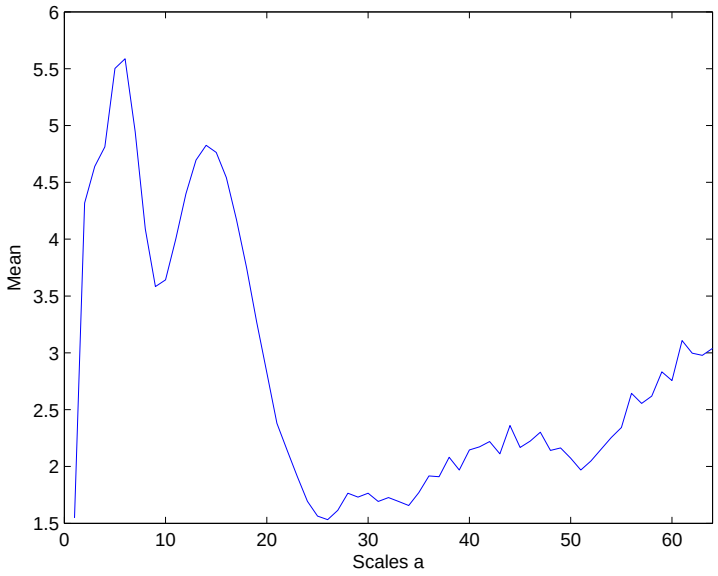

Figure 4.43: Energy Mean, Cutwail Botnet

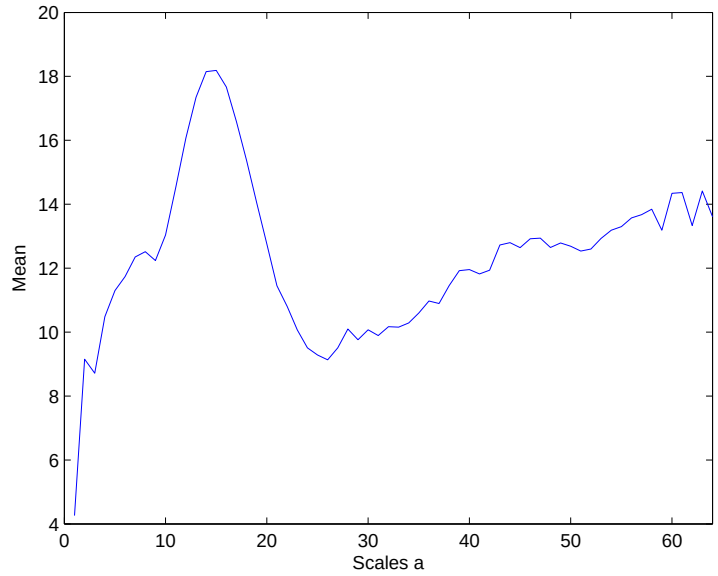

Figure 4.44: Energy Mean, Cutwail Botnet(2nd Capture)

The last pictures show that the mean values of the energy of the scalograms are quite different from each other, as expected. However, if we only observe the line behaviour, they are similar: both present an initial peak, followed by a significant decrease, and after that they start presenting a regular increase.

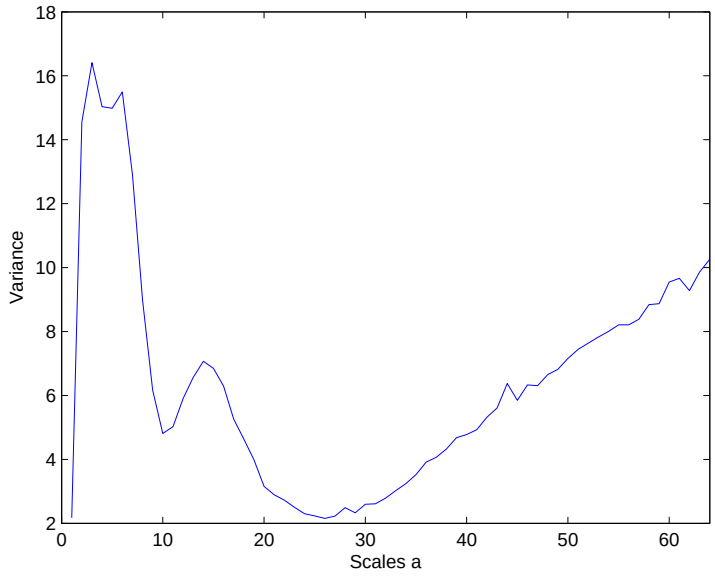

Figure 4.45: Energy Variance, Cutwail Botnet

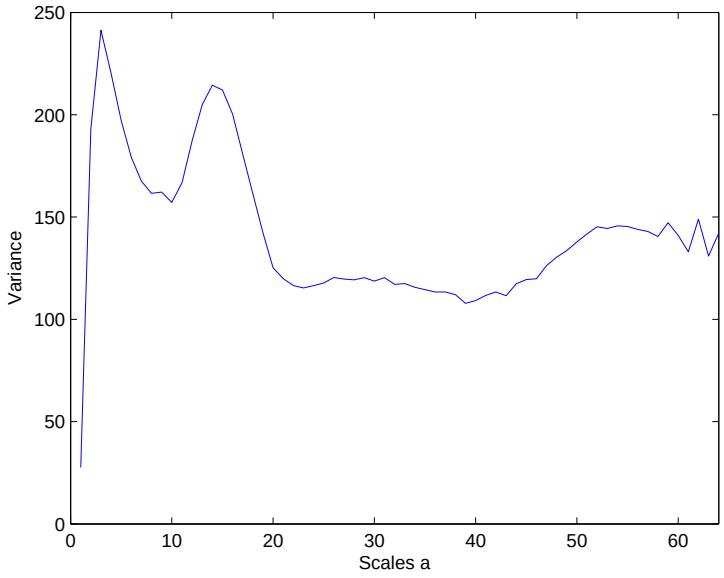

Figure 4.46: Energy Variance, Cutwail Botnet(2nd Capture)

The variance analysis shows very different results. In the first case, we can observe that the maximum value is around 16, while in the second case it is close to 250. This obviously leads to very different graphs.

4.3 Bobax

The malware used from this Botnet was downloaded from [35] on July 2011. The rest of the analysis process followed the same conditions of the previous Botnets analysis.

The traffic obtained in the second capture did not reveal any relevant changes when compared to the first capture, so it was not included in this document. It is important to point out that this capture was the one that generated more traffic.

4.3.1 General analysis

The traffic from Bobax followed the same behaviour throughout the whole duration of the capture. Right away after the malware was installed, a lot of DNS queries were exchanged in port 1042, known for being used by trojans. Actually, many of these queries were actually under the Unknown UDP label that will be shown next. In this capture, we also observed a lot of SMTP packets (only in 1st hour). Most of them were under the Unknown TCP label. Some HTTP packets were also exchanged, and sporadically some HTTP/XML Notify messages. HTTP was the second most used protocol of the Unknown TCP packets. However, these packets were mostly SMB packets. Around 400 thousand SMB packets were exchanged per hour. It was also possible to observe some NBNS packets.

Unknown UDP packets were once again mainly used for DoS attacks, using ports that are known to be used for that type of attacks.

This Botnet definitely behave like expected, considering its amount of HTTP traffic, DNS lookups, DoS attacks and, essentially, SMTP packets.

4.3.2 High-Level Analysis

Once again, the protocols involved in the generated traffic were analysed.

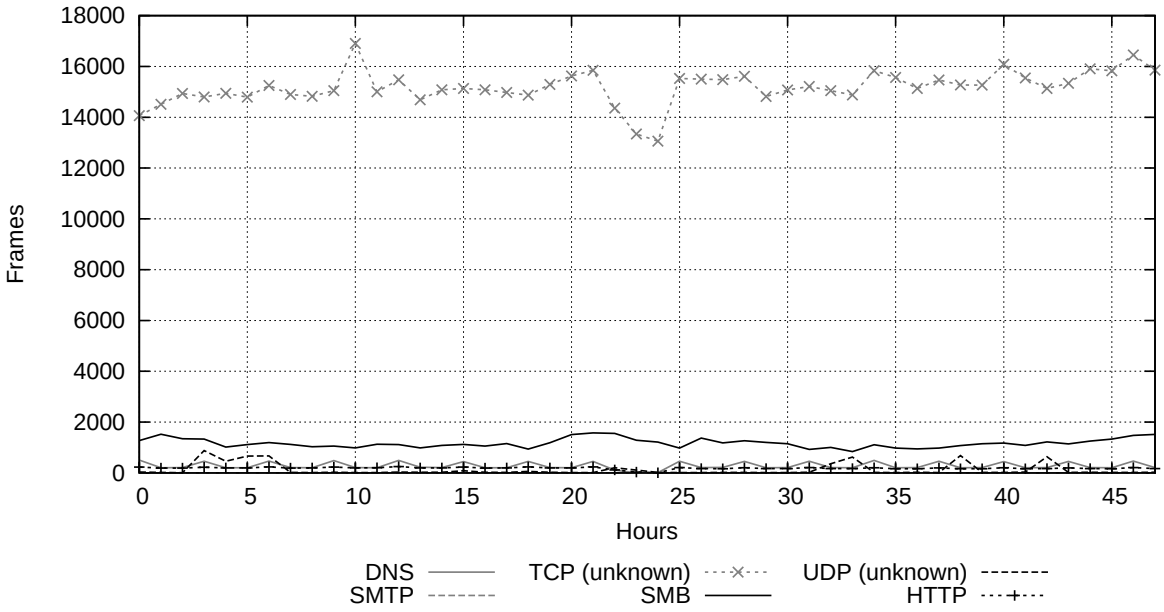

Figure 4.47: Protocols(Upload), Bobax Botnet

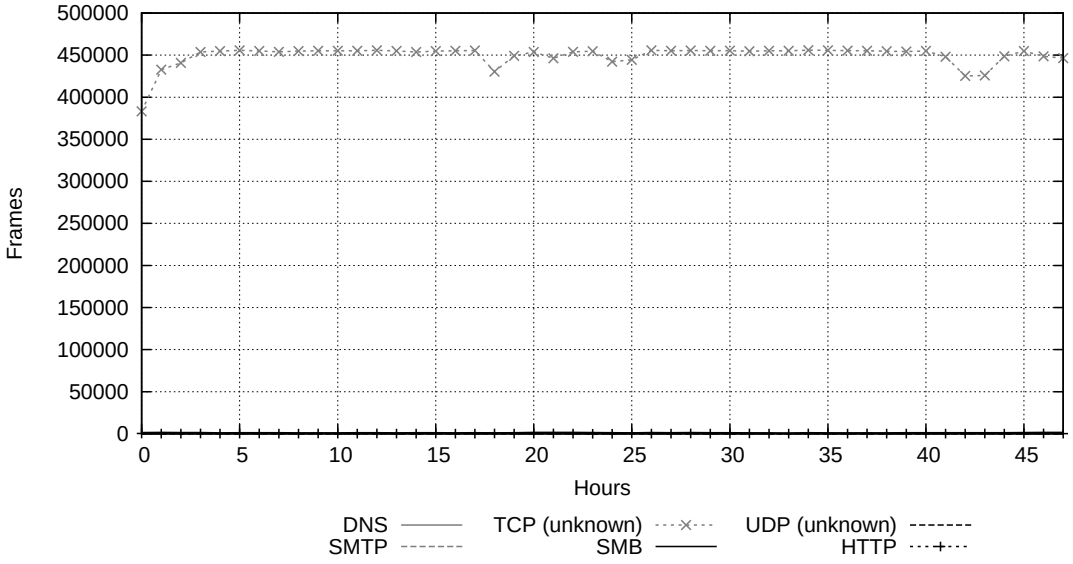

Figure 4.48: Protocols(Download), Bobax Botnet

In these two pictures, it is visible that most of the generated traffic was filtered as unknown TCP. The Upload picture, despite showing only Unknown TCP traffic, also contains traffic from all the other protocols, although in a much lower quantity.

In the Download picture, it is possible to see a clear pattern in DNS, SMB, SMTP, Unknown UDP and HTTP packets. Again, they have relatively small numbers when compared to the number of Unknown TCP packets.

As explained before, the vast majority of Unknown TCP packets are SMB packets, although there are also HTTP packets and some packets from other protocols. Unknown UDP packets are mostly DNS packets or packets used for DoS attempts.

Next, the number of packets generated per hour and per minute was also analysed.

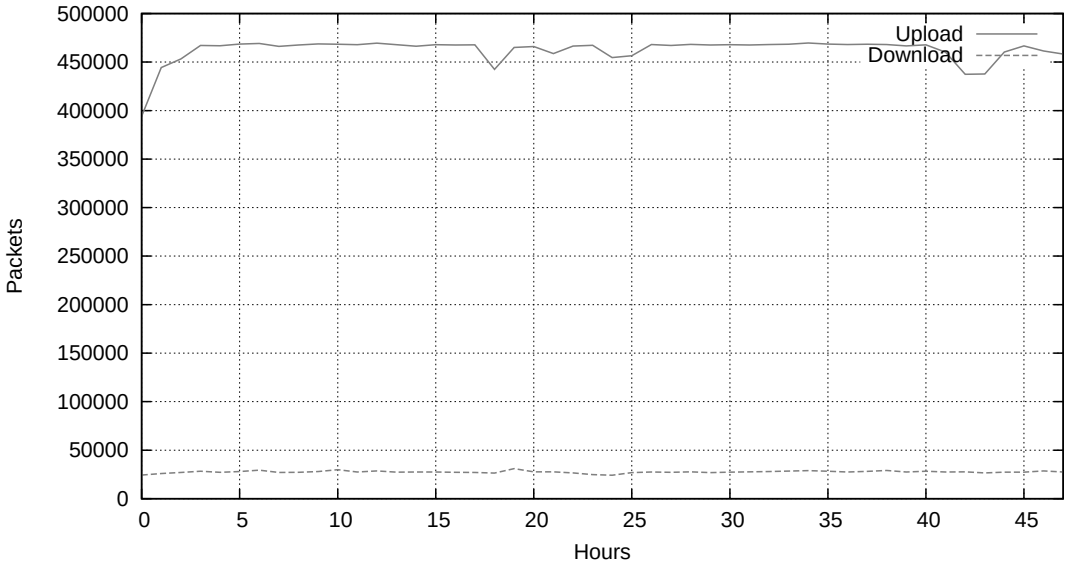

Figure 4.49: Packets per hour, Bobax Botnet

The amount of packets per hour observed in the previous picture is a clear sign that we are facing Botnet generated traffic. This should be an instant warning to take measures to protect the infected machine. It is also important to stress the difference between the number of Upload and Download packets. The amount of Upload packets are in the order of 470 thousand packets per hour, while Download packets are in the order of 40 thousand packets. There are no significant peaks on the traffic generated that should be pointed out.

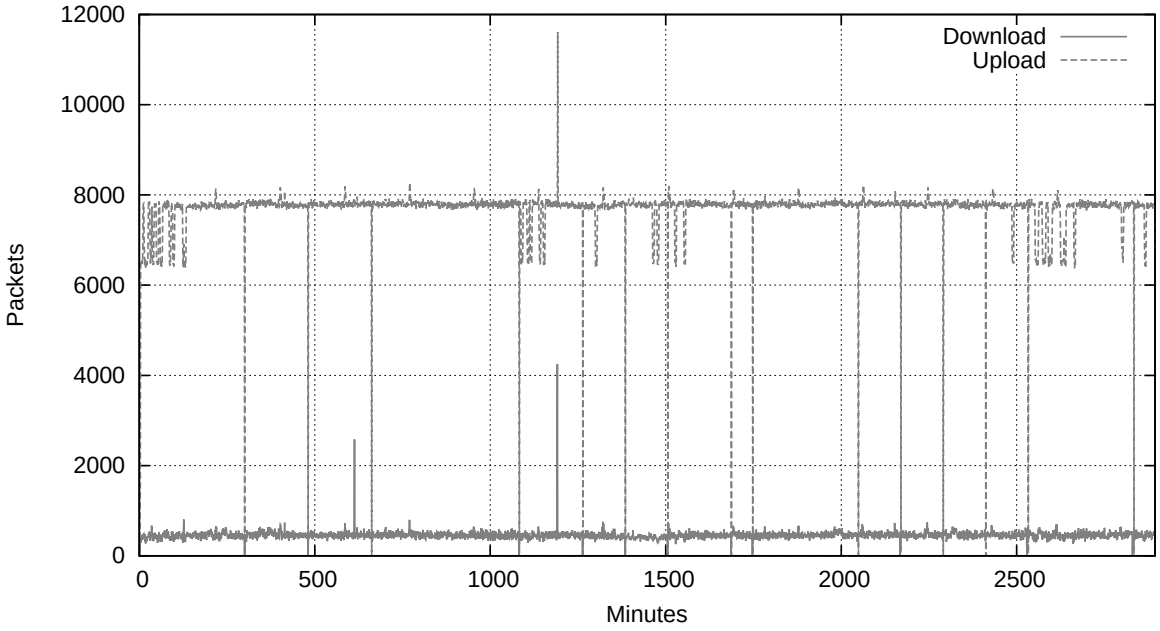

Figure 4.50: Packets per minute, Bobax Botnet

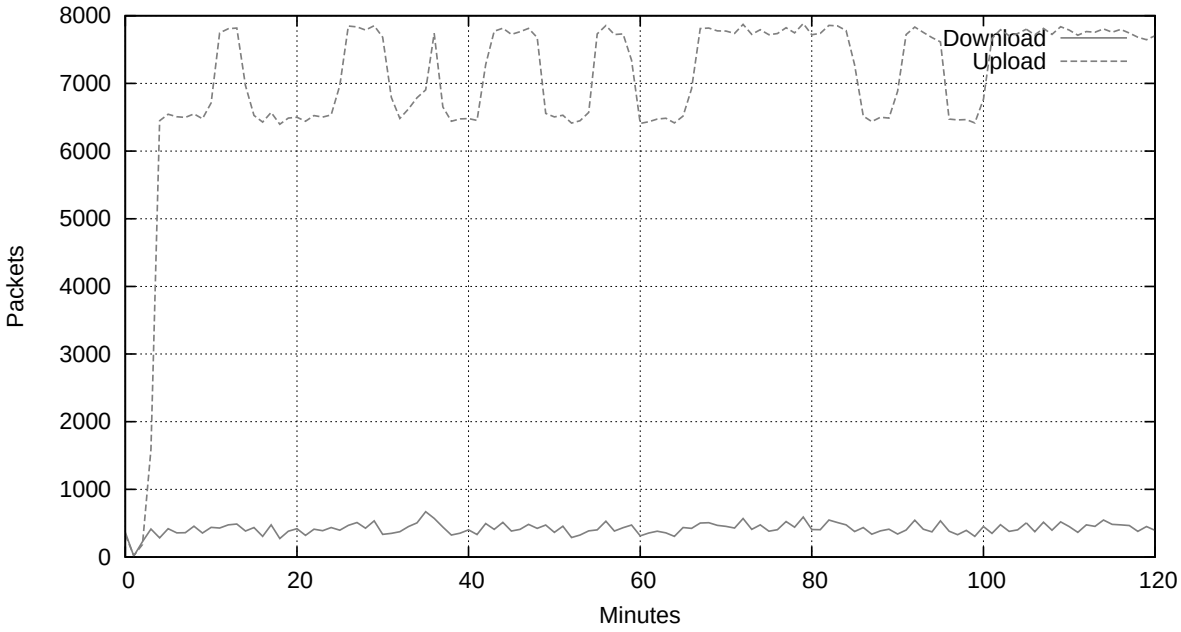

Figure 4.51: Sample of packets per minute, Bobax Botnet

From the previous pictures it is possible to observe that the number of packets per minute still have a visible pattern, specially for Download packets. This Botnet generates more than eight thousand packets per minute.

The next step consisted in calculating the amount of generated data, from the captured packets packets. Following the same procedures that were used before, we obtained the statistics that are shown next.

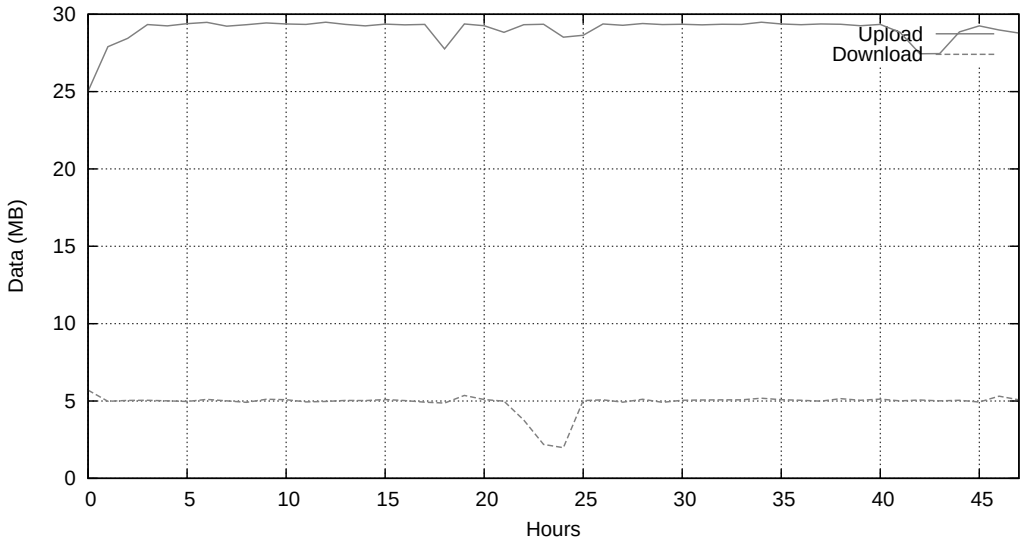

Figure 4.52: Amount of traffic per hour, Bobax Botnet

The previous picture shows that, despite the huge difference that exists in the number of Uploaded and Downloaded packets, the difference in the amount of traffic is not so significant. Uploaded packets are responsible for a rate of around 30MB per hour, while Downloaded packets generate around 5MB per hour. Again, there are no significant peaks in the amount of generated traffic, which results in a clear typical behavioural pattern.

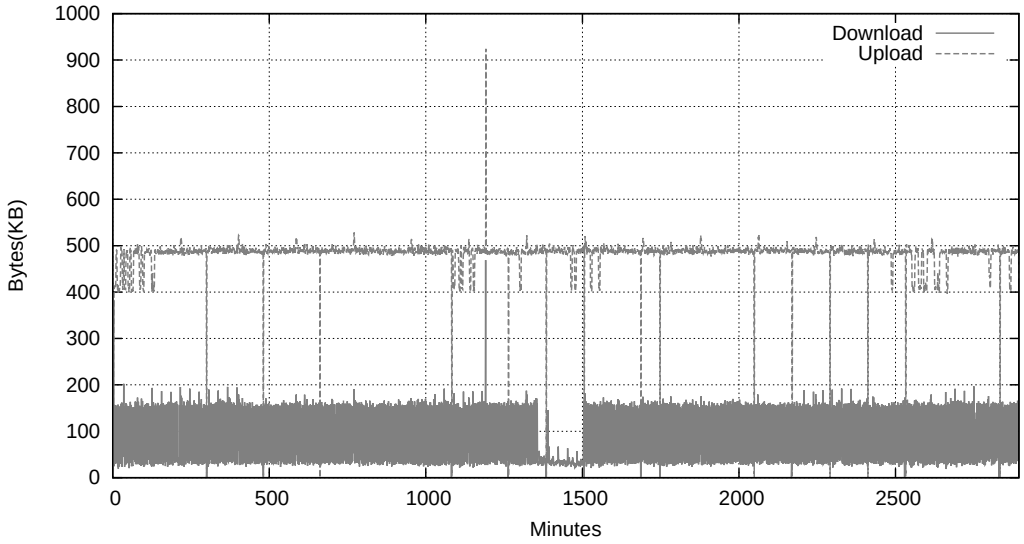

Figure 4.53: Amount of traffic per minute, Bobax Botnet

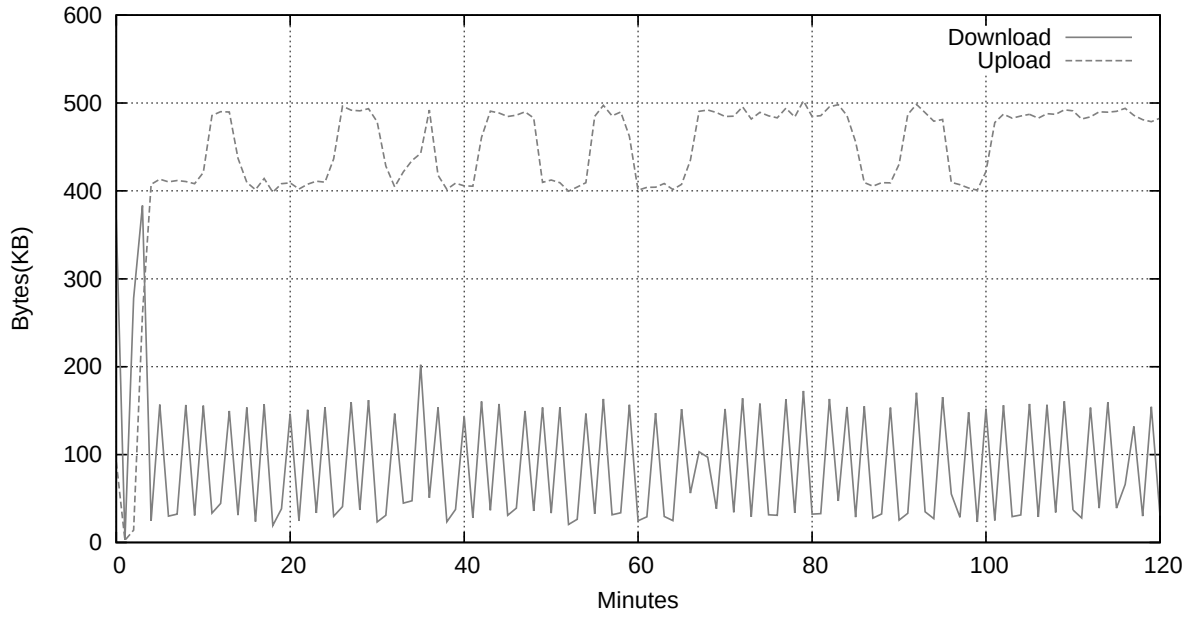

Figure 4.54: Sample of amount of traffic per minute, Bobax Botnet

When observing the amount of traffic per minute, it is harder to get a clear pattern due to the variances that exist over time. The amount of generated data is not significant, being around 600KB per minute. So, an analysis exclusively based on this statistic would not raise much suspicions, except for the regularity that is observed.

Let us now analyse the amount of unique peers contacted over time, which can be seen in the next picture.

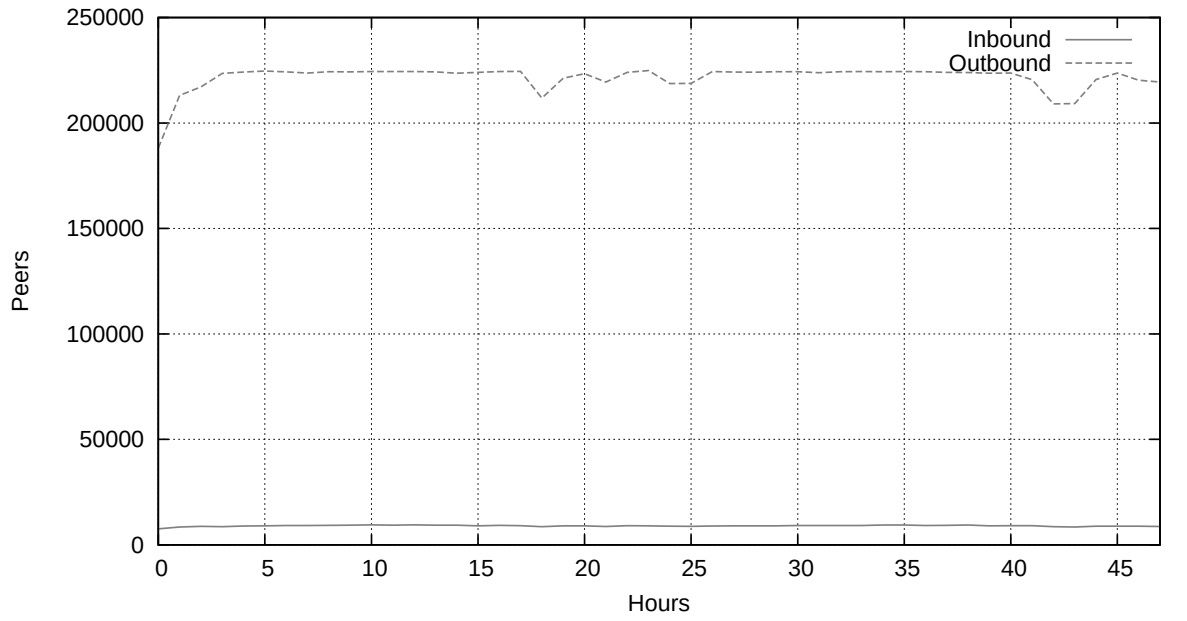

Figure 4.55: Unique peers per hour, Bobax Botnet

Even in the number of contacted peers, we have a clear behavioural pattern. The observed values raise suspicions about Botnet infection, because they are in the order of 225 thousand peers per hour.

The next procedure consists of analysing the TCP Session Establishment attempts. This analysis can be seen in the following pictures.

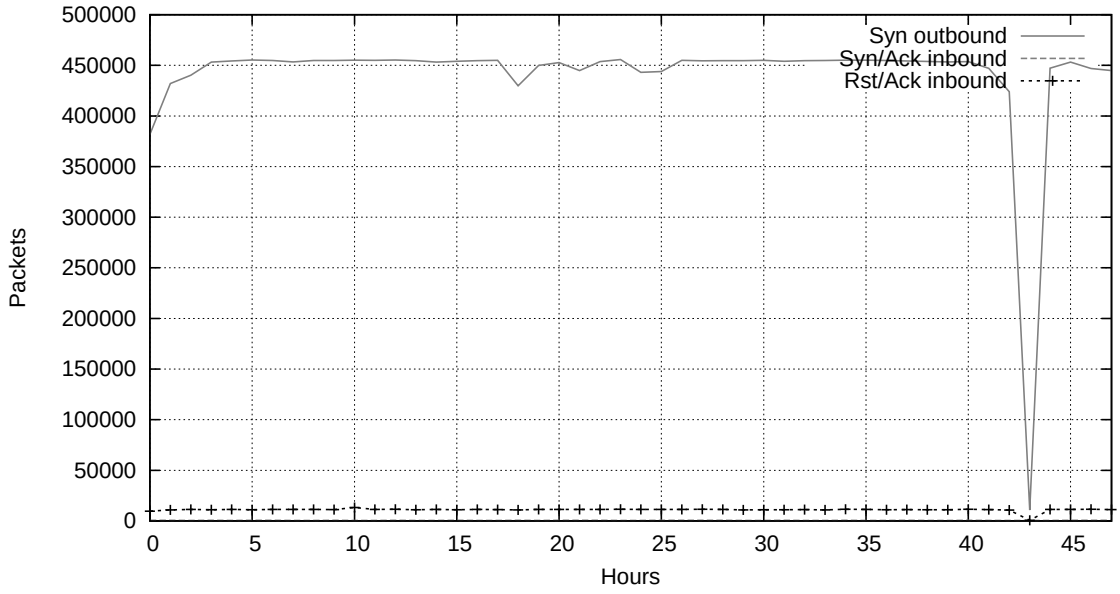

Figure 4.56: TCP Session Establishment(Outbound), Bobax Botnet

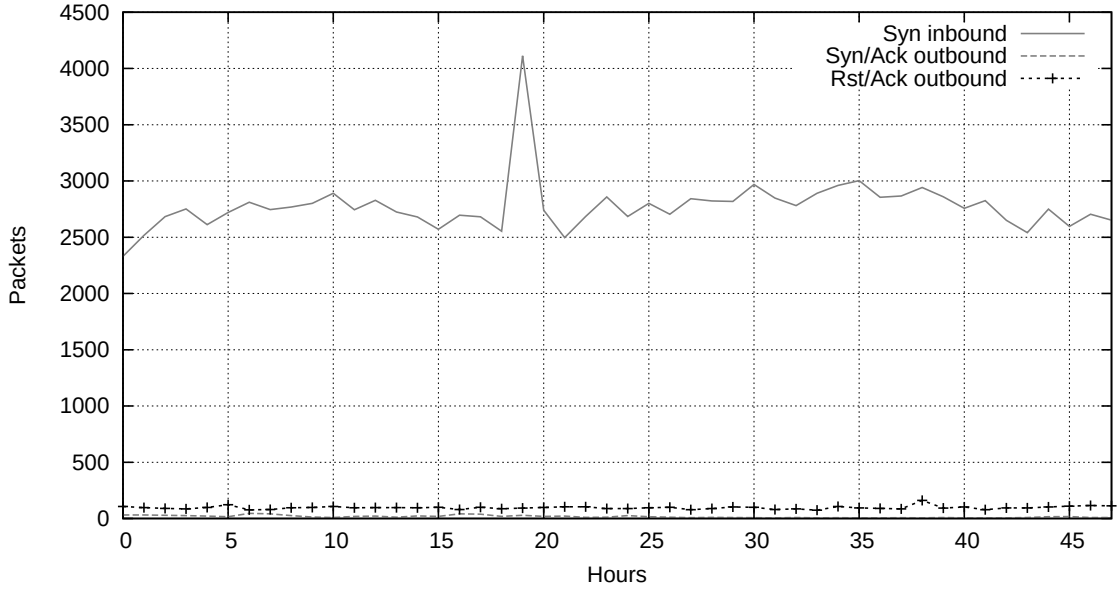

Figure 4.57: TCP Session Establishment(Inbound), Bobax Botnet

From the previous pictures we can see that most of the SYN packets did not obtain any reply. Only a small number was replied with the RST/ACK flags set, and an even smaller number with the SYN/ACK flags.

To conclude this interesting Botnet High-Level analysis, it is interesting to see a world map showing the location of the peers that communicated with the infected machine.

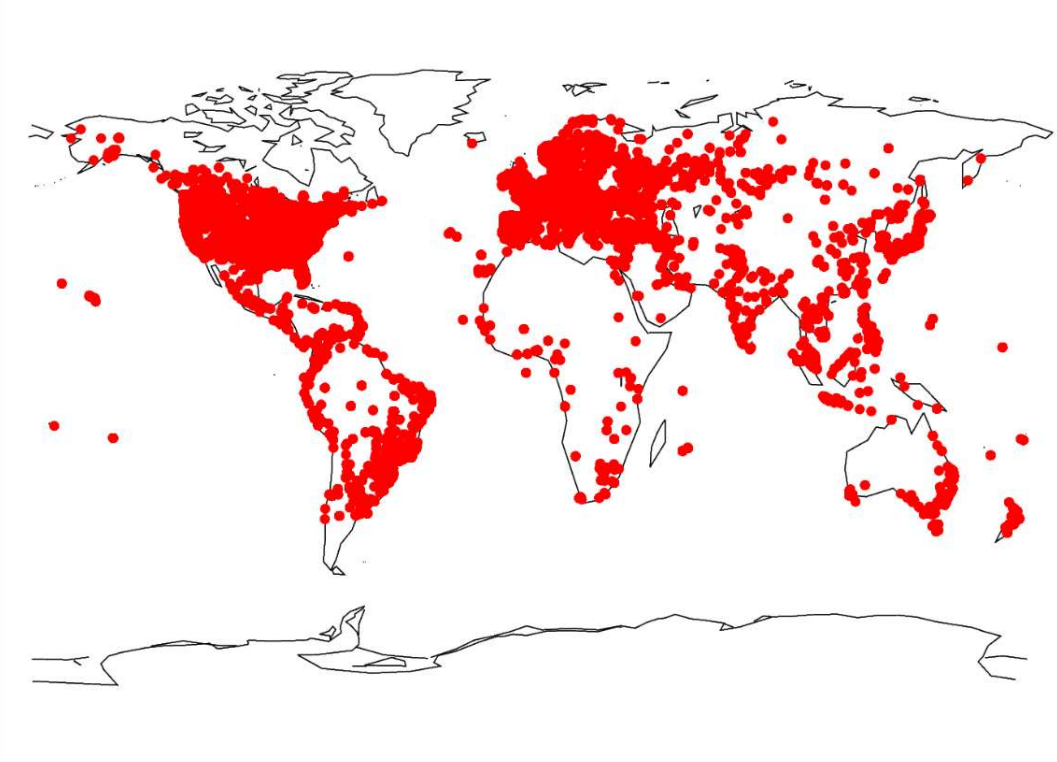

Figure 4.58: World Map, Bobax Botnet

Bobax's infected machines are located everywhere in the world, although we can defend that the most infected continents are Europe and America. The countries that suffer more infections are the United States of America and China.

In this case, as it was seen in the number of peers, there are many more Outbound than Inbound connections. This, as expected, lead to two different World Maps, resulting in a higher amount of black dots, representing the Outbound peers. In order to observe the differences, we will present a World Map that considers Outbound peers.

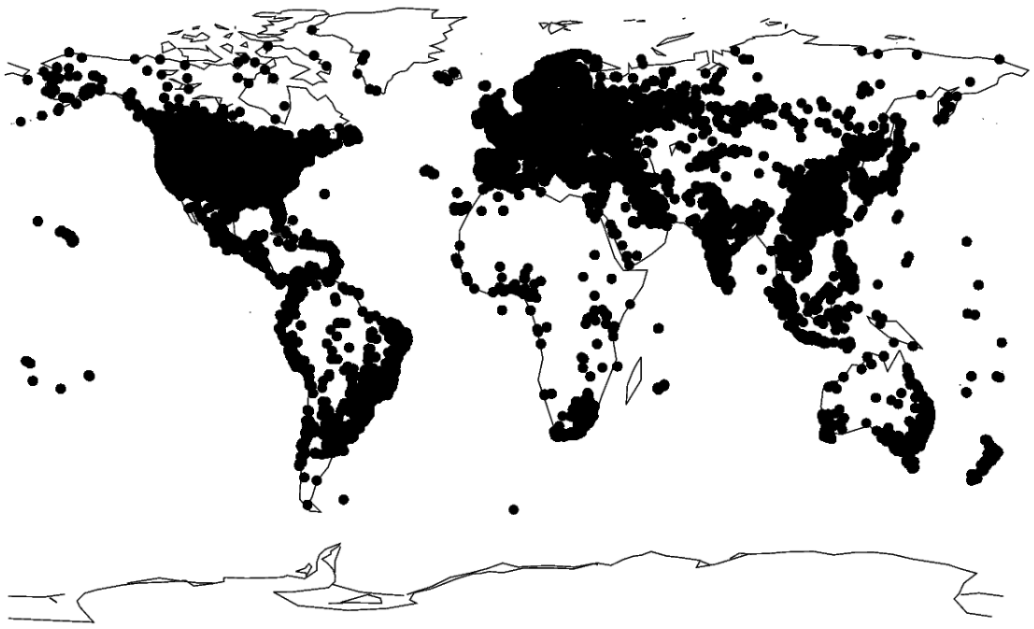

Figure 4.59: World Map, Bobax Botnet(Outbound Connections)

4.3.3 Low-Level Analysis

The scalogram is shown in the next Figure. next,

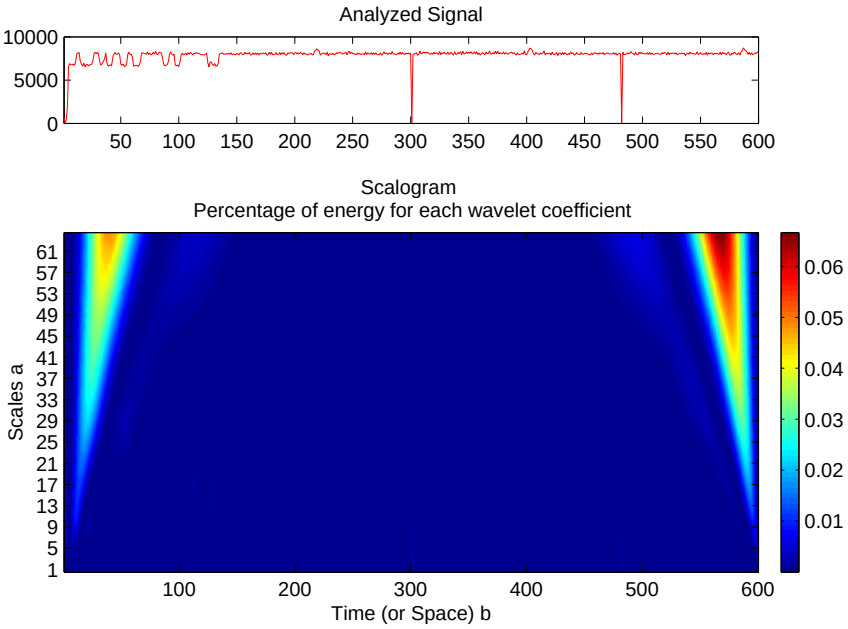

Figure 4.60: Scalogram, Bobax Botnet

This scalogram shows that there are high percentages of energy close to the beginning and the end of the analysed signal. Besides those two points, and as it has been confirmed on the high level analysis, the signal is very similar over time with very few peaks. So, this graph shows the low variance of the signal, except for the higher scales.

Analysing now the mean and the variance of the energy, we obtained the following results:

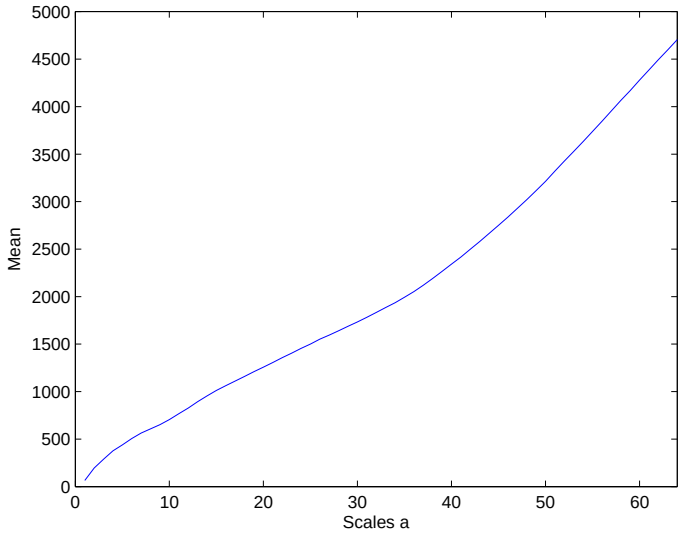

Figure 4.61: Energy Mean, Bobax Botnet

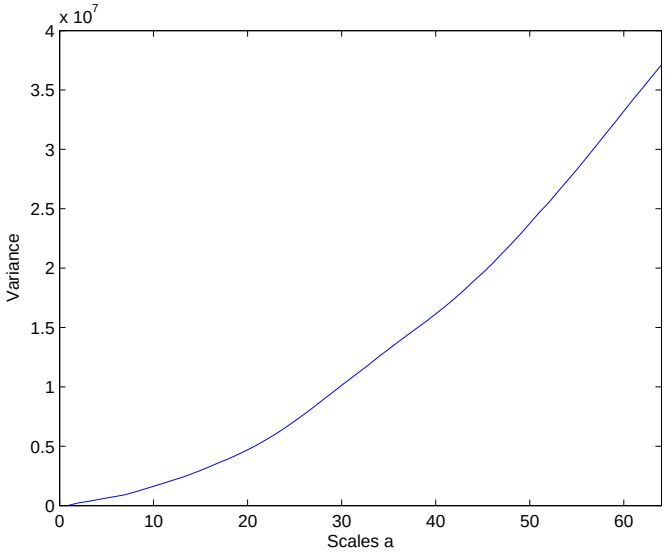

Figure 4.62: Energy Variance, Bobax Botnet

Both metrics present a similar behaviour, always showing an increase as the scale increases.

4.4 Lethic

Traffic was captured during a 48 hours period, in order to make a deep analysis of this Botnet behaviour. The malware used was downloaded from [35] on September of 2011.

Like Bobax, the traffic obtained in the second capture was pretty much similar to the traffic of the first one, so once again it was not considered in this document.

4.4.1 General analysis

After the malware was installed, a significant number of DNS queries started to be exchanged. Then many NBNS queries start being exchanged regularly. They use port 137, which has been reported to be used by the trojan Msinit. Unknown TCP traffic, which will be presented in the next subsection, is actually mostly composed by HTTP packets using the alternate port 8090. There are also some DHCP packets exchanged, informing and acknowledging. Some HTTP packets that were attempting to change the Certificates List were also detected, and some were followed by a packet containing info for the Certificate Revocation List. Besides, among Unknown TCP packets, some SMB packets were also discovered, as well as packets going through port 6000, which is a port usually used by virus/trojans. Unknown UDP packets are suspected to be used for DoS attacks, because they use ports between 33435 and 33438, which are typically used for this type of attacks. Finally, only a couple of SMTP packets were generated.

We can then conclude that this Botnet, despite not having generated too much SMTP packets, unlike it was expected, generated a significant amount of HTTP packets, including attempts for Certificate List changes. The number of NBNS packets also surpassed the expectations. The rest of the results are as expected, except the total amount of generated packets.

4.4.2 High-Level Analysis

The protocols involved in the traffic generated by this Botnet can be seen in the next figures.

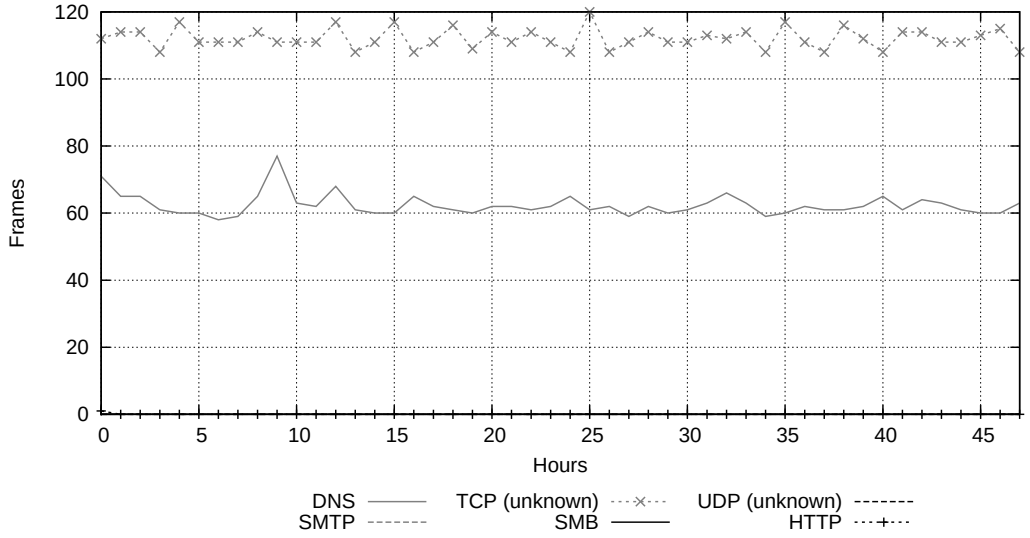

Figure 4.63: Protocols(Upload), Lethic Botnet

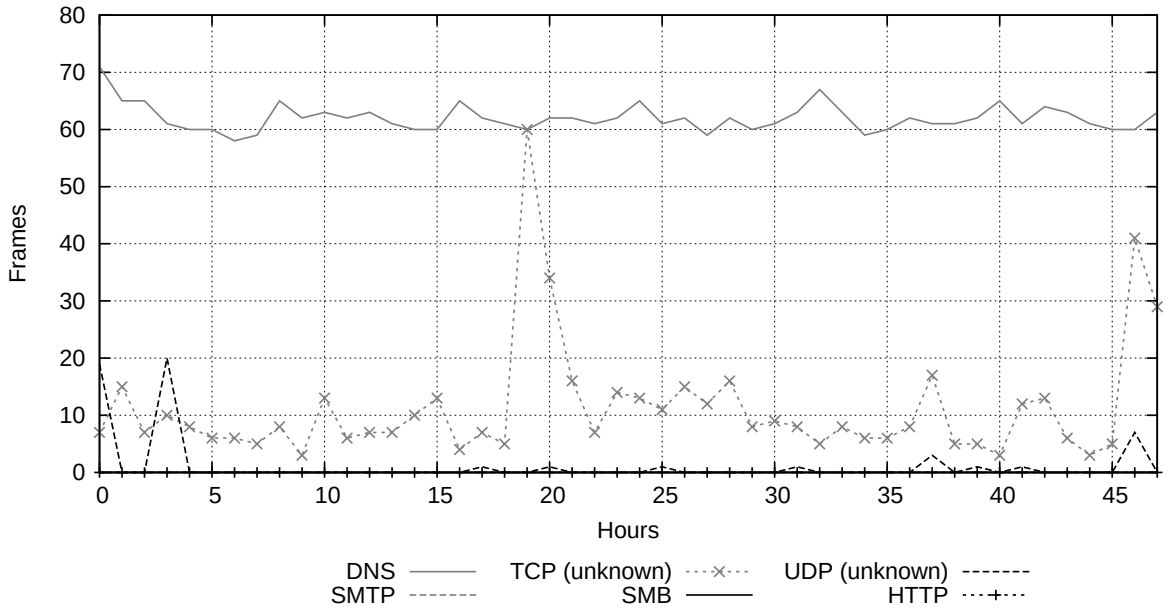

Figure 4.64: Protocols(Download), Lethic Botnet

In this capture we have two major protocols involved, DNS and Unknown TCP. In the Upload direction, we have also some Unknown UDP packets, and the amount of DNS packets is higher than the amount of Unknown TCP packets. When analysing the number of Download packets, this situation reverses.

As explained in the General Analysis, Unknown TCP packets are mostly HTTP packets, as well as SMB packets.

Let us now analyse the number of packets generated per hour and per minute.

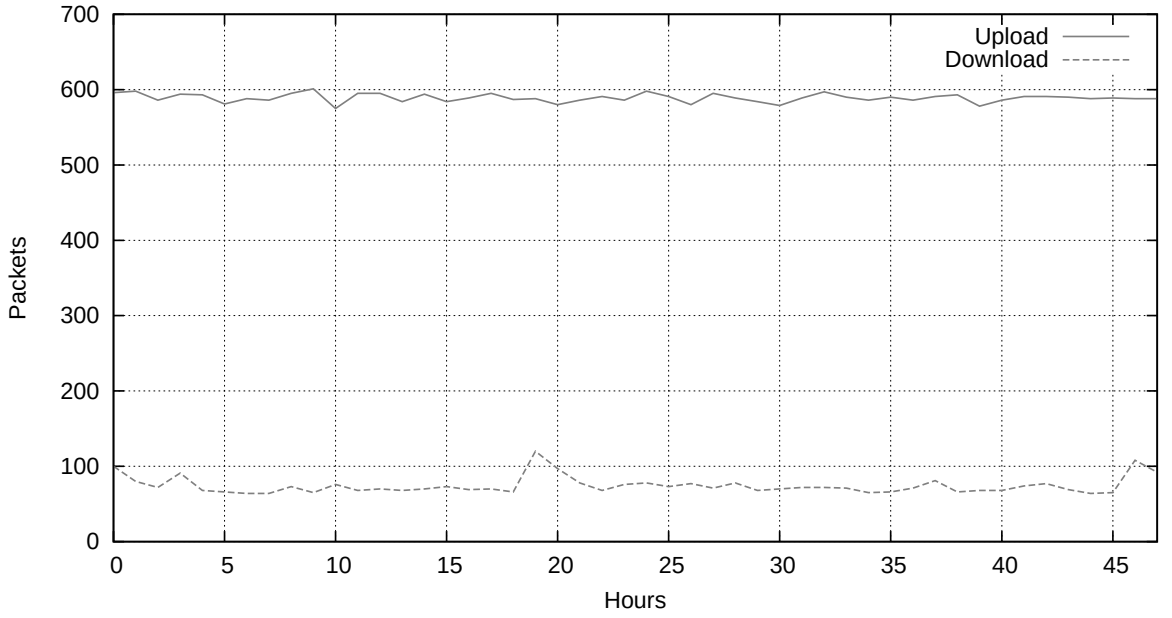

Figure 4.65: Packets per hour, Lethic Botnet

The previous picture presents a simple pattern in the number of packets per hour. The amount of Uploaded packets is around six times higher than the amount of Downloaded packets. There are not any relevant changes in the amount of generated traffic that worth to be pointed out.

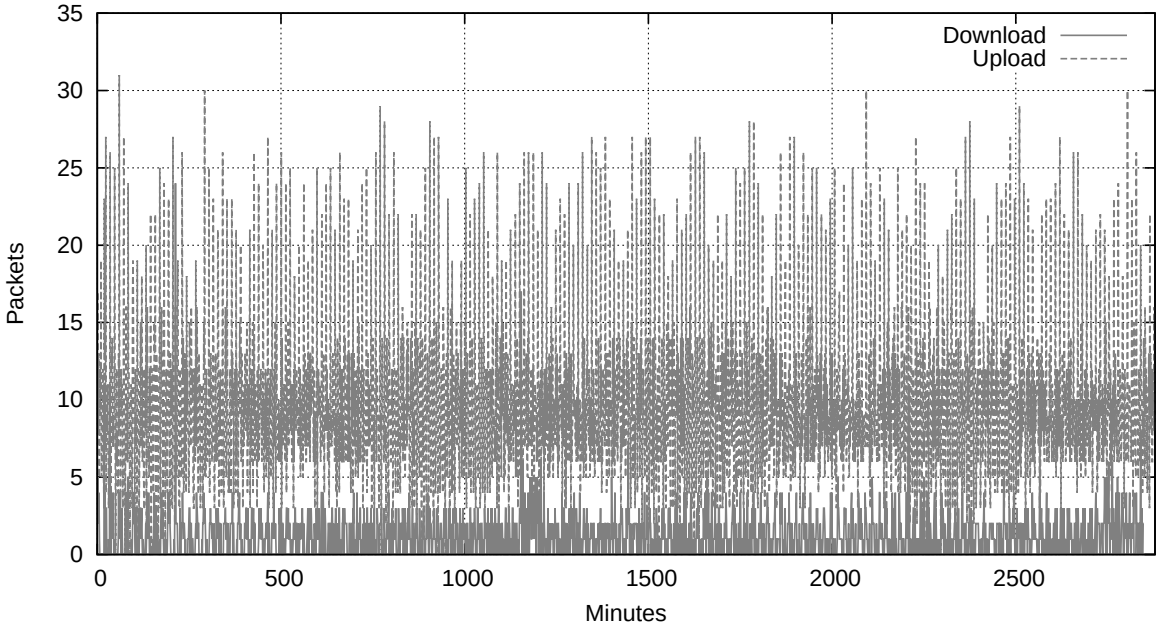

Figure 4.66: Packets per minute, Lethic Botnet

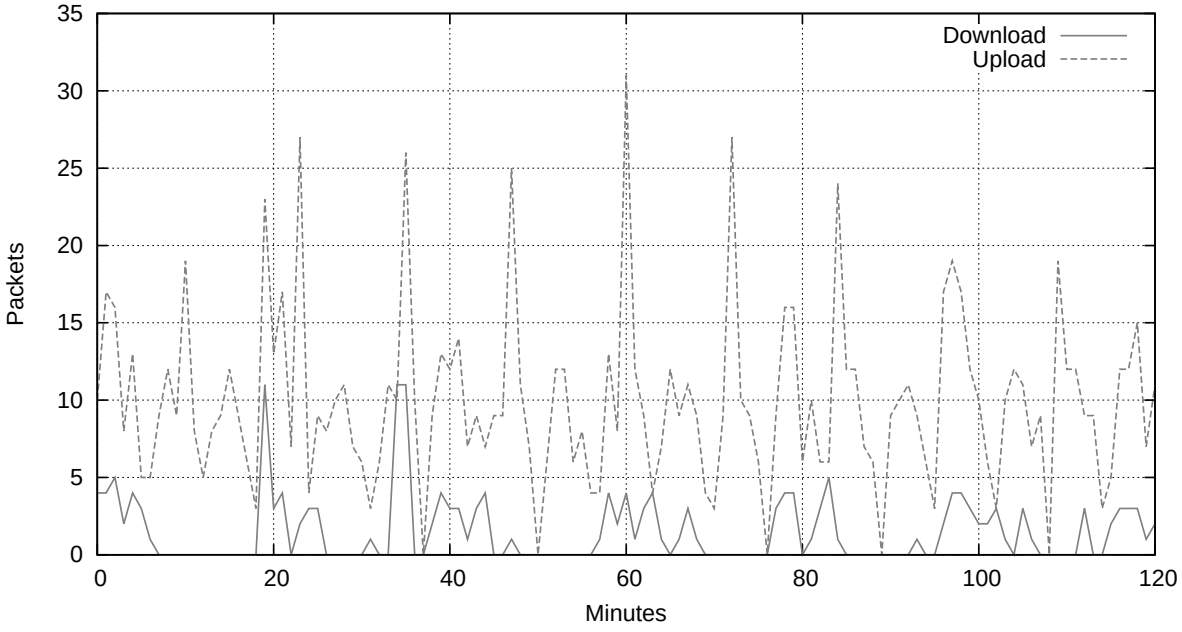

Figure 4.67: Sample of packets per minute, Lethic Botnet

From the last two pictures, the variance in the Botnet behaviour becomes more perceivable. The numbers presented are not enough to raise any suspicions that we are facing a possible Botnet infection.

From the captured packets we calculated the amount of generated data. Following the same criteria of the previous procedures, we obtained the statistics shown in the next page.

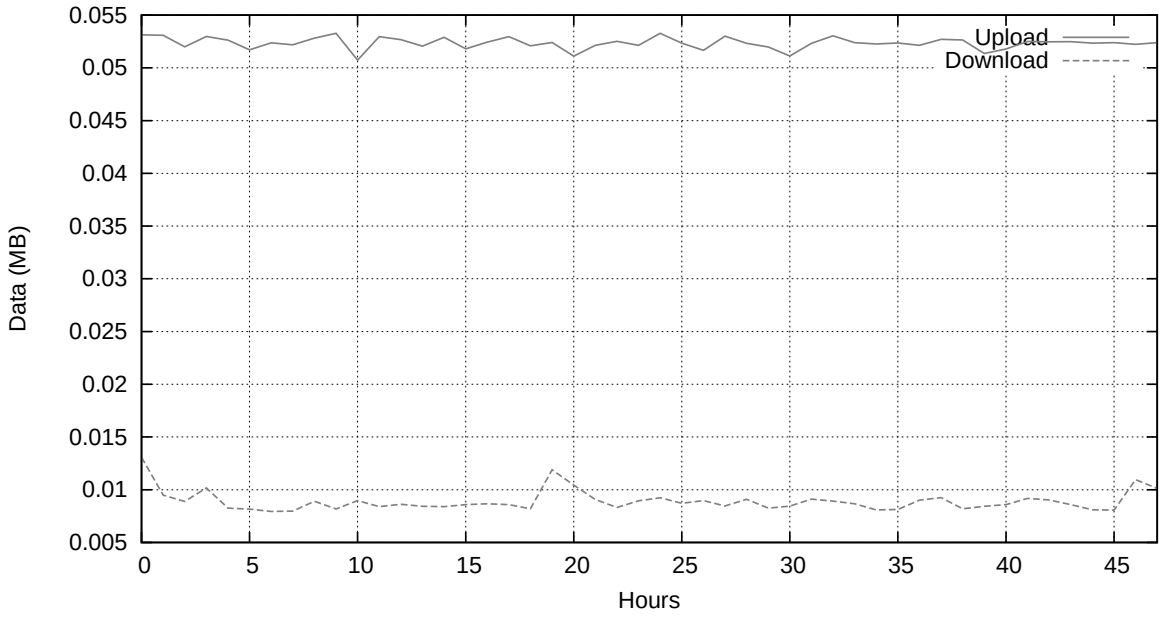

Figure 4.68: Amount of traffic per hour, Lethic Botnet

This picture is pretty similar to the one corresponding to the amount of packets per hour. As expected, the difference between Upload and Download packets is maintained and the ratio between them is approximately the same. The same simple pattern can be observed.

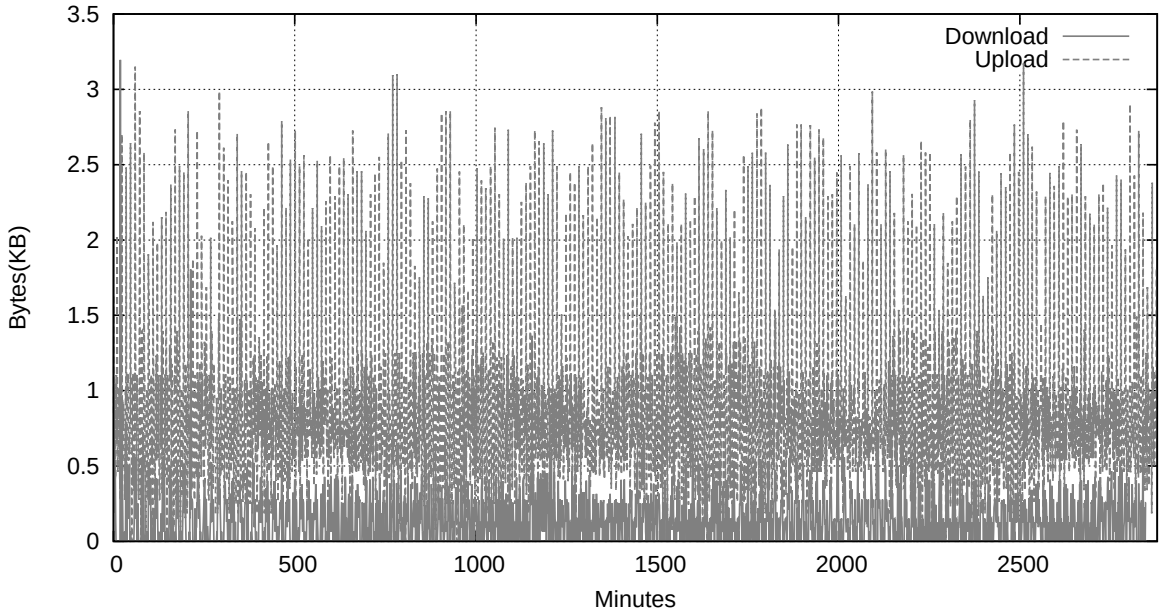

Figure 4.69: Amount of traffic per minute, Lethic Botnet

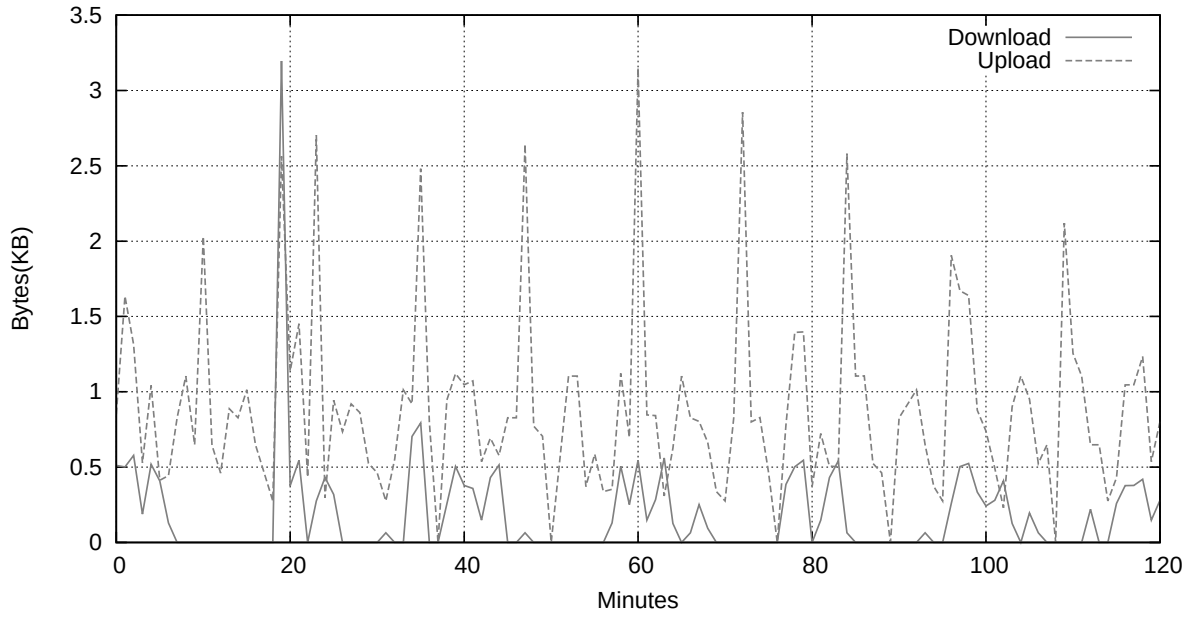

Figure 4.70: Sample of amount of traffic per minute, Lethic Botnet

Once again, this is pretty similar to the amount of packets per minute. This procedure, by itself, could not raise any suspicion since it generated around 1.5KB per minute, which is a very low value.

The amount of unique peers contacted over time revealed interesting values. This can be seen in the next picture.

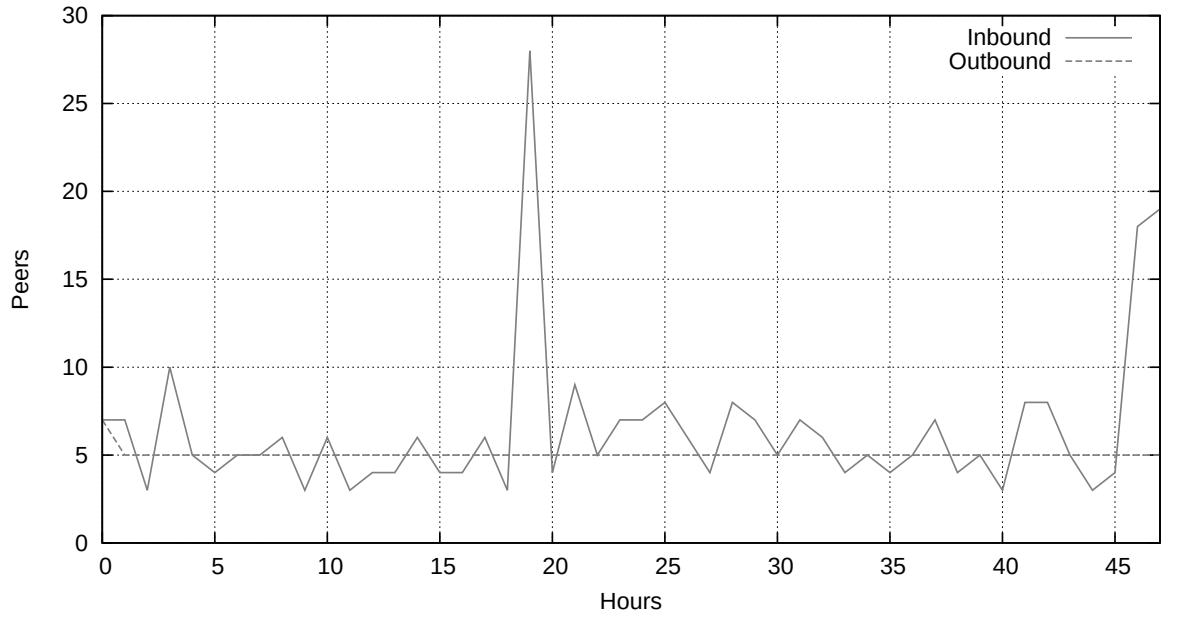

Figure 4.71: Unique peers per hour, Lethic Botnet

Despite having a rate of around 5 peers per hour, there is a peak around hours 20 and 47. The peaks, as it was seen in the Protocols details, happen because of the increase of Unknown TCP packets using port 8090, therefore HTTP packets.

The next procedure consisted of analysing the TCP Session Establishment attempts. The results can be seen in the next figure.

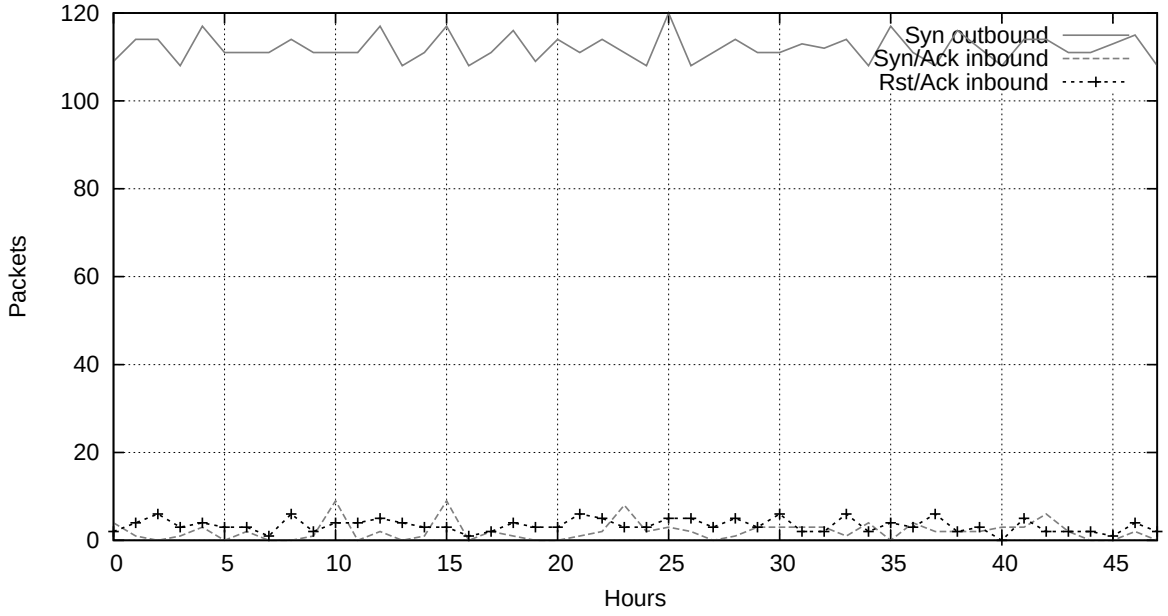

Figure 4.72: TCP Session Establishment(Outbound), Lethic Botnet

From the previous picture we can say that most of the SYN packets did not obtain any reply. Only a small number replied with the RST/ACK flags active or with the SYN/ACK flags active.

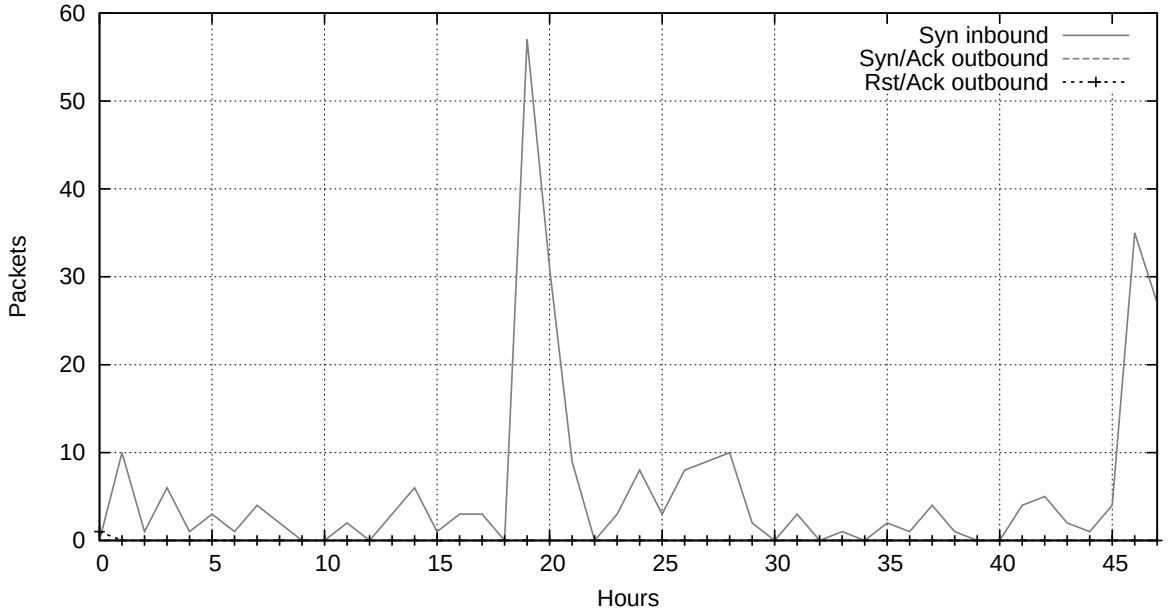

Figure 4.73: TCP Session Establishment(Inbound), Lethic Botnet

In this case, most of the received SYN packets have not been replied back. This reveals an anomaly in the communication, meaning that we are facing a Botnet infection.

To conclude this analysis, let us look at the world map showing the location of the peers that communicated with the infected machine.

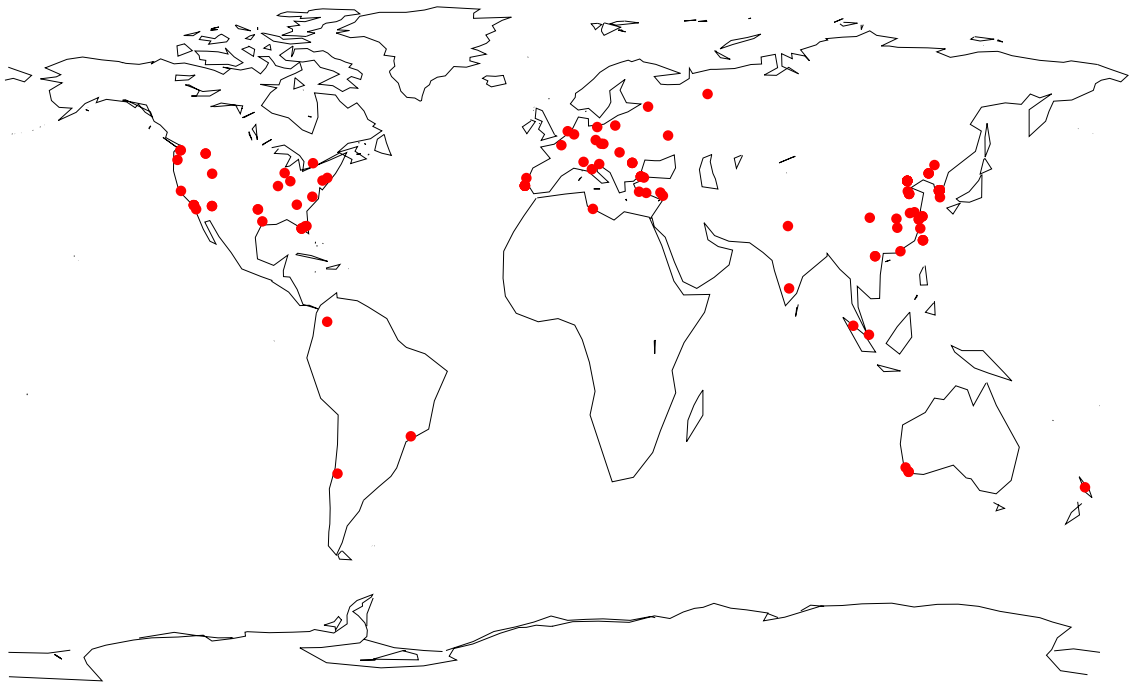

Figure 4.74: World Map, Lethic Botnet

This Botnet did not generate much traffic and most of the infected machines that were detected in this capture are from China and the United States of America.

We have already observed that there is a constant rate of Outbound peers, which leads to a World Map with only three dots in the entire World. So, for this analysis we only considered Inbound peers.

4.4.3 Low-Level Analysis

To conclude the analysis on the Botnets, we present the scalogram for the Lethic capture.

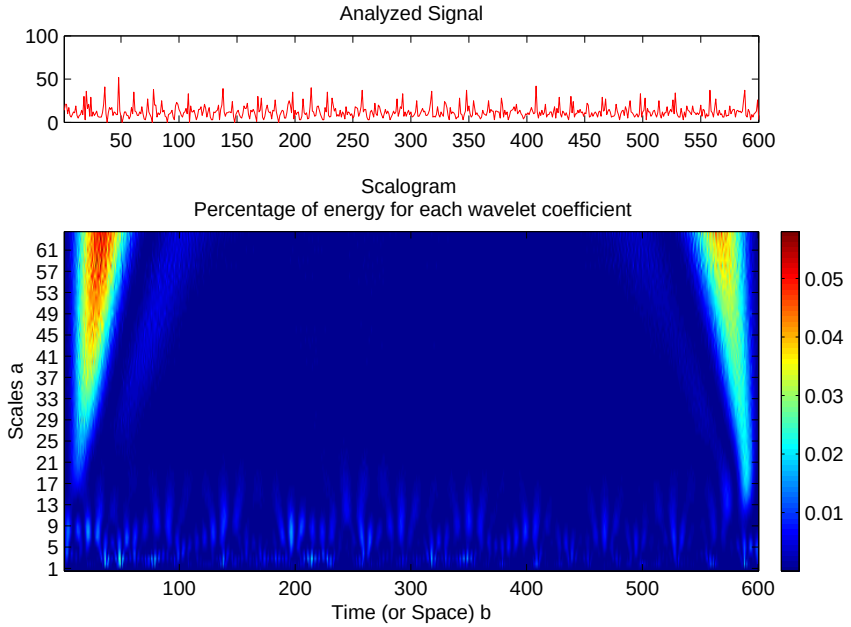

Figure 4.75: Lethic, Bobax Botnet

The last scalogram presents a similar behaviour to the Bobax scalogram. It also presents high percentages of energy close to the beginning and end of the trace. It is also possible to see the lower coefficients and the pattern they follow. This is easily explained by the variance of the signal over time.

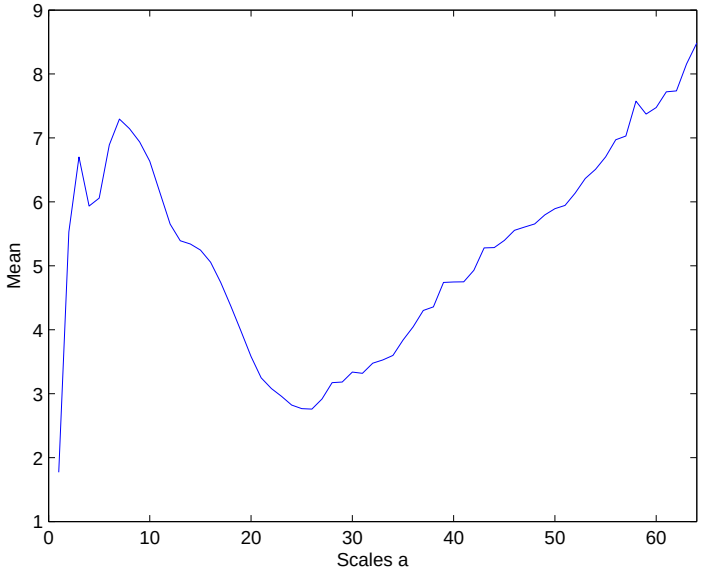

Figure 4.76: Energy Mean, Lethic Botnet

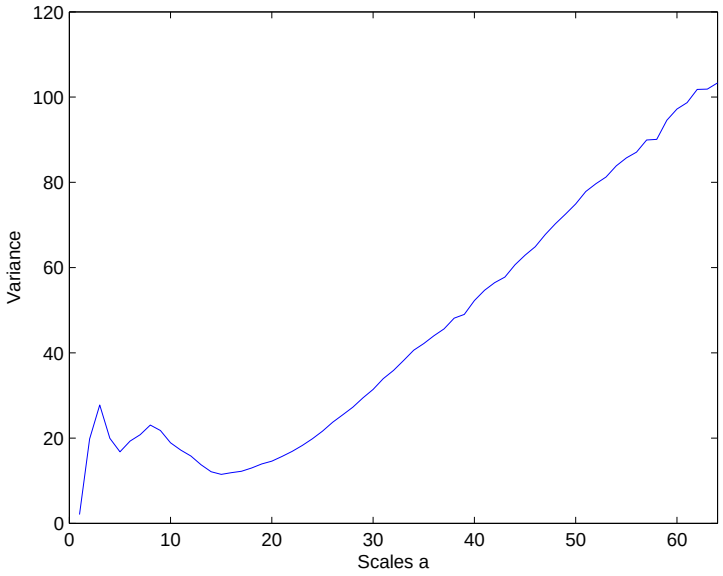

Figure 4.77: Energy Variance, Lethic Botnet

Once again the mean and the variance have quite distinct values. Both show a peak at the initial moments, followed by a decrease and a regular increase until the end of the analysed signal.

4.5 Kazy

The malware used for this trojan was downloaded from [35] on April 2011. Traffic was captured for a period of 48 hours, in order to make a deeper analysis.

This malware created a new process every time it was executed, disguised under the name "iexplorer.exe", which is a common behaviour of this trojan, or "malware.exe", which was the name of the executed malware.

Three captures were made for this trojan. The second capture produced some really interesting values, very different from the first capture, but due to a technical issue (an energy fail), the length of the capture was reduced to 30 hours. In an attempt to replicate the same results, a third capture was made, but unfortunately the results obtained were pretty similar to the first capture. Thus, even though the second capture was "incomplete", we decided to include the first two captures in this dissertation.

4.5.1 General analysis

After the trojan malware was downloaded and installed, a significant number of DNS queries were exchanged. After the initial burst of DNS packets, HTTP packets started being exchanged. After that, the major part of the exchanged traffic was Unknown TCP, that will be seen in the High-Level Analysis. Most of this traffic was HTTP on port 80 for the first hour; after this first hour, port 50000 started being used, which is a port known to be used by the Subsari malware. But the most part of the generated Unknown TCP packets were SMB packets. The trojan maintained this behaviour until the end of the capture. Only a couple of SMTP packets were discovered in the Unknown TCP packets.

The results of the second capture were not quite different from the ones corresponding to the first capture. Without considering the amount of packets generated, they also started with a burst of DNS queries. After that, HTTP and SMB packets were exchanged, with SMB corresponding to the highest percentage. Then, after the first hour and until the end of the capture, most of the generated traffic was Unknown TCP. Analyzing this unknown TCP traffic, we could conclude that it is mostly composed by SMB packets. There were also some HTTP packets throughout the capture, besides HTTP packets using port 81. This port is usually used by a malware named RemoConChubo. Like happened in the first capture, port 50000 (usually used by Subsari) was present.

We can then conclude that this trojan, despite not generating a significant number of SMTP packets, unlike it was expected, generated a significant amount of HTTP packets, both through port 80 and 81. The number of SMB packets was also different from the expected behaviour. This strongly indicates that the objective of this trojan is to perform DoS attacks, as well as finding private information in the infected machine. The rest of the results are as expected, and the three captures made for this trojan were very useful.

4.5.2 High-Level Analysis

This trojan generated the following traffic distribution among different Protocols.

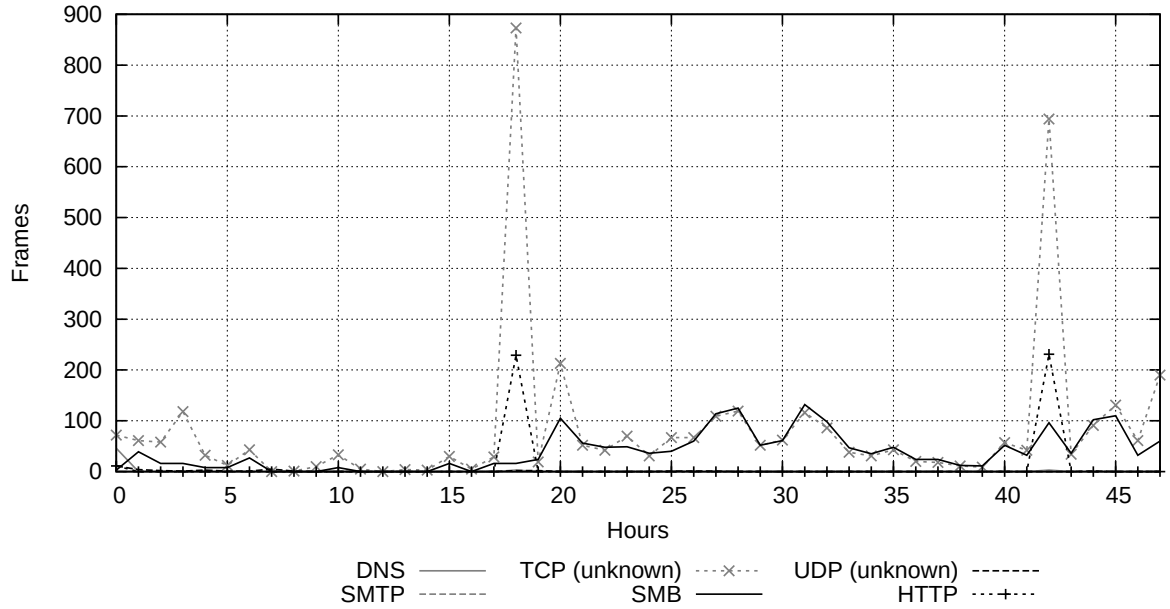

Figure 4.78: Protocols(Download), Kazy Botnet

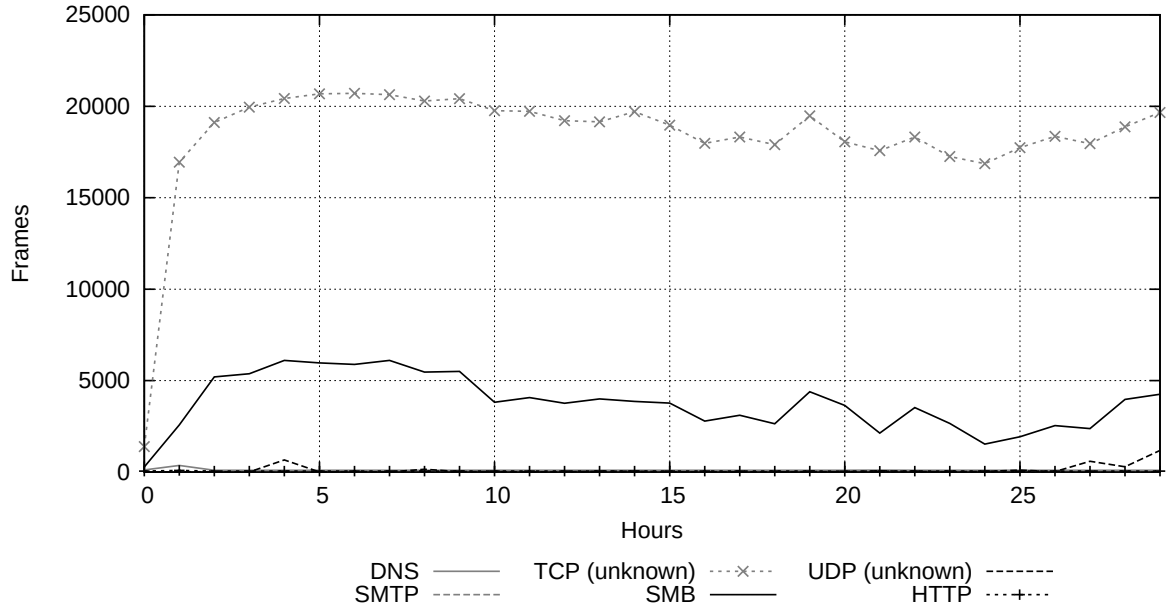

Figure 4.79: Protocols(Download), Kazy Botnet(2nd Capture)

By analyzing both captures, we can state that the most present protocols were SMB and Unknown TCP. In the first capture, it is possible to also see some DNS packets, as well as HTTP packets. In the second capture all protocols can be observed, except SMTP.

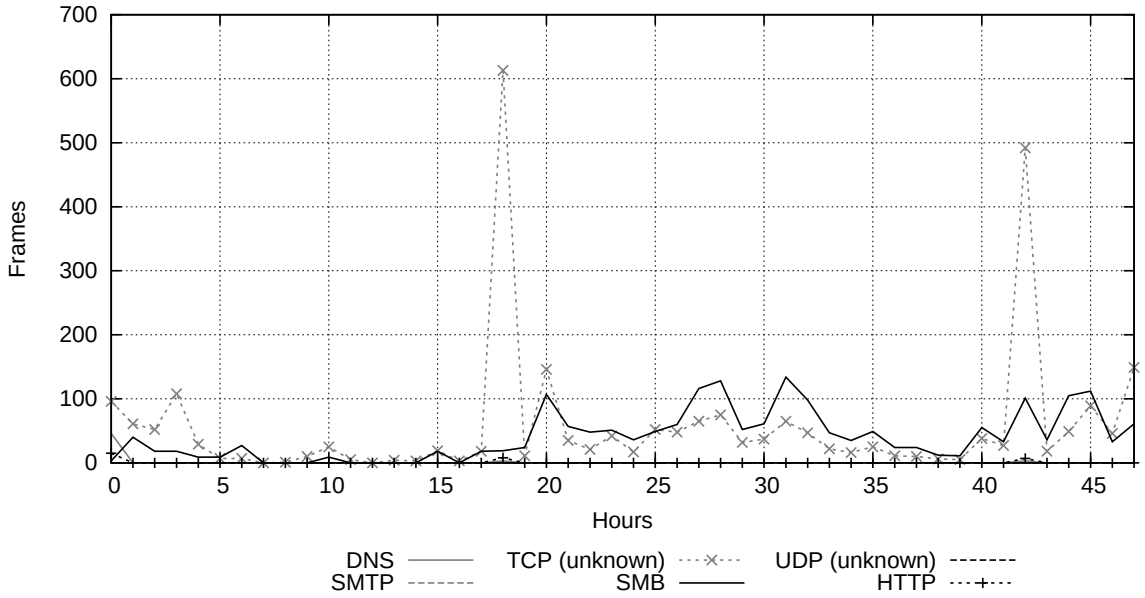

Figure 4.80: Protocols(Upload), Kazy Botnet

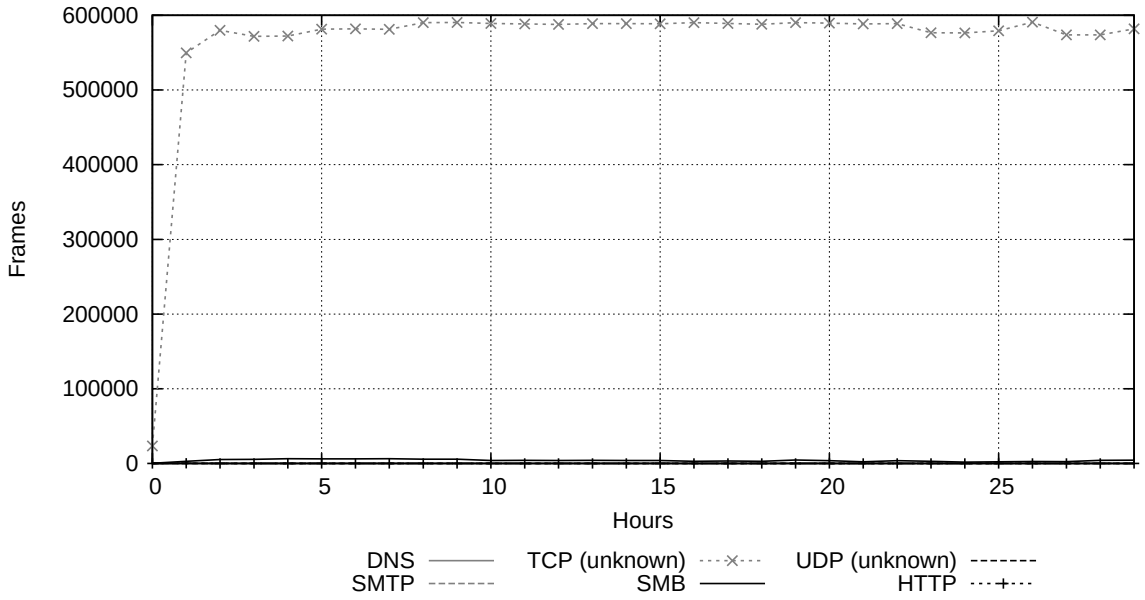

Figure 4.81: Protocols(Upload), Kazy Botnet(2nd Capture)

The number of Downloaded packets is very similar to the number of Uploaded packets. The biggest difference is in the second capture, where the number of Downloaded packets have an almost steady rate of sixty thousand Unknown TCP packets per hour, while Uploaded packets are around twenty thousand Unknown TCP packets. In the Upload direction we also have some Unknown UDP packets, and the amount of DNS packets is higher than the amount of Unknown TCP packets. When analyzing the Download packets, this situation reverses. It is also important to note the difference in the number of SMB packets. They were around four thousand per hour in the Upload direction, being almost insignificant in the Download direction.

As it was already discussed, it is important to point out that the Unknown TCP packets were mostly SMB packets for the first capture, while on the second they were almost from SMB and HTTP. Unknown UDP packets were mostly used for DoS attacks on both captures.

The next analysis was related to the amount of generated packets, and the statistics illustrated in the following figures were obtained.

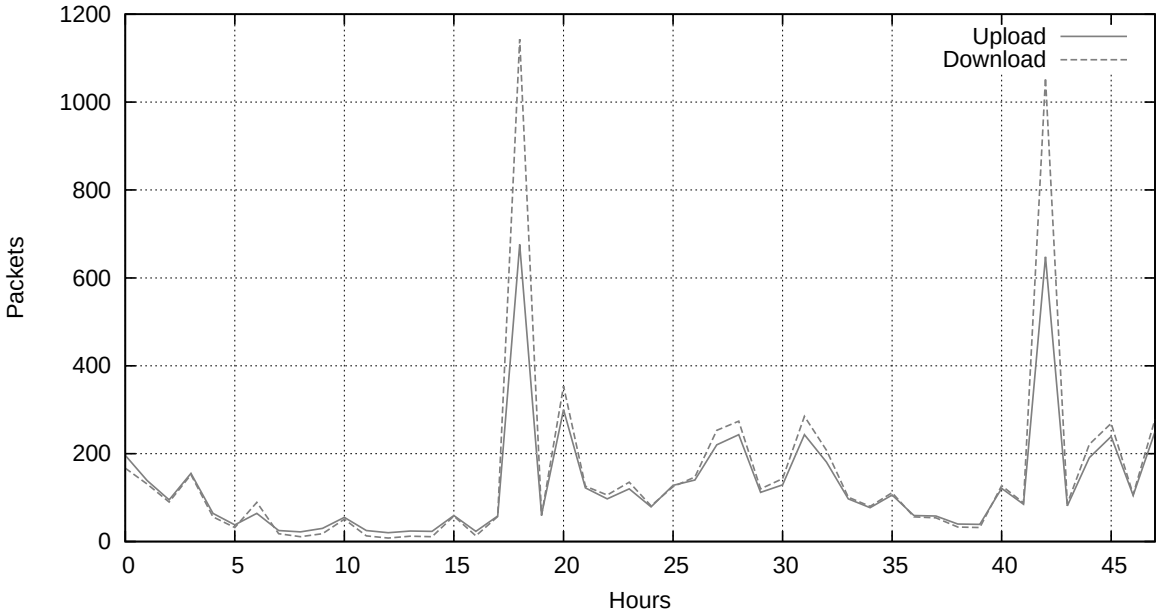

Figure 4.82: Packets per hour, Kazy Botnet

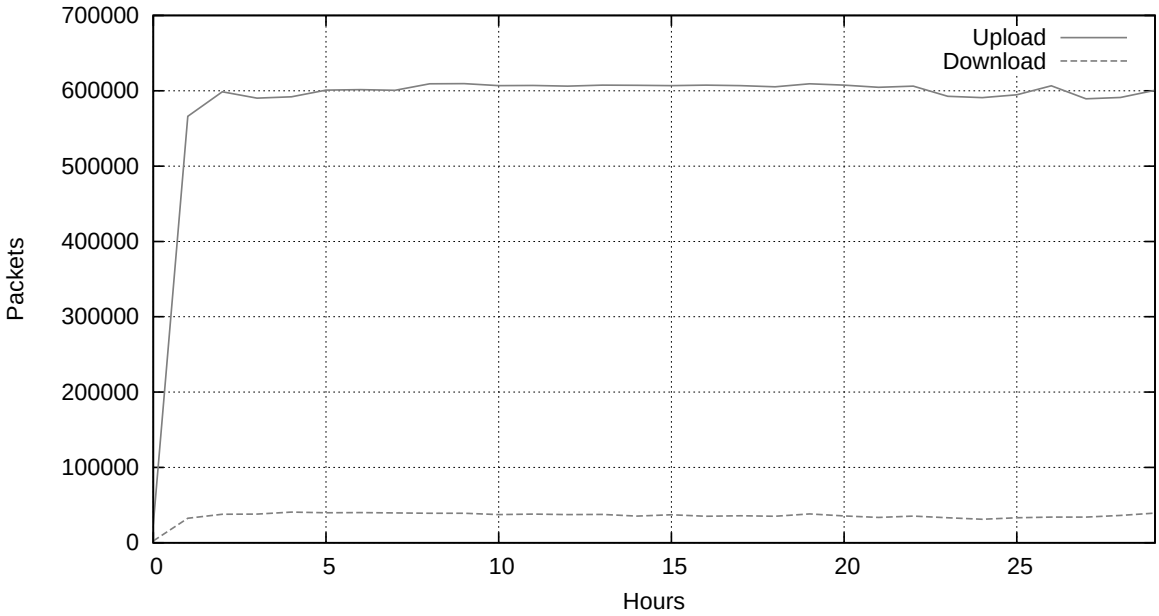

Figure 4.83: Packets per hour, Kazy Botnet(2nd Capture)

The number of packets generated in the first capture does not have a very erratic pattern, except for the two visible peaks. In the second capture, the pattern is really clear. It is important to note that in the first hour the number of Upload packets is very small when compared to the number at any other hour.

This analysis provides valuable information. In the first capture, the traffic generated would not be enough to claim that we were facing a trojan infection, but in the second capture this clearly resembles a case of infection. The normal rate of Upload packets in the second capture was around six hundred thousand packets per hour.

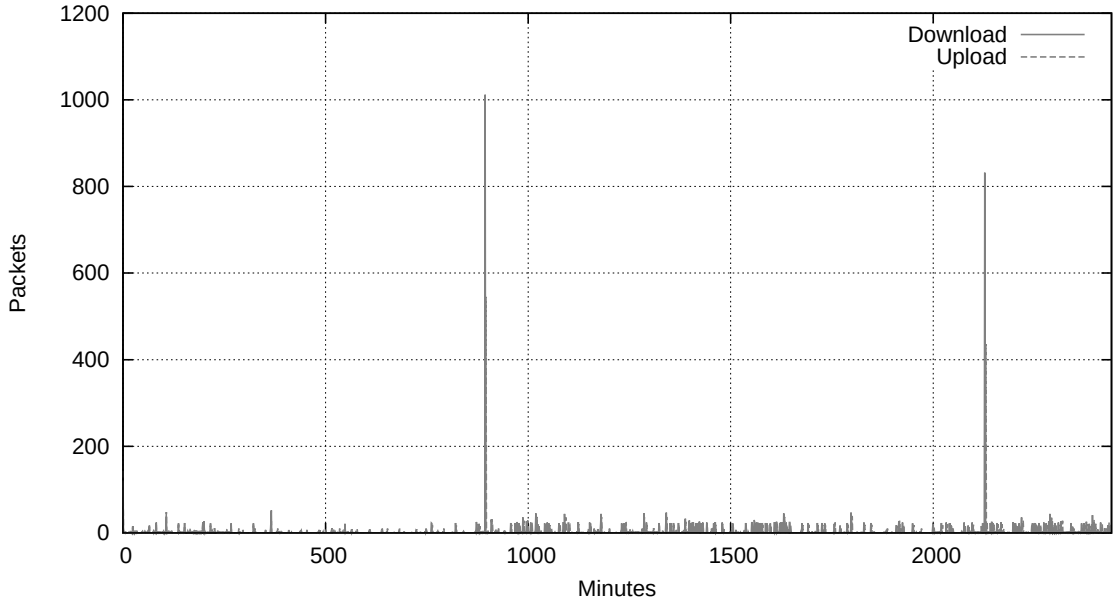

Figure 4.84: Packets per minute, Kazy Botnet

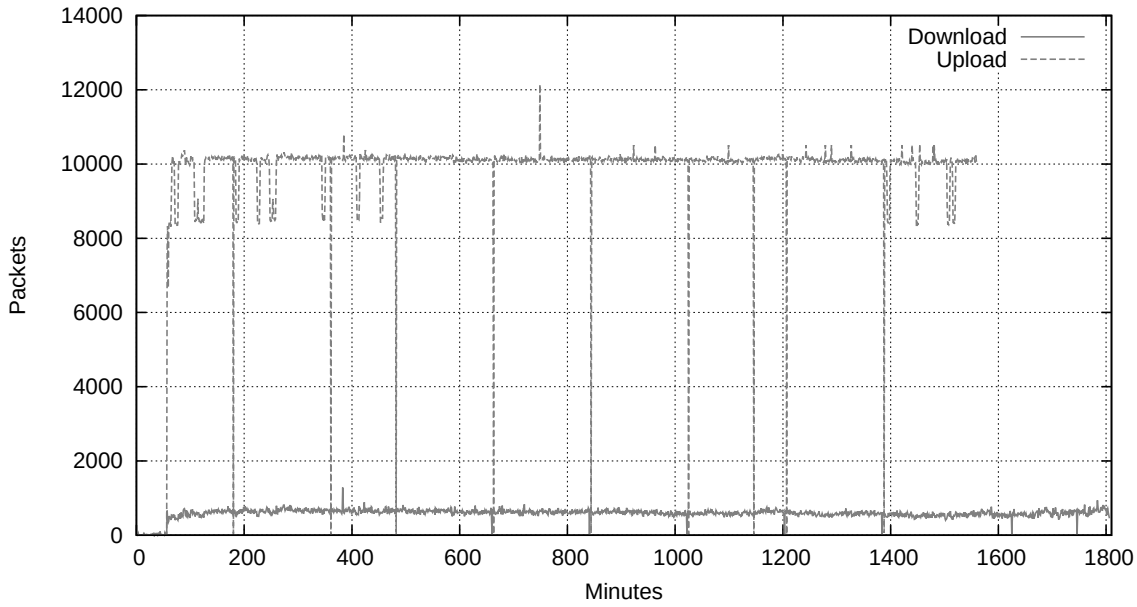

Figure 4.85: Packets per minute, Kazy Botnet(2nd Capture)

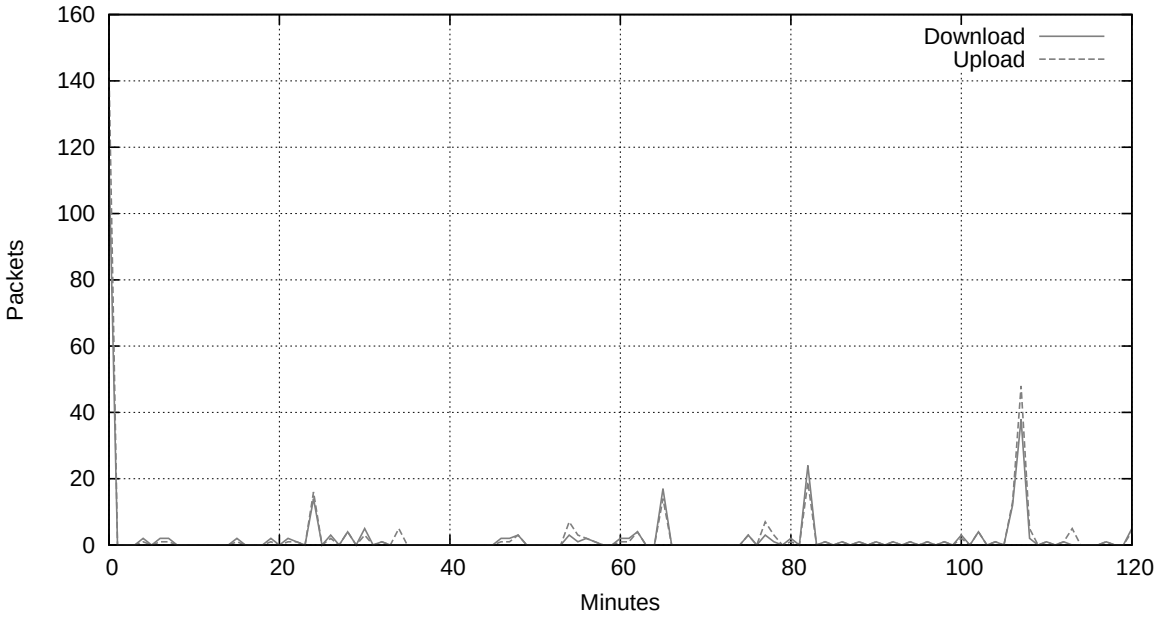

Figure 4.86: Sample of packets per minute, Kazy Botnet

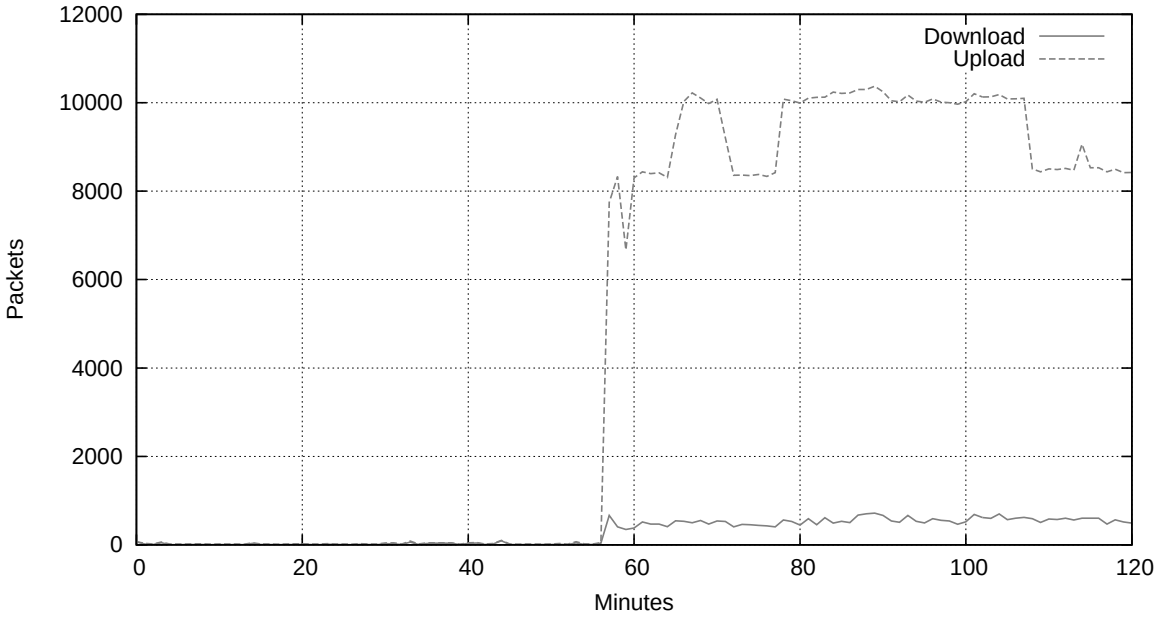

Figure 4.87: Sample of packets per minute, Kazy Botnet(2nd Capture)

The last four pictures were made to highlight the behaviour of the generated packets. As it is visible, there are various bursts of packets, specially on the second capture. We can see in the sample of the second capture that the normal rate is around ten thousand packets per minute, but occasionally there are bursts of six hundred thousand packets per minute. The first capture has a more regular rate of packets, without much variance.

The next analysis consisted in observing the amount of data generated from those packets. Following the same criteria used in the previous procedures, we obtained the statistics that are shown in the next page.

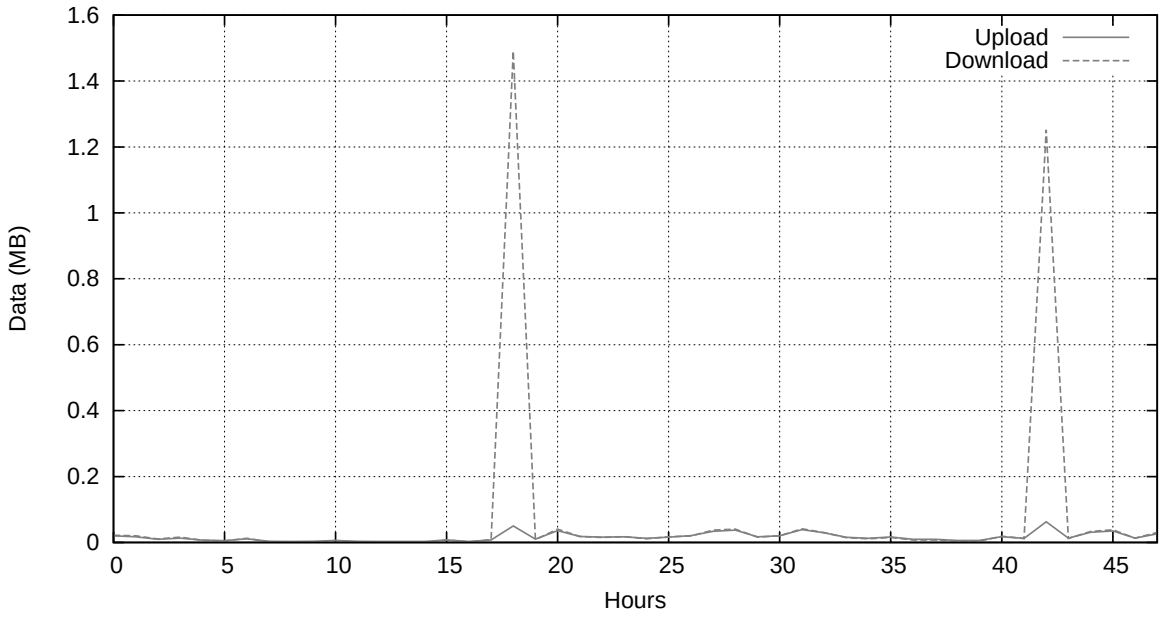

Figure 4.88: Amount of traffic per hour, Kazy Botnet

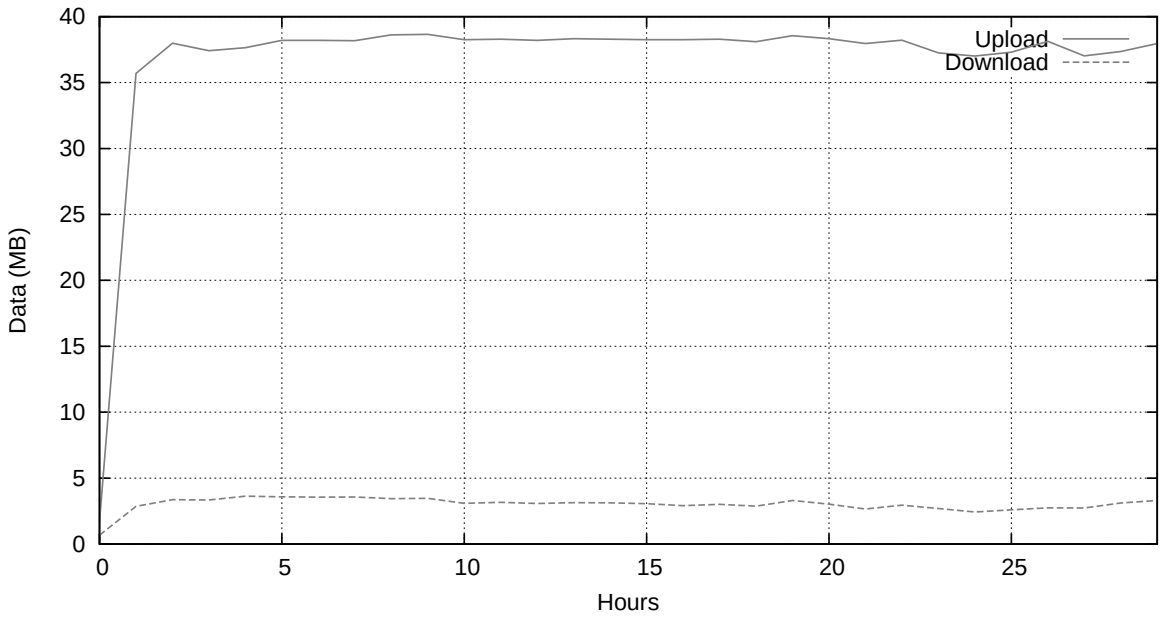

Figure 4.89: Amount of traffic per hour, Kazy Botnet(2nd Capture)

These two images show that the behaviour is very similar to what was seen in the number of packets per hour. The first capture presents the same two peaks as before, and the second also shows no relevant peaks. For the first capture, the values of the amount of traffic per hour are regular, with a very short amount of generated traffic. The second capture is a different case, where the amount of traffic can raise an alarm because the average rate of Uploaded packets is around 40MB per hour.

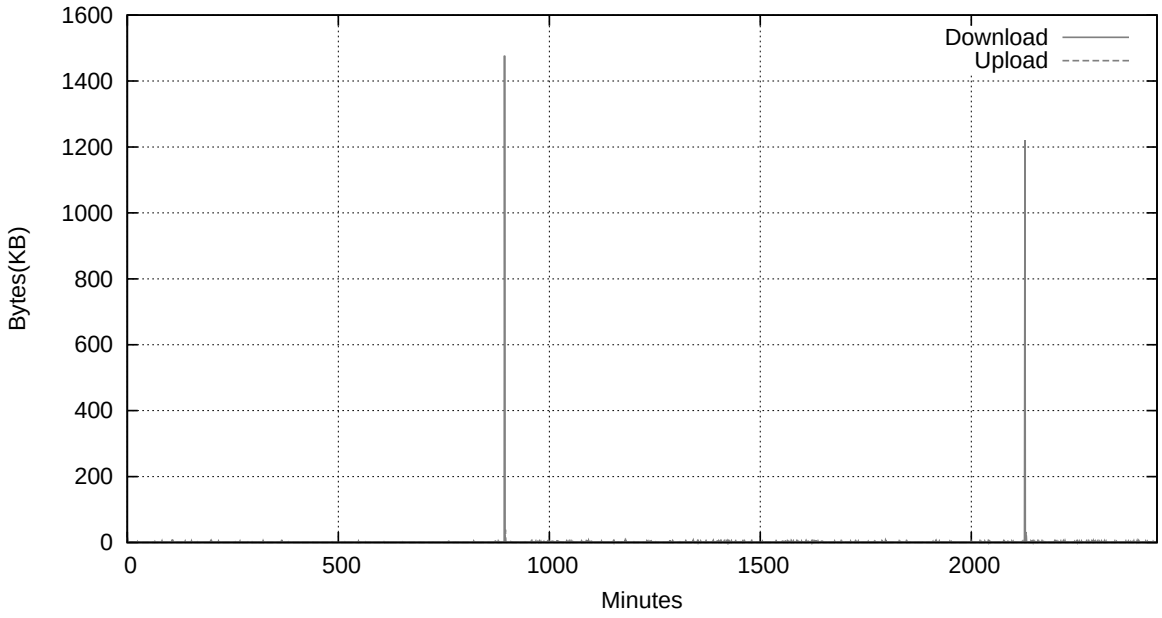

Figure 4.90: Amount of traffic per minute, Kazy Botnet

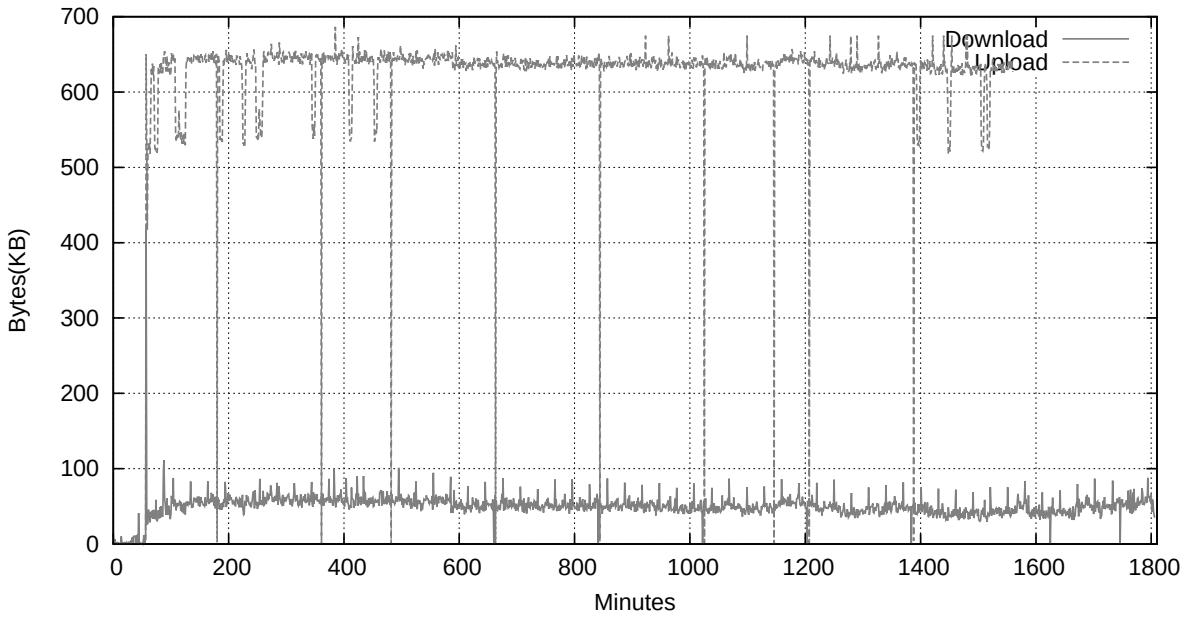

Figure 4.91: Amount of traffic per minute, Kazy Botnet(2nd Capture)

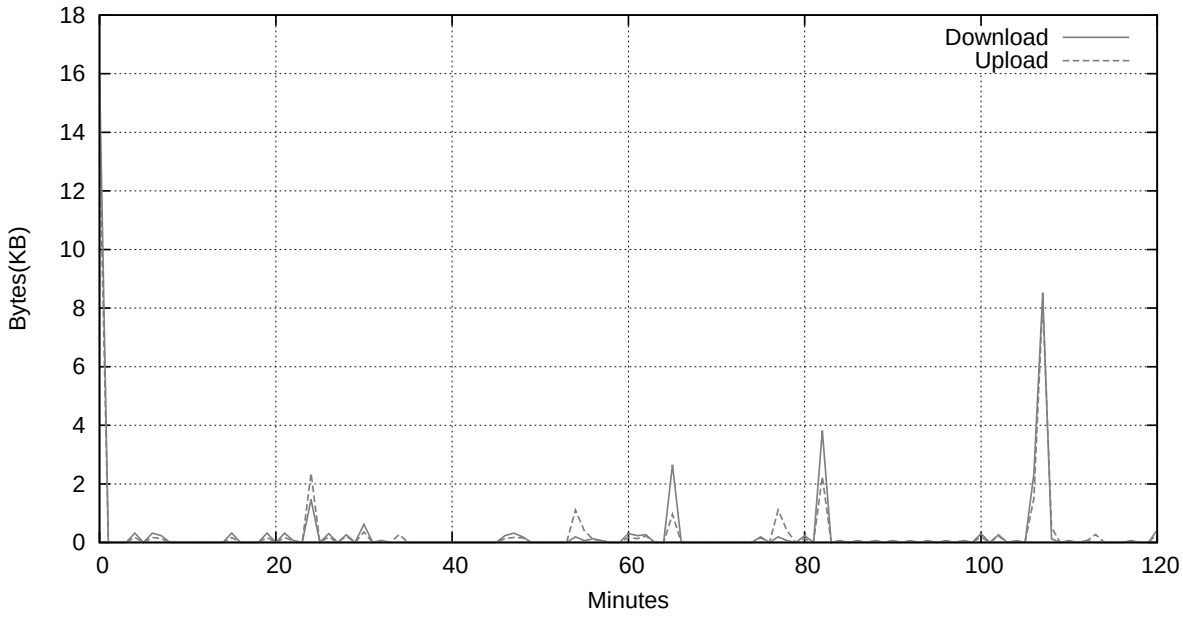

Figure 4.92: Sample of amount of traffic per minute, Kazy Botnet

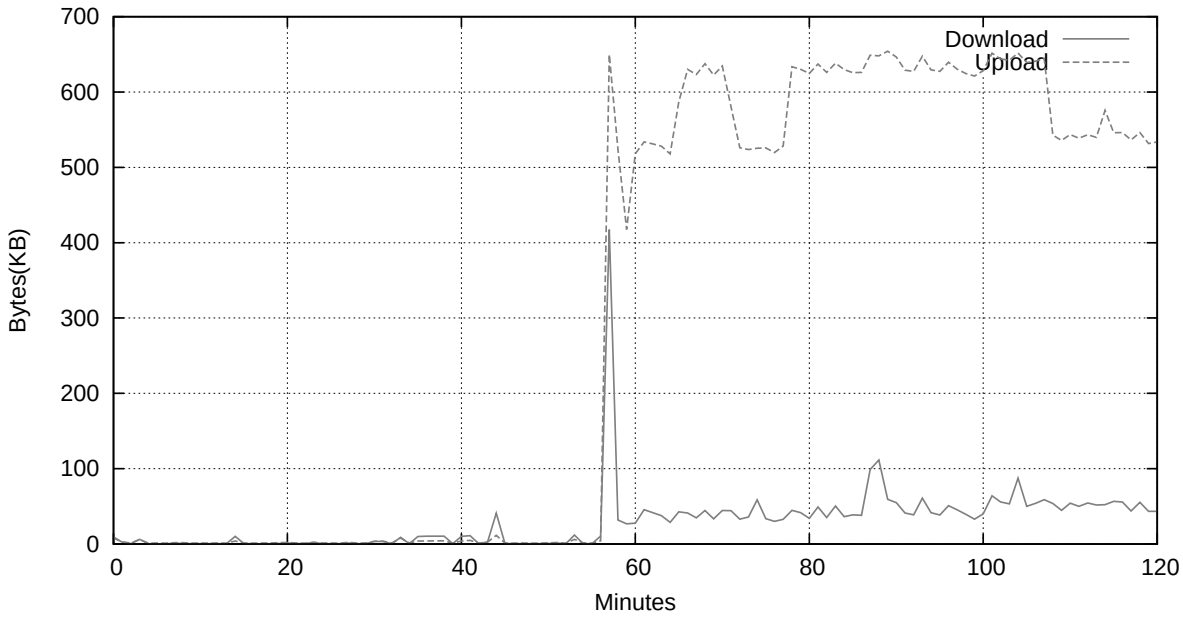

Figure 4.93: Sample of amount of traffic per minute, Kazy Botnet(2nd Capture)

Once again, the second capture presents a high variance on the amount of generated traffic. The first capture follows a similar pattern as the one observed in the packets per minute statistic. The sample from the second capture clearly shows the difference in behaviour that is exhibited by this trojan between the first and the second hours.

The next step was to analyse the number of peers involved.

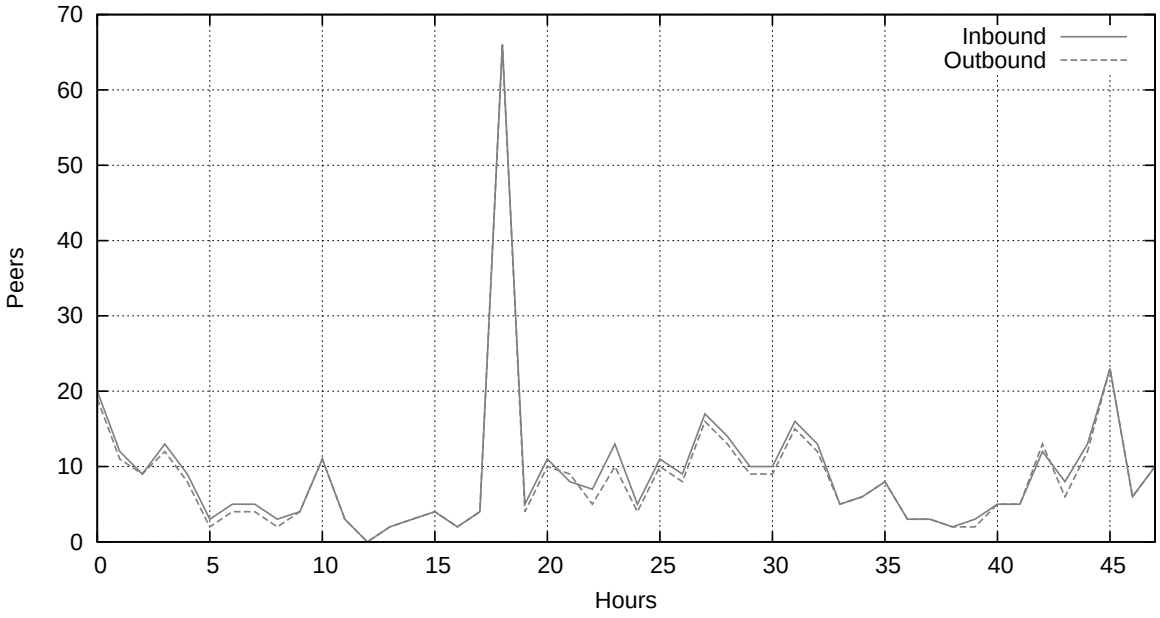

Figure 4.94: Unique peers per hour, Kazy Botnet

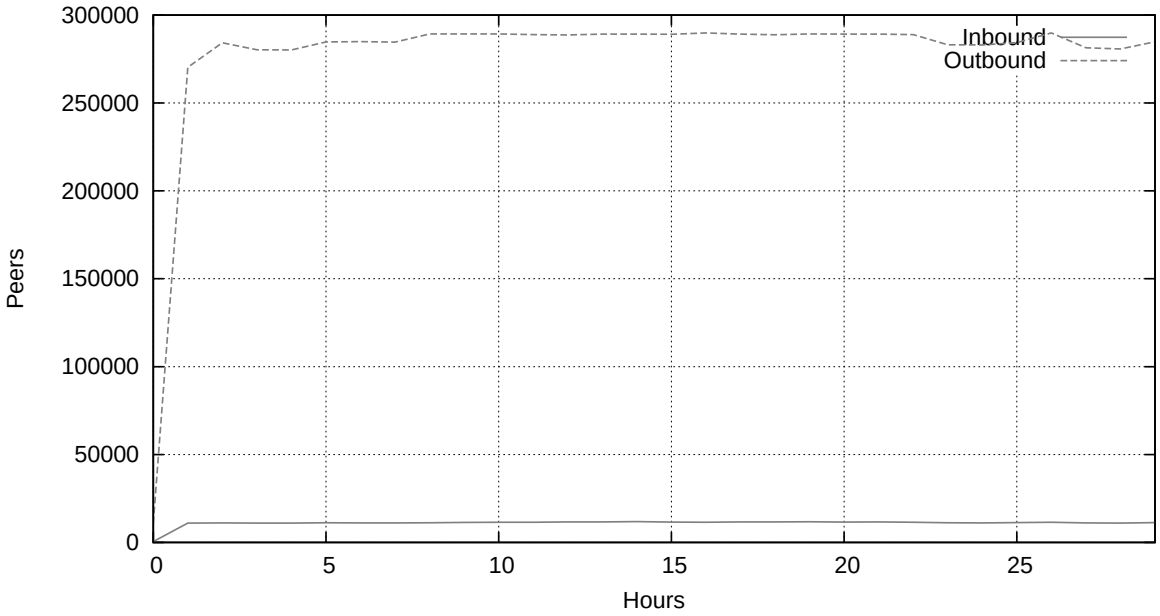

Figure 4.95: Unique peers per hour, Kazy Botnet(2nd Capture)

The first capture did not raise much suspicions, with an average of near ten peers per hour and one peak located around hour 18, with around 60 peers. This peak leads to an increase on the TCP Session Establishment attempts. Meanwhile, the second capture provided a very different result. The number of peers contacted per hour was in the order of 290 thousand peers. There were no significant peaks registered in the second capture, just an increase from the first hour to the remaining of the capture.

Next, TCP Session Establishments were analyzed. The results can be seen in the following pictures.

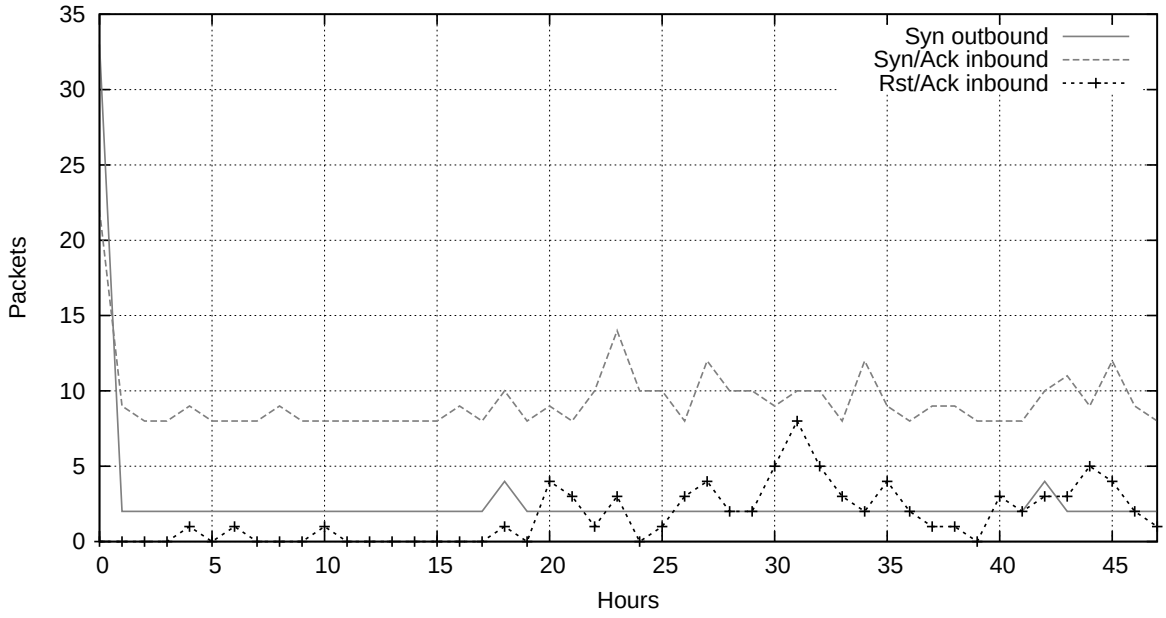

Figure 4.96: TCP Session Establishment(Outbound), Kazy Botnet

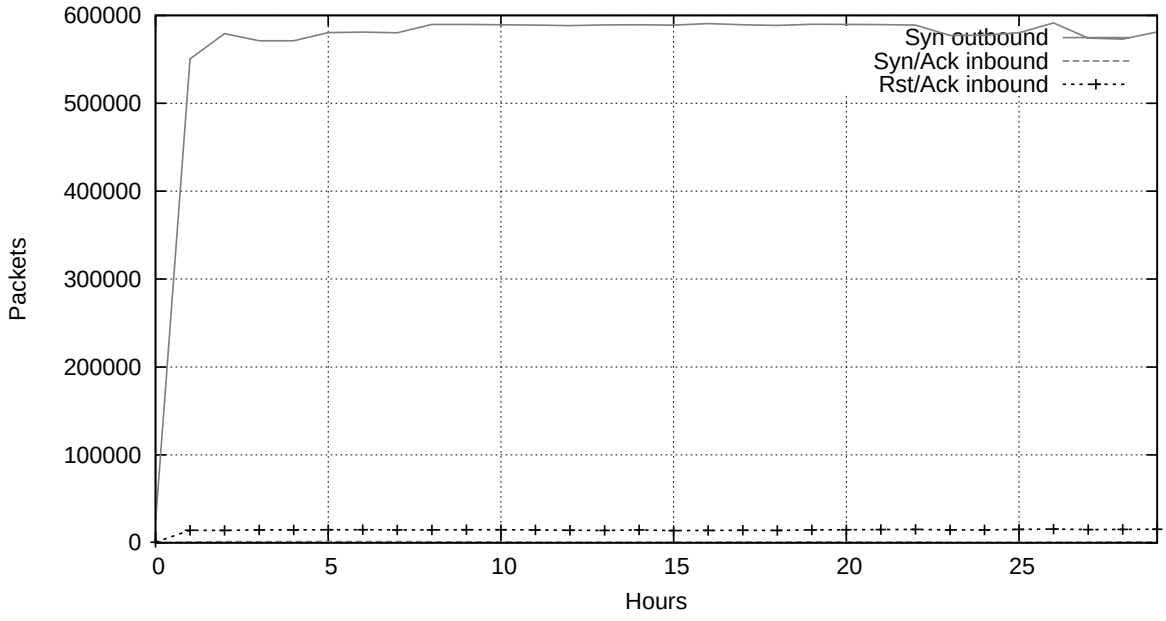

Figure 4.97: TCP Session Establishment(Outbound), Kazy Botnet(2nd Capture)

In the first capture we see that there are more SYN/ACK than SYN packets. This value is not normal and is an indication of a possible infection. There are a couple of RST/ACK packets as well, but in average lower than the number of SYN packets.

In the second picture, we see that there is a huge amount of generated SYN packets, and the major part of them do not have any response, either SYN/ACK or RST/ACK. This suggests a possible infection, since this number is not normal.

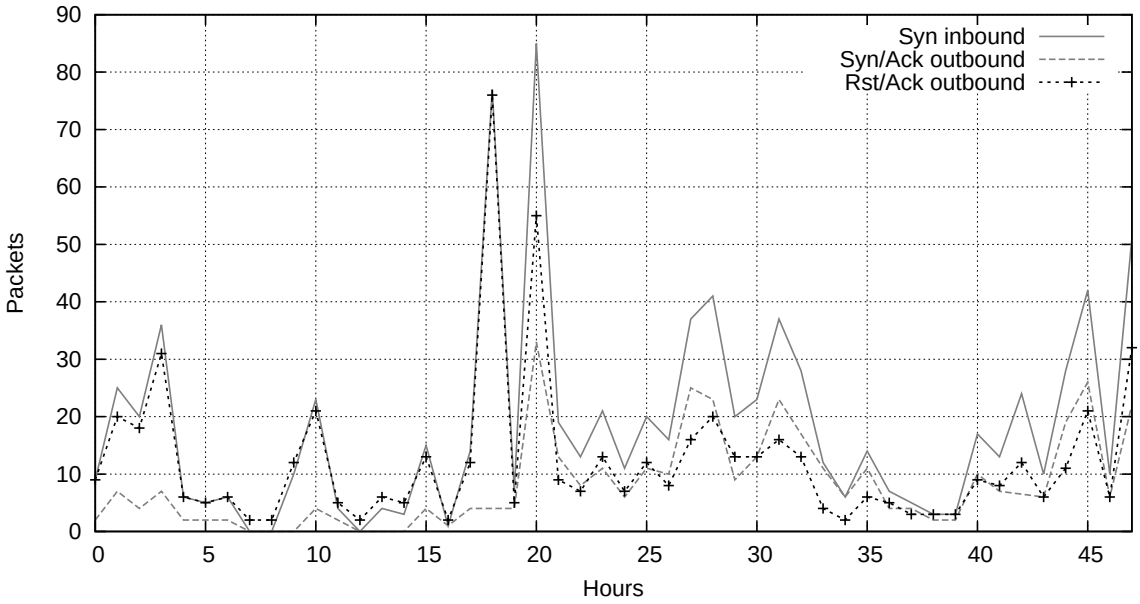

Figure 4.98: TCP Session Establishment(Inbound), Kazy Botnet

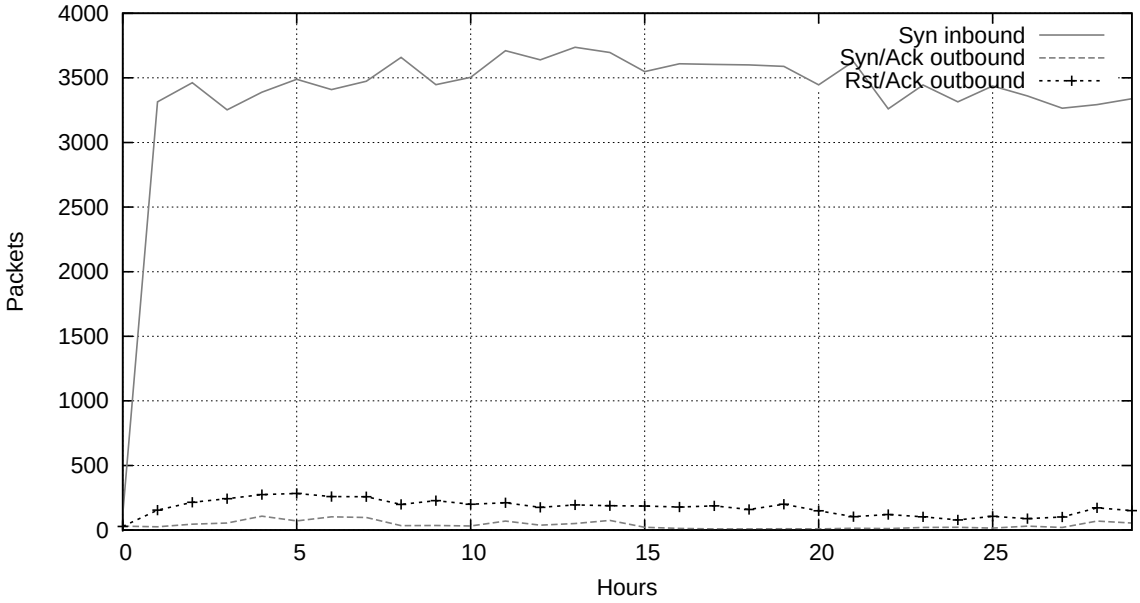

Figure 4.99: TCP Session Establishment(Inbound), Kazy Botnet(2nd Capture)

In the last two pictures, the situation changes. In the first one, most of the SYN packets have a reply. In the first hours most of the SYN packets were replied with a RST/ACK, but after hour 20 the number of SYN/ACK as RST/ACK packets were almost always the same. The amount of generated RST/ACK are a warning to take further actions.

The second picture shows a similar behaviour to the Outbound attempts. The number of SYN packets received, however, dropped from six hundred thousand to around four thousand packets. Therefore, it is easier to observe the SYN/ACK and RST/ACK packets sent. Nonetheless, a lot of SYN packets did not obtain any reply. Once again, this suggests that we are facing an infection.

Some world maps representing the location of the peers that communicated with the infected machine were also made.

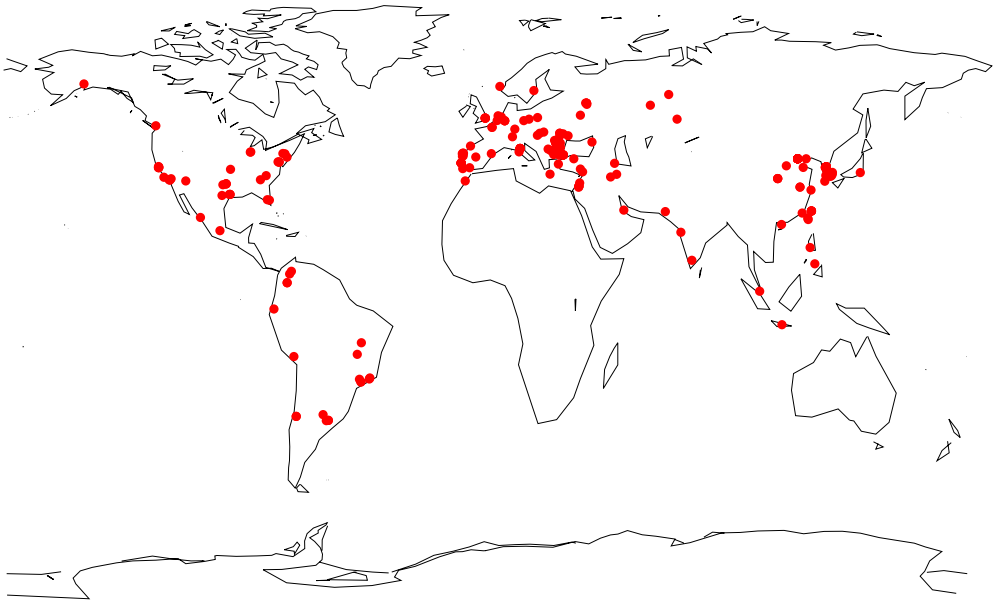

Figure 4.100: World Map, Kazy Botnet

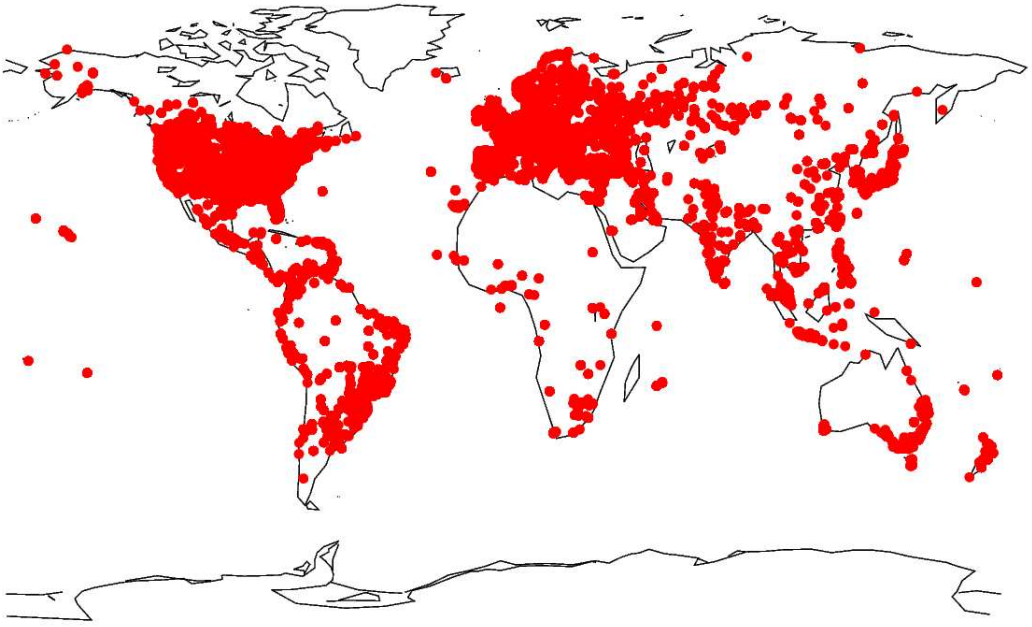

Figure 4.101: World Map, Kazy Botnet(2nd Capture)

In the first capture, we can observe that most of the peers are from Europe and America. The main infected countries are China and the United States of America. The second capture, however, generated a totally different map. Basically, all continents had a lot of infected machines. The most affected continents are Europe and America. The countries that seem more affected are however, Russia and the United States of America. This picture shows very well the propagation of this trojan and the danger it poses.

Regarding the Inbound/Outbound amount of peers, we already observed that in the first capture they were very similar. This originated two identical World Maps, so only Inbound data was considered. However, in the second capture, we saw that there were many more Outbound than Inbound peers. As suspected, this resulted in two different World Maps. In order to better understand their difference, we decided to include it.

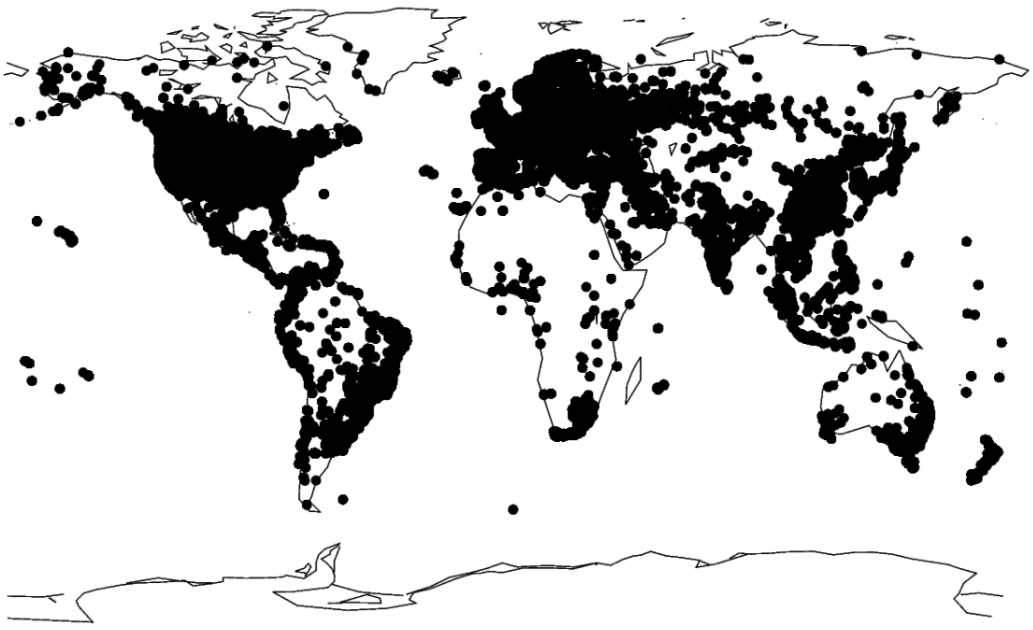

Figure 4.102: World Map, Kazy Botnet(2nd Capture, Outbound Connections)

4.5.3 Low-Level Analysis

To end this chapter, two scalograms were generated, one for each capture.

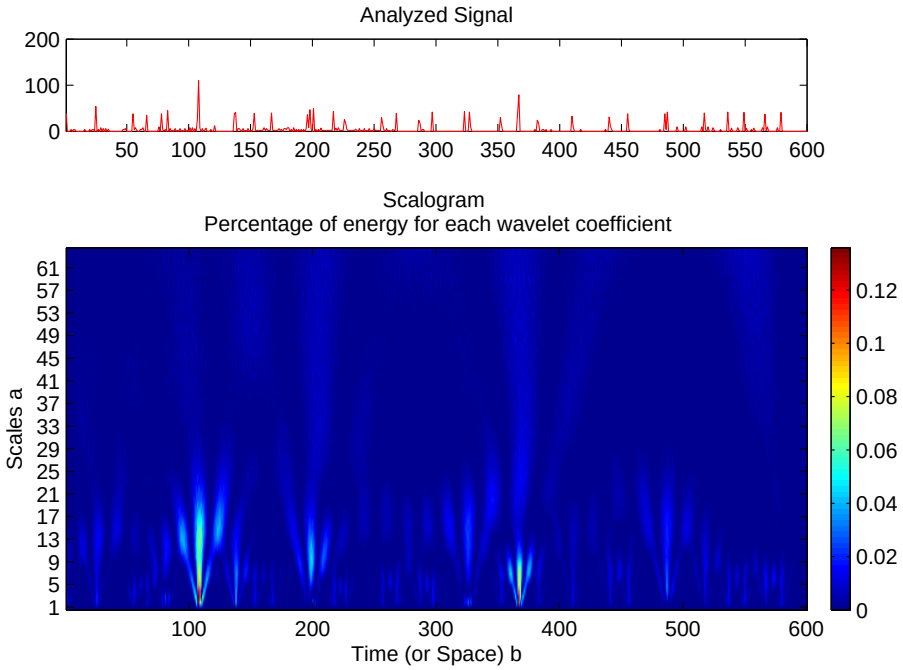

Figure 4.103: Scalogram, Kazy Botnet

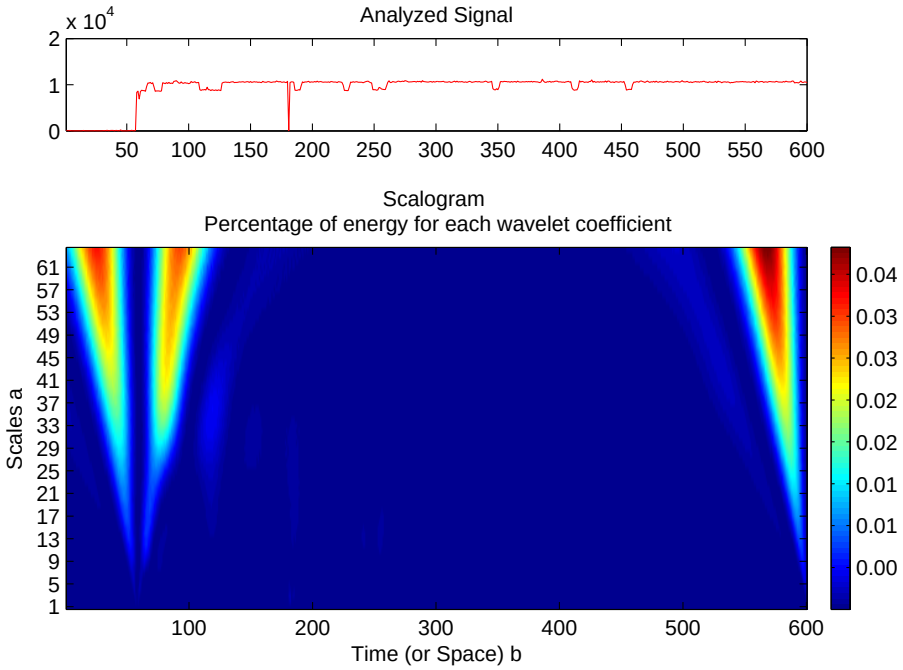

Figure 4.104: Scalogram, Kazy Botnet(2nd Capture)

The last two scalograms show very well the difference between the first and second captures. This first only reveals a significant increase in energy percentage around minute 110 and 380, where there is a higher variance on the analysed signal. The rest of the scalogram follow a fairly regular pattern, without much variance. Meanwhile, the second reveals a huge percentage of energy at the beginning and end of the time scale, like happened in other scalograms. It also has a peak in the energy percentage around minute 60, contrasting with the first hour where the signal energy was almost null. This high variance on the signal was the reason for the peak.

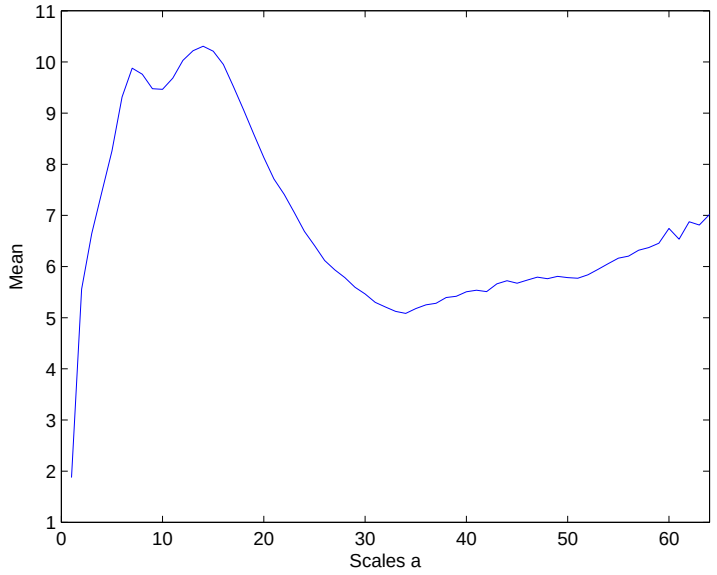

Figure 4.105: Energy Mean, Kazy Botnet

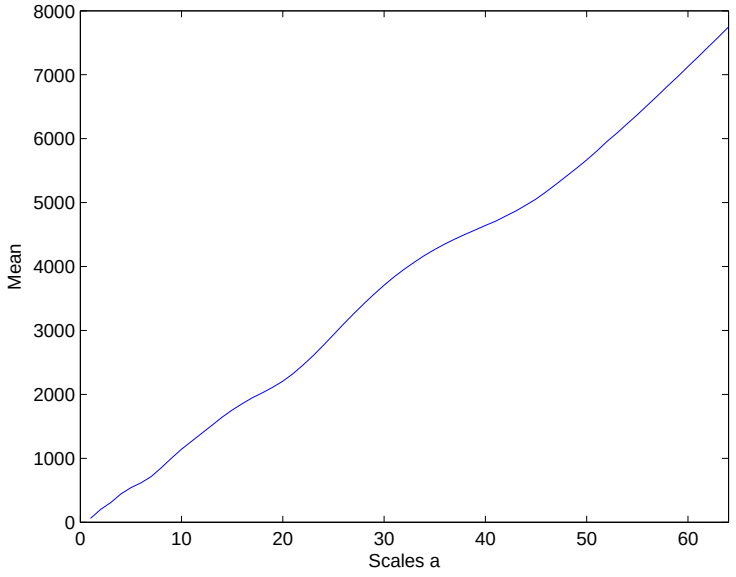

Figure 4.106: Energy Mean, Kazy Botnet(2nd Capture)

These last two pictures also stress the difference between both captures. The first has a peak around the initial moments followed by a regular increase until the end of the signal. The second presents a steady increase from the beginning to the end of the signal. The values in both captures are also very different, as expected.

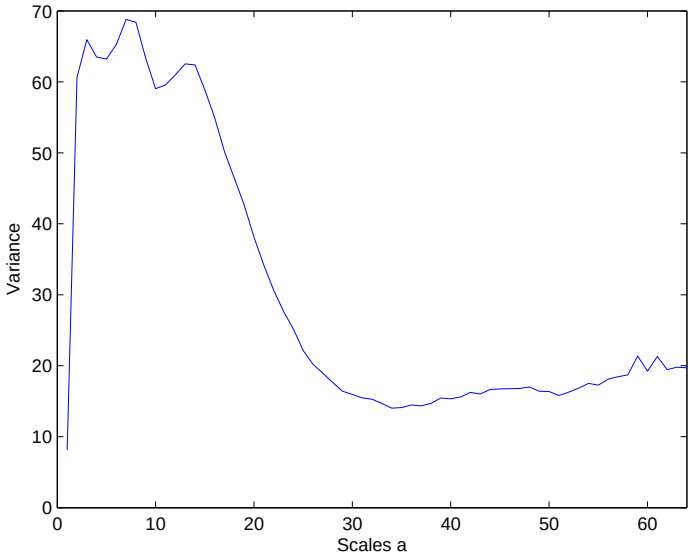

Figure 4.107: Energy Variance, Kazy Botnet

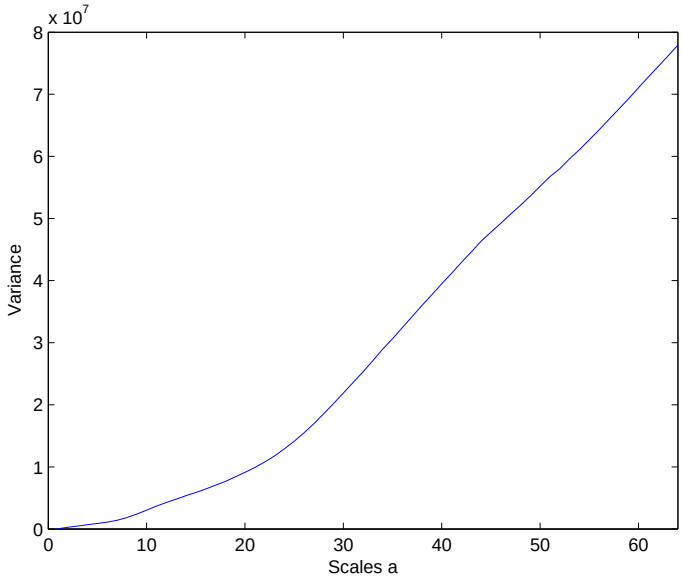

Figure 4.108: Energy Variance, Kazy Botnet(2nd Capture)

Comparing the variance, the results were similar to what was observed on the mean analysis. Once again, the first capture presents a peak in the first moments, followed by a small regular increase, and the second presents a steady increase. Again, the values between the two are very different, as well as their behaviour.

Chapter 5

Conclusions

5.1 Final conclusions

Most of the initial objectives for this dissertation were accomplished. The characterization of the different types of Botnets was fulfilled. It was interesting to observe the evolution from the first IRC Botnets to current state of the art Botnets. After an initial research about the top HTTP Spam Botnets, some popular Botnets were selected, installed and successfully studied, leading us to conclude that they performed as expected. It was also interesting to see the update on the top Botnets and observe that the suspicions that Lethic would become quite relevant became true. After some attempts, it was possible to find malware for all the selected Botnets. These HTTP spam Botnets were deeply studied and analysed. Although the traffic generated in the infected machine was not always as expected, many common behaviours were found, clearly indicating the objective of the used malware. Being able to geographically locate the infected machines was also very important to observe the spread of the analyzed Botnets. Being able to capture traffic in different conditions and using more realistic environments could also be very important, possibly leading to a more accurate analysis.

Due to the evolution of the Botnets, it is becoming harder to characterize their traffic, since they tend to disguise themselves, cypher their traffic and present more intermittent activity. From the analysis that was conducted in this dissertation, it was possible to understand that HTTP Botnets mainly focus on searching for private information (communicating the collected information via HTTP) and sending Spam emails.

5.2 Future Work

The following topics should be considered for future work:

- Make longer captures
- Improve the low-level analysis
- Implement detection mechanisms
- Include more realistic simulation environments

- Use different Operating Systems.

Longer captures would mean a larger time window, which could increase the probability of capturing more different behaviours from the Botnets. We know that many Botnets only operate strictly when they need to, showing behaviors different from normal.

Scalograms should be further explored in order to look for more behavioural patterns that can help in the Botnet detection phase. This could include a deeper analysis of the generated packets, a deeper look at the ports used in the communications, among other techniques.

It is very important to implement detection mechanisms and take defensive measures when an infection is detected. Using the captures and statistics of this dissertation, it would be interesting to implement rules that could be used to detect and prevent infections and see how they would perform.

Another very important topic is to include more realistic simulation environments. In this dissertation, the machine was only used to become infected and capture traffic, and all this happened after being formatted. It would be important to include a more realistic machine, similar to a regular machine. It would be also useful to infect an already infected machine with another malware from a different Botnet, to see if they would perform independently, or if one of the Botnets could affect the other.

At last, it would be interesting to explore the same mechanisms that were used here in different Operating Systems.

Bibliography

[1] B. Waldecker, "A review on irc botnet detection and defence," 2009.

[2] F. Research, "Industry statistics." http://email-museum.com/reports/industry-statistics/, 2009.

[3] B. Krebs, "Invasion of the computer snatchers." http://www.washingtonpost.com/wp-dyn/content/article/2006/02/14/AR2006021401342.html, 2006.

[4] http://news.bbc.co.uk/2/hi/8547453.stm. [Retrieved June 3rd, 2011].

[5] "Botnet." http://en.wikipedia.org/wiki/Botnet.

[6] "What is a botnet?." http://what-is-what.com/what_is/Botnet.html, 2007.

[7] Hackersbay, "What is ddos attack and how does it work?." http://www.hackersbay.in/2011/01/what-is-ddos-attack-and-how-does-it.html. [Retrieved July 25th, 2011].

[8] Symantec, "Message labs intelligence 2010 annual report," tech. rep., 2011.

[9] Symantec, "Symantec intelligence report: June 2011," tech. rep., 2011.

[10] J. Liu, Y. Xiao, K. Ghaboosi, H. Deng, and J. Zhang, "Botnet: Classification, attacks, detection, tracing, and preventive measures.," *EURASIP J. Wireless Comm. and Networking*, pp. 1–11, 2009.

[11] P. Correia, "Caracterizacao estatistica de botnets," Master's thesis, Universidade de Aveiro, 2011.

[12] *Botnet Detection and Mitigation*, (Pace University, White Plains, NY, USA), Proceedings of Student-Faculty Research Day, CSIS, 2010.

[13] H. Project, "Know your enemy: Tracking botnets." Published on the Web.

[14] J. Govil, "Examining the criminology of bot zoo," in *Information, Communications Signal Processing, 2007 6th International Conference on*, pp. 1 –6, dec. 2007.

[15] P. Barford and V. Yegneswaran, "An inside look at botnets," in *Malware Detection*, Boston, MA: Springer US, 2006.

[16] R. Puri, "Bots & botnets: An overview," *SANS Institute*, 2003.

[17] E. Cooke and F. Jahanian, "The zombie roundup: Understanding, detecting, and disrupting botnets." http://www.usenix.org/event/sruti05/tech/full_papers/cooke/cooke_html/.

[18] M. S. I. Report, "Irc botnets." http://www.microsoft.com/security/sir/story/default.aspx!botnetsection_irc. [Retrieved September 17th, 2011].

[19] B. Botezatu, "Anatomy of a botnet." http://www.malwarecity.com/blog/anatomy-of-a-botnet-196.html.

[20] I. Arce, E. Levy, and E. Levy, "An analysis of the slapper worm," *IEEE Security & Privacy*, 2003.

[21] J. Stewart, "Sinit p2p trojan analysis." http://wwww.securiteam.com/securityreviews/6J0022A95E.html.

[22] J. Stewart, "Phatbot trojan analysis." http://www.secureworks.com/research/threats/phatbot/?threat=phatbot.

[23] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling, "Measurements and mitigation of peer-to-peer-based botnets: A case study on storm worm," 2008.

[24] P. Wang, S. Sparks, and C. C. Zou, "An advanced hybrid peer-to-peer botnet," in *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, (Berkeley, CA, USA), USENIX Association, 2007.

[25] M. Polychronakis, P. Mavrommatis, and N. Provos, "Ghost turns zombie: Exploring the life cycle of web-based malware." http://www.usenix.org/event/leet08/tech/full_papers/polychronakis/polychronakis_html/.

[26] P. Bacher, T. Holz, M. Kotter, and G. Wicherski, "Know your enemy: Tracking botnets." http://www.honeynet.org/node/52.

[27] J. R. Binkley and S. Singh, "An algorithm for anomaly-based botnet detection," in *Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet - Volume 2*, (Berkeley, CA, USA), pp. 43 –48, USENIX Association, 2006.

[28] W. Lu, M. Tavallaee, and A. A. Ghorbani, "Automatic discovery of botnet communities on large-scale communication networks," in *Proceedings of the 2009 ACM Symposium on InformAtion, Computer and Communications Security*, pp. 1 –10, March 2009.

[29] "What is a botnet trojan?." http://www.dslreports.com/faq/14158.

[30] "Kazy trojan." http://www.spywareremove.com/removekazytrojan.html.

[31] M. S. Labs, "Grum." http://www.m86security.com/labs/spambotitem.asp?article=898.

[32] M. S. Labs, "Pushdo." http://www.m86security.com/labs/i/Pushdo,spambot.900~.asp.

[33] M. S. Labs, "Bobax." http://www.m86security.com/labs/spambotitem.asp?article=901.

[34] M. S. Labs, "Lethic." http://www.m86security.com/labs/spambotitem.asp?article=1205.

[35] http://www.offensivecomputing.net/.

[36] http://code.google.com/p/pygeoip/.