Macrothink Institute™

Network Protocols and Algorithms
ISSN 1943-3581
2012, Vol. 4, No. 4

Characterization and Modeling of Top Spam Botnets

Nuno Rodrigues

Polytechnical Institute of Bragança/Instituto de Telecomunicações

Campus Universitário de Santiago, 3810-193 Aveiro, Portugal

Tel: 351-234-377-900 E-mail: nuno@ipb.pt

Rui Sousa

University of Aveiro, DETI/Instituto de Telecomunicações

Campus Universitário de Santiago, 3810-193 Aveiro, Portugal

Tel: 351-234-377-900 E-mail: ruisousa@ua.pt

Paulo Salvador

University of Aveiro, DETI/Instituto de Telecomunicações

Campus Universitário de Santiago, 3810-193 Aveiro, Portugal

Tel: 351-234-377-900 E-mail: salvador@ua.pt

António Nogueira

University of Aveiro, DETI/Instituto de Telecomunicações

Campus Universitário de Santiago, 3810-193 Aveiro, Portugal

Tel: 351-234-377-900 E-mail: nogueira@ua.pt

Received: July 6, 2012 Accepted: November 17, 2012 Published: December 16, 2012

DOI: 10.5296/npa.v4i4.2058 URL: http://dx.doi.org/10.5296/ npa.v4i4.2058

Abstract

The increasing impact of the Internet in the global economy has transformed Botnets into one of the most relevant security threats for citizens, organizations and governments. Despite the significant efforts that have been made over the last years to understand this phenomenon and develop detection techniques and countermeasures, this continues to be a field with big challenges to address. Several approaches can be taken to study Botnets: analyze its source code, which can be a hard task because it is usually unavailable; study the control mechanism, particularly the activity of its Command and Control server(s); study its behavior, by measuring real traffic and collecting relevant statistics. In this work, we have installed some of the most popular spam Botnets, captured the originated traffic and characterized it in order to identify the main trends/patterns of their activity. From the intensive statistics that were collected, it was possible to conclude that there are distinct features between Botnets that can be explored to build efficient detection methodologies. Based on this study, the second part of the paper proposes a generic and systematic model to describe the network dynamics whenever a Botnet threat is detected, defining all actors, dimensions, states and actions that need to be taken into account at each moment. We believe that this type of modeling approach is the basis for developing systematic and integrated frameworks and strategies to predict and fight Botnet threats in an efficient way.

Keywords: Spam Botnet, statistical characterization, network security, malware, network resilience model.

1. Introduction

In the last years, communication networks have expanded their usage, importance and impact levels in the global economy. Nowadays, significant parts of our daily lives are directly or indirectly related with the Internet, with the use of services like the e-mail, online news or entertainment, teleworking, business transactions, home banking, social networks and much more. This importance and dependence degree raised this network to the level of a global critical infrastructure, where possible failures and disruptions have a tremendous impact in the global economy. However, in many aspects this new importance level was not accompanied by an increase of its reliability, availability and security [1] or, in other terms, resilience [2]. From the three disciplines that mainly characterize network resilience, security is the most challenging one.

The range of security threats that can affect Internet is immense and increasingly complex, reinforced with the beginning of a new era where cyber-war between nations is a reality. Network security is a very broad topic that includes issues like confidentiality, authenticity, integrity, authorization or non-repudiability. The lack of security on computers and networks is created, in a first instance, by the existence of vulnerabilities that can become a threat. Threats can become attacks, which can result in compromised systems. One of the most common security threats in current networks and computer systems is the use of software with malicious functionalities, known as malware. Malware is a generic term that encompasses specific malicious pieces of software like rootkit, virus, worm, spyware, trojan horse, sniffer and many others. A large set of infected computers (bots) that is remotely and coordinately controlled by an attacker (botmaster) is known as a Botnet. Although Botnets are used for many different malicious purposes, nowadays the most relevant uses are for political and financial benefits [3].

From a defender's perspective, it is very important to understand the trends and practices of Botnets. There are several approaches to study this phenomenon: analyze its source code, study the activity of its Command and Control (C&C) server(s), study the generated traffic by allowing a chosen machine to become infected by an executable bot and analyzing all possible scanning activities/actions triggered by the Botnet. In this work, we used this approach to characterize the traffic generated by each bot: we have installed some of the most popular spam Botnets by allowing the infection of a selected machine, captured the exchanged traffic and processed it in order to obtain relevant statistics that could be used to build characteristic patterns/behaviors for each Botnet. Grum, Cutwail and Bobax were the selected spam Botnets, mainly due to their activity importance level. The results obtained show that there are some distinct features between different types of spam Botnets (like, for example, the temporal evolution on the number of contacted peers or the variety of protocols involved in the communications between bots and C&C servers) that can be explored to build efficient detection methodologies. In fact, a deep understanding of some Botnets or Botnet types (and their corresponding behaviors) can be used to define heuristic rules, based on traffic statistics, in order to identify their activity and trigger subsequent actions to prevent generalized infections.

Whenever a Botnet is detected, it is necessary to deploy appropriate countermeasures that should limit the threat and/or eliminate it. Countermeasures can be grouped into three main categories: technical, regulatory and social methods [4]. If the identification of possible countermeasures that can fight and remove Botnet threats in a local network is nowadays reasonably well achieved, their systematic application needs to be significantly improved. Cleaning infected machines using anti-virus software, applying traffic filtering rules or blocking network elements' ports are relatively common measures taken by network administrators in the case of a Botnet detection. However, since these threats become more and more sophisticated, the fighting procedures need to be systematized and automated. Besides, having the ability to model all network states (from a security perspective) can help predict future network states/behaviors based on available (input) events. This systematization will obviously allow the deployment of automated countermeasures for any detected threat. So, this paper also proposes a generic network model that is able to describe the different network dynamics under the presence of a Botnet threat: all actors, dimensions, states and actions that need to be taken into account at each moment will be defined, allowing the development of appropriate inference procedures that can infer the values of different model parameters based on real data. So, the main goal is to define an exact model of the procedure flows that are necessary to cope with a machine or system infected or compromised by a particular Botnet type.

The paper is organized as follows. Section 2 presents the most relevant background on Botnet infrastructures, detection approaches and countermeasures; Section 3 describes the characterization methodology that was used in this work, presents the spam Botnets that were selected and the main obtained results, including a brief discussion on them; Section 4 presents the modeling approach, including all possible network states and all actions that originate state transitions, besides discussing the necessary steps to infer the model parameters and use it to help network managers and administrators; finally, Section 5 presents the main conclusions.

2. Background on Botnets

A Botnet is a large collection of computing systems that is infected with the same piece of malware (bot) and is remotely controlled by one or more attackers (botmasters), using a specific C&C infrastructure [1], with the purpose of performing malicious actions like sending spam email, triggering distributed denial-of-service attacks (DDoS), capturing private information or propagating other types of malware. Infected computers and networks become unstable and, frequently, unable to operate normally.

Nowadays, it is estimated that millions of infected systems exist in the Internet, being part of thousands of Botnets. According to Fossi et al. [5], the Rustock Botnet controlled more than 1 million bots. If in the last years economic benefit has been the major motivation for Botnet deployment, recently we are witnessing its increasing use for political purposes [6], [7] and for several underground cybercrime activities [3]: unsolicited mass mailing (SPAM), click frauds and pay per install, identity theft, DDoS.

2.1 Botnet infrastructures

The Botnet C&C infrastructure includes bots and a control entity, using an addressing mechanism and one or more protocols to maintain a communication channel and distribute commands between the infected computers and the botmasters [8]. The C&C infrastructure can have a centralized, decentralized or locomotive based architecture.

In a centralized architecture bots act only as clients, connecting and receiving commands from one or more servers. This architecture is based on a client-server communication model, where HTTP and IRC are the most common communication protocols. According to Fossi et al. [9], in 2009 HTTP was used on around 69% of all detected Botnets, while the remaining 31% where based on the IRC protocol, which continues to have an important role essentially due to its simplicity and ability to support a potentially unlimited number of participants commanded in parallel through a single channel. Private conversations are also supported, being possible to send commands to individual bots. The use of HTTP has obvious advantages: HTTP is a text-based protocol that uses easy to implement request-response methods and, due to its generalized utilization, avoids its traffic to be normally blocked by firewalls and anti-virus. Centralized infrastructures can be based on single central C&C servers or in a multilayered structure of servers and bots. In this second alternative, servers can be divided into different roles: some can be used for command and control and others for delivering contents to bots. Bots can also perform different roles in the Botnet structure.

In decentralized architectures, also known as peer-to-peer architectures, there is no differentiation between clients and servers. All nodes participating in the Botnet perform the same set of roles, being known as peers. The communication protocol is also based in peer-to-peer models. With this architecture, botmasters control bots by inserting commands and updates in an arbitrary point of the Botnet, which makes their localization almost impossible and provides a very high degree of anonymity. There are no central servers to mitigate and disable. However, the propagation of commands through the Botnet is slower when compared to centralized approaches. There are some Botnets that use hybrid infrastructures, with a centralized infrastructure as the primary option and an alternative peer-to-peer backup channel.

Locomotive Botnets use a central C&C infrastructure that is constantly moving over time. This means that the C&C servers are continuously changing, with the support of the DNS service.

A highly complex DNS-based technique was used by Botnet developers to increase its resilience and anonymity: the so called fast-flux service [3]. Fast flux is a DNS technique used by botnets to hide phishing and malware delivery sites behind an ever-changing network of compromised hosts acting as proxies. The basic idea behind this technique is to have numerous IP addresses associated with a single fully qualified domain name, where the IP addresses are swapped in and out with extremely high frequency, through changing DNS records. With this service, it is possible to use several bots as proxy servers to transparently

forward malicious communications from clients to a malicious server. The proxy servers hide to the outside the malicious services that are available in the malicious server. The main characteristic of this mechanism is the use of round-robin DNS with very short TTL values associated with the DNS resource records in order to rapidly and continuously change the IP addresses of the bot proxies, being extremely difficult to follow and intercept these communications.

2.2 Botnet detection and countermeasures

Since Botnets act with discretion, their detection is very challenging. One of the solutions that have been used for Botnet detection and tracking is based on honeynets [10], a set of honeypots. An honeypot is an intentionally insecure computational system that is placed in the network with the objective of detecting and capturing traffic from Botnets in order to understand their characteristics and *modus operandi*. The most important Botnet detection techniques that have been proposed are based on passive monitoring and analysis of the network traffic, and can be classified into four main categories [11], [12]:

- Signature-based: these techniques are based on previous knowledge about malware and Botnets. One known example is the Snort [13] tool, an open source intrusion detection system (IDS). The main drawback of this type of systems is that they can only detect known Botnets and malware.
- Anomaly-based: these techniques are based on the detection of traffic anomalies, like high volumes of traffic, high delay or jitter, unusual ports or unusual system behavior [14]. However, if the Botnet traffic seems to have normal patterns, this type of methods cannot detect it. Botsniffer [15] is an anomaly-based detection tool.
- DNS-based: these techniques apply the same principles of the anomaly-based techniques to the specific case of DNS traffic.
- Mining-based: since the other techniques are not effective to detect C&C traffic, this approach uses data mining techniques to perform this identification. In reference [16], Masud et al. presented a very promising data mining identification methodology.

When a Botnet is detected, it is necessary to do all the possible to mitigate the threat, taking measures to shut it down if possible. Because of the dissimulated nature of these systems, this is a challenging task. The most common approach is based on searching for central weak points in the Botnet infrastructure that can be disrupted or blocked. In general, two main approaches exist: classical countermeasures and offensive strategies [8]. In the classical countermeasures group, the three most common used techniques are:

- Taking down the C&C server. Whenever possible, this is the most effective and fast way to shut down the Botnet. However, it is only applicable to Botnets with a central infrastructure and if the location of the C&C server is known. The cooperation of the service provider where the server is connected to is fundamental in this step. Besides, depending on the Botnets, bots can be prepared to spread and perform tasks autonomously, without communicating with the C&C server.
- Sinkholing malicious traffic. If shutting down the C&C server is not possible, the traffic between bots and this server can be redirected to a sinkhole. This can be done at the

routing level, either in a local or global scale, obviously depending on the cooperation between organizations and ISPs.

- Cleaning infected systems. If this is the most sustainable measure to eliminate a Botnet threat, it is also the most difficult due to the extremely large spectrum of client systems that normally are infected, covering many different geographical areas, different types of users, etc. The most common approaches are based on the use of up-to-date anti-virus and personal firewalls in the end user systems. However, usually these tasks are not controlled by the network and system administrators, which make them so difficult to implement.

The effective implementation of classical countermeasures clearly depends on the organizational and political cooperation between different entities, which is usually a slow process when compared to the urgency that is required to fight these threats. Additionally, the most recent Botnet threats use increasingly sophisticated obfuscation techniques that make the application of classical countermeasures even more difficult. To solve these limitations, some new proactive offensive approaches have been proposed [8]:

- Mitigation: an offensive approach against the Botnet infrastructure, similar to temporary DoS attacks to C&C servers, trapping and blocking connections from infected machines or malicious domains.
- Manipulation: this approach relies on bugs found in bots to access the C&C channel, intercepting commands and forge new fake commands to change their behavior. In the limit, fake commands can order the bots self-destruction.
- Exploitation: this approach explores bugs in the C&C servers or even in the bots to gain control over them and promote their destruction from inside.

Despite being technically feasible and very effective, these types of techniques raise several ethical and legal questions, as the name (offensive) suggests. If fact, the use of these techniques usually implies the unauthorized access to infected machines and infrastructures, which means using the same (and many times illegal) rules as the attackers.

In [17] the authors proposed a new approach for collective cyber threat defense efforts based on the public health models that are used in several countries. In this proposal, authors defend the use of health certificates for all systems connected to the Internet. These certificates demonstrate the health condition of each device and can be used by service providers to allow or block access to specific resources (like home banking platforms, for example). Despite being an interesting theoretical approach, many practical questions need to be addressed in order to implement this model, ranging from the specification of certificates and protocols to the construction of a global infrastructure that can manage the system.

Another innovative approach is described in [18], where the authors propose the idea of using virtual bots to create uncertainty in the attack capacity of each Botnet. This study advocates that this uncertainty has a significant impact on the profits of botmasters and attackers, which means that the economic benefits can be destroyed or mitigated and the corresponding interest in using the Botnet will automatically decrease.

3. Analyzing Spam Botnets

3.1 Analysis Methodology and Selected Botnets

We have installed three popular spam Botnets by allowing the infection of a selected machine with the corresponding malware. The generated traffic was captured using wireshark [19] and processed a posteriori in order to obtain relevant statistics. Fig. 1 illustrates the data measurement/collection framework.

Since some bots have the ability to detect Virtual Machines (VMs), VMs were not used. The honeypot machine was always formatted before each infection, in order to prevent interference between the traffic generated by different Botnets. The operating system (OS) was Microsoft Windows XP Service Pack 3, since it is one of the most targeted OSs. In [20] it was possible to find malware for each one of the selected Botnets. The captures lasted for 48 hours, giving the possibility to better infer the behavior of each Botnet and observe its pattern in a long time basis. No other tasks were being performed on the infected computer while it was capturing traffic, in order to reduce any other generated traffic besides the one corresponding to the Botnet activity.

Several parameters/statistics of the traffic flows were collected/calculated, like for example: the flow starting/ending instant, the protocols involved, the number and type of active flags on TCP flows, the number of exchanged packets, the number of exchanged bytes, the flow duration, the contacted peers and their geographical location, the number of DNS queries and the periodicity of the communications.

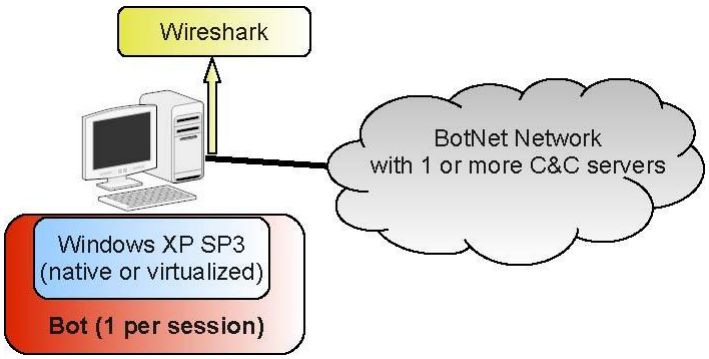

Figure 1. Framework for measuring Botnets traffic and inferring the most important statistics.

According to [21], the top ten spam Botnets in June 2011, in terms of percentage of spam, volume of spam per day, size and infected countries are the ones represented in Table I. Appropriate malware was obtained for three of these Botnets: Grum, Cutwail and Bobax.

Table 1. Top ten spam Botnets in June 2011.

Botnet	% of Spam	Volume of Spam Per Day	Estimated Size	Most Infected Countries
Cutwail	16.1	9.6 Billions	800K to 1.2M	India (10%), Russia (9%)
Xarvester	6.7	4 Billions	57K to 86K	UK (18%), France (13%)
Maazben	3.1	1.9 Billions	520K to 780K	South Korea (14%), Russia (10%)
Lethic	3.1	1.8 Billions	230K to 340K	South Korea (25%), Russia (15%)
Grum	3.0	1.8 Billions	200K to 290K	India (14%), Russia (14%)
Bagle	2.7	1.6 Billions	140K to 200K	India (15%), Argentina (8%)
Fivetoone	2.3	1.4 Billions	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 (8%)
DarkMailer	0.5	43 Millions	1K to 1.5K	France (27%), USA (16%)

Grum, also known as Tedroo, mainly focus its activity 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 [22]. It has five key features: a Kernel-based root-kit; reports to a C&C server via HTTP (Hypertext Transfer Protocol) on port 80; downloads plain text spam templates and address lists from a web-server; has multiple control servers and performs DNS MX lookups to send spam. This Botnet tries 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.

Cutwail, also referred as Pandex or Pushdo, among other names, has been active since 2007 and mainly focus in sending spam promoting pharmaceuticals, designer rip-offs or software. It also distributes malware regularly, sending emails attachments, usually .zip files. Nowadays, this Botnet also sends malicious campaigns, using social networking brands, distributing also phishing emails mainly targeting customers of several financial institutions. The main Cutwail's features are: reports to a C&C server on port 80, resorting to encrypted HTTP and performs DNS lookups to send spam and uses templates. The Cutwail behavior is described in detail in reference [22]: it connects to its control server using HTTP, through port 80, using an encrypted tunnel, and listens on a random UDP port for commands from its control server. At the host, it is able to download malware and, after installing it, it creates different processes, mainly with the purpose of notifying the Botmaster and running its commands.

Bobax, which can also be found under the name of Kraken or Oderoor, has been working since 2007 and has the following features: reports to control server using UDP, through port 447; uses dynamic domain name providers; performs DNS MX lookups to send spam; has multiple recipients per message; uses templates and 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 [22]. Like Cutwail, it creates processes to execute on Windows start-up, and hides its malware registering itself as a random service name. It has also the capability of searching for potential email addresses. After this process, it receives a template from the server to send to its targets.

3.1 Results Obtained

3.1.1 Grum Botnet

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. After this period, most of the traffic that was filtered as TCP Unknown used port 80, so it is in fact HTTP traffic. Traffic filtered as HTTP was generated during 150 seconds, also with a periodicity of 50 seconds, consisting of various GET requests for different types of files (.exe, .gif, .png). After some time, Unknown TCP packets (directed through port 445) were exchanged, corresponding to a communication of the Server Message Block (SMB) over TCP/IP. The objective of this activity was to find shared files. A Session Initiation Protocol (SIP) packet was then received, including information that is necessary to get options from an IP address. These 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 in the literature as a port used by virus or trojans. Some SMTP packets were also detected over time, reinforcing the idea of spam intents. The capture that was made followed a regular trend, with the vast majority of the packets belonging to HTTP and SMB. These packets continuously queried services through the NetBIOS Name Service (NBNS) and tried to establish sessions using the NetBIOS Session Service (NBSS). By looking at Fig. 2, we can clearly see 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. After that, besides unknown TCP, there are also some SMB and unknown UDP packets. In the Download direction, it is possible to observe few packets from three different protocols (HTTP, DNS and SMB), although this only happens in the first hour.

The number of generated packets (left part of Fig. 3) increased over time, being always higher in the download direction. There are peaks in the amount of generated traffic around hours 23, 37 and 43: in these peaks, it is possible to observe an increase of SMB session requests, as well as Remote Management requests. Grum generated a very limited amount of traffic, around 10 KB per hour. As expected from the amount of received packets, the amount of download traffic was also always higher than the amount of upload traffic.

We can see from the right part of Fig. 3 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 several attempts of TCP Session Establishment that were not successful. The session establishment attempts originated in the infected machine followed the expected behavior, since most of the packets generated in

response to SYN packets had the SYN/ACK flags active. However, the session establishment attempts received by the infected machine generated more packets with the RST/ACK flags active than packets with the SYN/ACK flags active, which means that the majority 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 a consequence of this high number of RST/ACK packets. In the 28th hour of the capture, a total of 84 RST/ACK packets were sent from the infected machine.

Finally, it is interesting to see the world map that geographically locates the peers that established communication with the infected machine. From Fig. 4, it is perceivable that Grum's infected machines are primarily located in Europe, Asia and America. The main infected countries were China and the United States of America.

3.1.2 Cutwail Botnet

For this Botnet, most of the packets that were filtered as Unknown TCP are also in fact HTTP packets. After a couple of hours, some HTTP/XML Notify packets were spontaneously exchanged. Some SIP Invite packets were also detected, together with some Telnet packets, but this activity is almost unperceivable and does not raise any suspicion. There were some NBNS Query packets as well, also using port 445, and a significant amount of SMB packets over TCP was also detected. Regarding Unknown UDP traffic, most of these messages were actually being exchanged for DoS attacks: in fact, many of the ports were recognized as the ones that are usually used for this type of security attacks. Around the third hour of the capture, a lot of Unknown TCP packets started being directed through port 50000. Although this port is known for being used by a trojan named SubSARI, this activity was related to protocol UPnP (Universal Plug and Play). By the end of the capture, there were some Remote Management packets. Once again, SMTP packets were detected but in a small quantity.

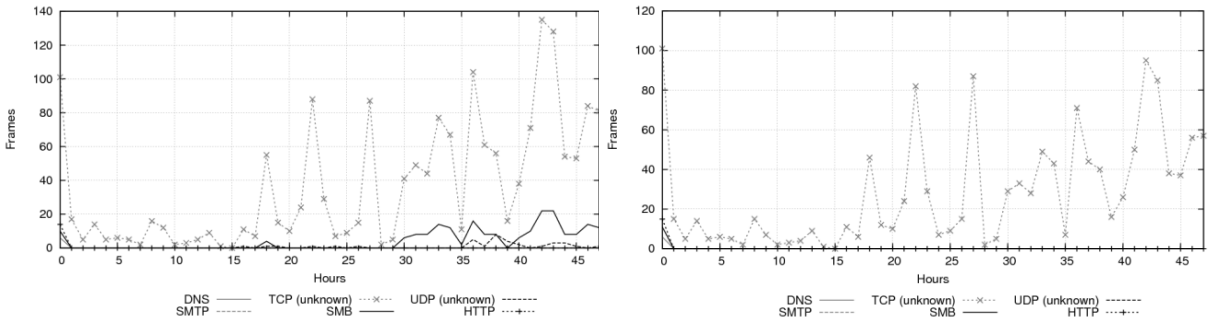

Figure 2. Grum Botnet protocols: (left) upload direction; (right) download direction.

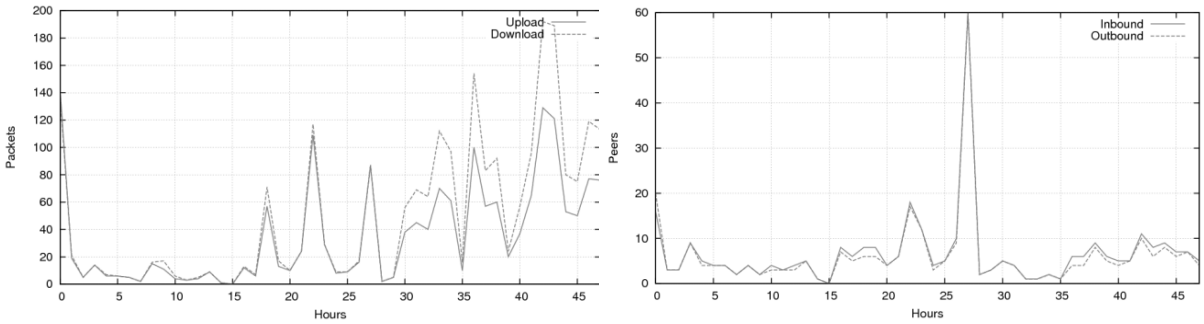

Figure 3. Grum Botnet statistics: (left) packets per hour; (right) unique peers per hour.

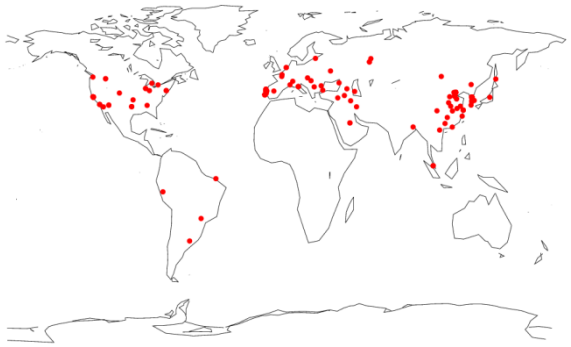

Figure 4. Grum Botnet: geographical location of the contacted peers.

The Cutwail Botnet exhibits a typical traffic pattern: it has a constant rate of sent HTTP packets and Unknown TCP packets, despite having a peak around hour 26, also follow a quite constant pattern. There is a small rate of SMB and Unknown UDP packets, which present a peak around the 25th hour due to a high number of TCP Session Establishments.

Protocols distribution in the download direction is quite similar to the upload distribution (Fig. 5), except regarding the absence of the peak in the number of Unknown UDP packets.

In the left part of Fig. 6 we can observe that both sent and received packets follow the same pattern, presenting a peak in the 26th hour. Once again, this peak is originated by several TCP Session Establishment attempts. There are almost always more Upload than Download packets. The right part of Fig. 6 represents the number of contacted peers per hour and we can see that there is a peak in the amount of contacted peers around hour 25, where the amount of contacted peers increases twenty times. This peak is also a result of the increase in the number of TCP Session Establishment attempts. It is also important to state that both captures contacted almost the same number of peers per hour, except at the moment of the peak occurrence. Most of the generated packets in response to SYN packets have the SYN/ACK flags active, although there are also RST/ACK packets, but in a very low number. However, there is a large number of unanswered SYN packets, which is not a common behavior and should be considered as an alarm for Botnet activity. This analysis corresponds to the session establishment attempts originated by the infected machine. When considering the session establishment attempts received by the infected machine, there are always more packets with

the RST/ACK flags active than packets with the SYN/ACK flags set, which means that most of the session establishment attempts were not successful. The peak around hour 25, which we have seen before, obviously conducted to this increase in the number of SYN and RST/ACK packets.

Fig. 7 shows that the infected machine communicated with hosts from all continents. The main infected ones are however Europe and Asia.

3.1.3 Bobax Botnet

Traffic from Bobax followed the same behavior throughout the whole duration of the capture. Immediately after the malware was installed, a lot of DNS queries were exchanged in port 1042, known for being used by trojans. Many of these queries were actually filtered as Unknown UDP packets. In the experimental capture we also observed a lot of SMTP packets, mainly in the first hour. Most of them were filtered as Unknown TCP. Some HTTP packets were also exchanged, and sporadically some HTTP/XML Notify messages. Regarding Unknown TCP packets, and performing a deeper inspection in order to understand their true origin, we can say that most of them are SMB packets, with HTTP being the second protocol in terms of the number of packets exchanged. 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 security attacks.

This Botnet definitely behave like expected, considering its amount of HTTP traffic, DNS lookups, DoS attacks and, essentially, SMTP packets. From Fig. 8, it is visible that most of the generated traffic was filtered as Unknown TCP. The Upload picture, despite showing mostly Unknown TCP traffic, also contains traffic from all the other protocols, although in a much less quantity. In the Download graph, it is possible to observe a clear pattern in DNS, SMB, SMTP, Unknown UDP and HTTP packets. Again, these protocols have relatively small number of packets when compared to the number of Unknown TCP packets. 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.

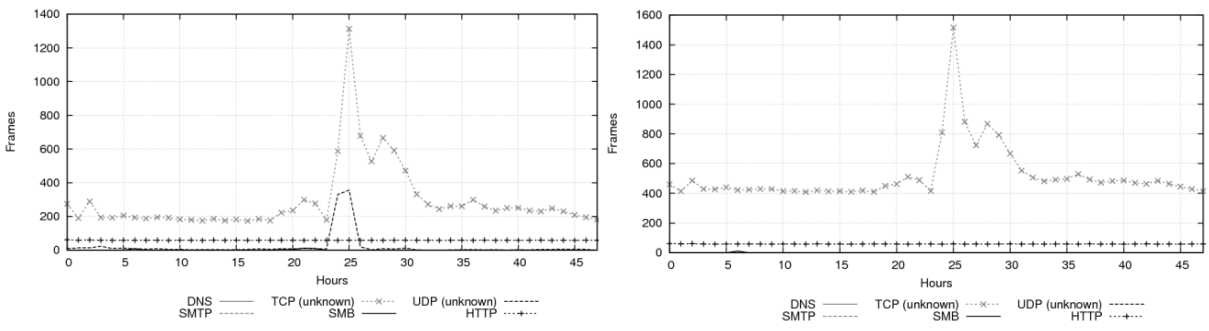

Figure 5. Cutwail Botnet protocols: (left) upload direction; (right) download direction.

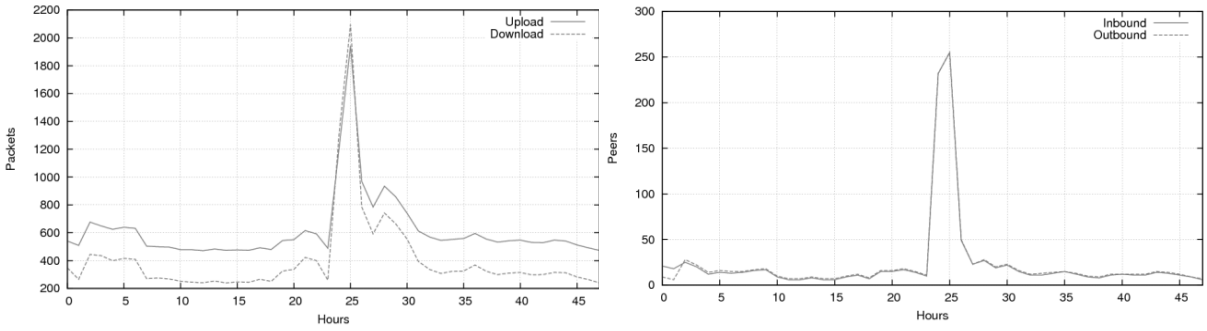

Figure 6. Cutwail Botnet statistics: (left) packets per hour; (right) unique peers per hour.

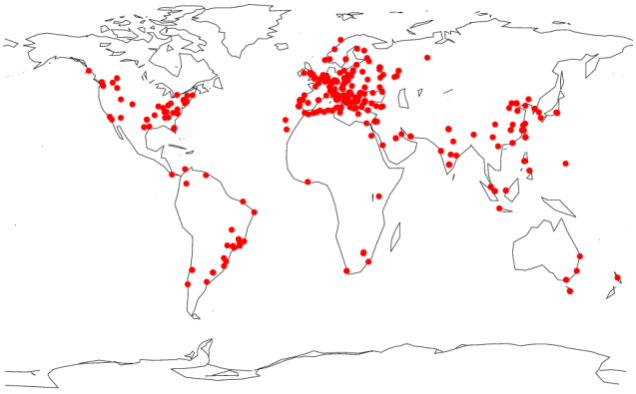

Figure 7. Cutwail Botnet: geographical location of the contacted peers.

The left part of Fig. 9 shows that the amount of packets per hour is a clear sign that we are facing Botnet generated traffic. This information should be an instant warning that should trigger measures to protect the infected machine or the network segment. It is also important to stress the difference between the number of Upload and Download packets: the amount of Upload packets is in the order of 470 thousand packets per hour, while Download packets are in the order of 40 thousand packets. Even in the number of contacted peers (right part of Fig. 9), we have a clear behavioral pattern. The observed values raise suspicions about Botnet infection, because they are in the order of 225 thousand contacted peers per hour. Most of the packets with the SYN flag active did not obtain any reply. Only a small number was replied with the RST/ACK flag set, and an even smaller number with the SYN/ACK flag.

Figure 10 shows that infected machines are located everywhere in the world. The most infected continents are Europe and America and the countries that suffer more infections are the United States of America and China.

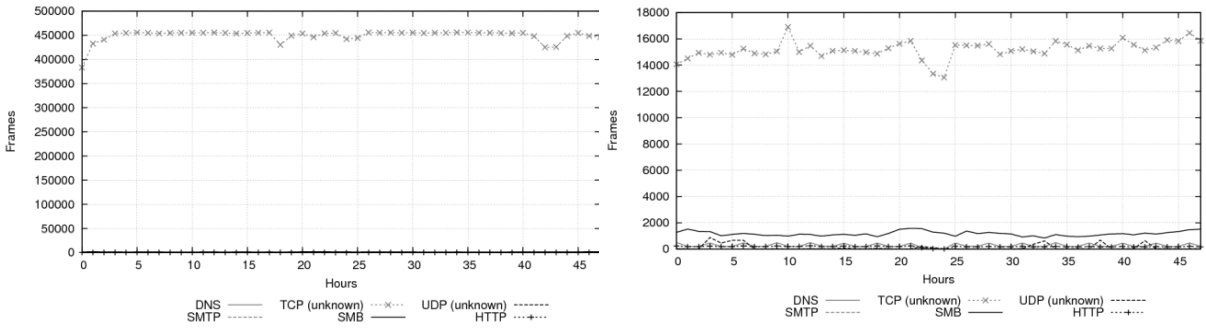

Figure 8. Bobax Botnet protocols: (left) upload direction; (right) download direction.

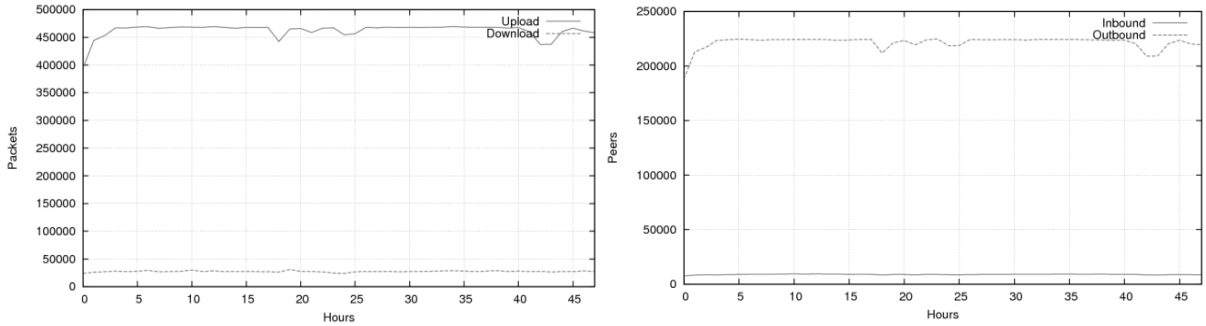

Figure 9. Bobax Botnet statistics: (left) packets per hour; (right) unique peers per hour.

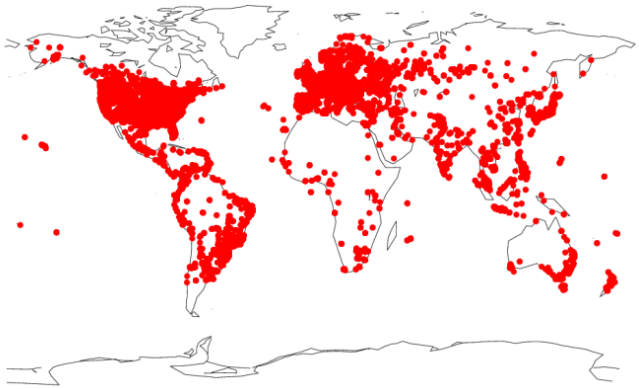

Figure 10. Bobax Botnet: geographical location of the contacted peers.

4. Modeling the Response

Although the identification of possible countermeasures that can fight and remove malware and spam Botnet threats in a local network is nowadays reasonably well achieved, their systematic and automated application needs to be significantly improved. Cleaning infected machines using anti-virus software, applying traffic filtering rules or blocking the ports of network elements are relatively common measures taken by network administrators in the case of a Botnet detection. However, since these threats are becoming more sophisticated, the fighting procedures need to be modeled and systematized, in order to increase their efficiency and scalability. This systematization will also facilitate the future deployment of frameworks that automate the countermeasures for any detected threat [23].

Thus, this section will present a generic network model that is able to describe the different network dynamics under the presence of a spam Botnet threat. All actors, dimensions, states and actions that need to be taken into account at each moment will be defined, allowing the development of appropriate inference procedures that can infer the values of the different model parameters from real data.

The model includes the description of the different network states, according to the degree of Botnet infection that is detected, and the actions that lead to state transitions. The finite state machine model that is proposed includes a detailed characterization of the possible states of all network elements (hosts, layer 2 devices, routers, etc), allowing a rigorous and precise knowledge of the network operation details at any given time instant. The nature of the proposed model allows its use in the prediction of the network states at future time instants.

Together with reliability and availability, security is one of the basic disciplines of network resilience. Under this context, security issues can be addressed using the two-phase ResiliNets strategy D^2R^2+DR, designed to improve network resilience in general [2]: the first phase of this strategy (D^2R^2) runs in real time and corresponds to the Defend, Detect, Remediate and Recover steps, while the second phase (DR) runs in background and includes the Diagnose and Refine steps. The model presented below is based on two basic assumptions: (i) network and host defenses can be broken and hosts can be infected by malware, becoming members of Botnets; (ii) current techniques and resources can detect the infection of hosts and the presence of Botnet activities in a local network. This means that this work will be focused in modeling the Remediate and Recover steps of the ResiliNets strategy, in the presence of Botnet threats.

4.1 The network model

The model considers a typical local network environment, where it is necessary to model the response behavior of the following actors: hosts, layer 2 devices (switches, wireless access points, etc) and routers. When considering the perspective of an individual host that is connected to the local network and can be infected by some piece of malware, becoming member of a spam Botnet, the following states and transitions can be identified:

- *Normal state* (hS1): the host is not infected with malware. The following transition action will affect this state:
  - *Malware infection* (hS1_a1): the detection of malware implies the change of the host to the Infected state.
- *Infected state* (hS2): some piece of spam malware was detected at the host. This state if affected by the following transition actions:
  - *Automatic clean system* (hS2_a1): if automatic defenses (e.g. anti-virus software) are able to fight this infection, the system can return to the Normal state;
  - *Filtering malicious traffic* (hS2_a2): if the defensive actions cannot automatically clean the system, the malicious traffic (i.e. spam messages) must be filtered in the local gateway/router and the host state will change to Quarantine.
- *Quarantine state* (hS3): if the infection cannot be automatically removed, the host

must be quarantined. This state can be changed by the following actions:

- *Manual clean system* (hS3_a1): if a manual cleaning of the system with existing tools (e.g. anti-virus) is successful, this implies the host transition to the Cleaned state;
- *Block all network traffic* (hS3_a2): if manual cleaning with existing tools is not possible and additional and more complex tasks are needed, the host transits to a disconnected mode, with the consequent blocking of all network traffic in the corresponding switch port or wireless connection.

- *Disconnected state* (hS4): if the infection cannot be controlled in a short time and is affecting the security and performance of other external elements, then the host must be temporarily disconnected from the network. This state can be changed only by the following action:
  - *Offline clean system* (hS4_a1): the system is cleaned with available tools and resources, definitively eliminating the threat. In some cases, a complete system formatting and re-installation should be necessary. If the action succeeds, the host transits to the Cleaned state.
- Cleaned state (hS5): after the quarantine or disconnected period, the host transits to the cleaned state, where all the previously applied contention measures are removed. The following action will change the system to the Normal state:
  - *Permit all network traffic* (hS5_a1): when the threat is definitively eliminated from the host, all the traffic filters that were previously activated can be removed and the host will transit again to normal operation state.

The finite state machine that represents all these states and transitions is represented in Fig. 11. The solid lines correspond to actions that occurred in the host while the dashed lines correspond to actions that occurred in other actors: hS2_a2 is applied at the local gateway/router, while hS3_a2 is applied at the layer 2 device where the host is directly connected to. Action hS5_a1 is applied in both the gateway and the layer 2 device.

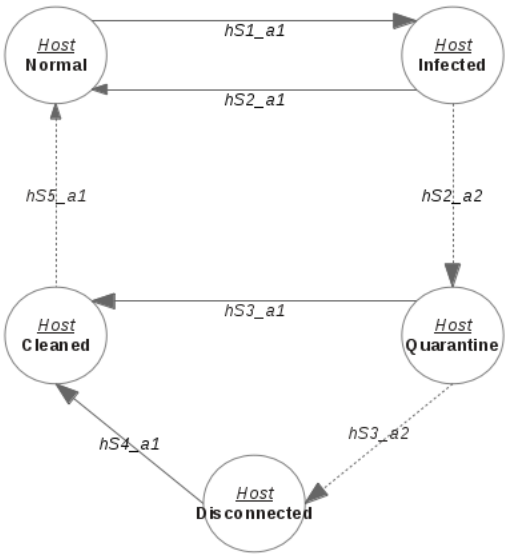

Figure 11. Finite state machine of an individual host.

In the same way, it is relevant to identify and characterize the states of layer 2 devices.

Since these devices physically interconnect network hosts, they represent the first point available to control the connect/disconnect tasks corresponding to each host. The states and relevant actions are:

- *Normal state* (lS1): if no actions are taken to disconnect a host from the network, all switch ports/wireless connections are in normal operation (enabled). The following action changes this state:
  - *Block first connection* (lS1_a1): if a first host connected to this device transits from the Quarantine to the Disconnected state, the corresponding switch interface or the wireless connection needs to be blocked (disabled), and the layer 2 device transits to the Blocking state.
- *Blocking state* (lS2): when the first host connected to this device transits from the Quarantine to the Disconnected state, the layer 2 device transits from Normal to Blocking state, disabling the corresponding switch port or wireless connection in order to block the physical connectivity for that host. This state remains active until no more switch or wireless connections are disabled due to this reason. The following actions occur in this state:
  - *Release connection* (lS2_a1): if a host connected to this device changes from Disconnected to Cleaned state and is not the last host in this situation, then this action is performed, restoring the corresponding physical connectivity. The layer 2 device remains in the same Blocking state until there are no more locally disconnected hosts;
  - *Block connection* (lS2_a2): if the layer two device is already in the Blocking state and a new locally connected host transits to Disconnected state, then this action is executed, blocking the corresponding switch interface or wireless connection;
  - *Release last connection* (lS2_a3): if the last host that was in the Disconnected state transits to the Cleaned state, then the corresponding connection is restored and the switch comes back to the Normal state.

Fig. 12 shows the finite state machine corresponding to the layer 2 devices.

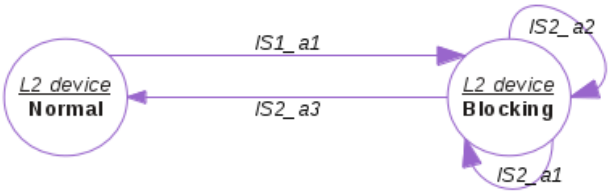

Figure 12. Finite state machine of the layer 2 devices.

The last relevant actor is the local gateway/router that interconnects different IP networks of the LAN. The states and actions that characterize this device are:

- *Normal state* (gS1): if no malware activities were detected in the local network, the router is operating in the normal state. The following actions will change its state:
  - *Add first host traffic filters* (gS1_a1): if the first malicious activities related with spam malware were detected in the local network and cannot be automatically removed, it is necessary to activate filters that can prevent malicious traffic from going outside. This action activates a filter rule to control the malicious traffic of the first local host that changed the state from Infected to Quarantine, making the router transit to the Filtering state;

- *Blocking all traffic* (gS1_a2): if an unexpectedly generalized contamination with spam malware is detected in the local network, this action immediately blocks all local traffic from going outside, changing the state of the router directly from Normal to the Blocking state.

- *Filtering state* (gS2): in this state, the router is filtering malicious (spam) traffic from infected local hosts and can be subject to the following actions:
  - *Remove host traffic filters* (gS2_a1): this action occurs when a quarantined host, which was not the last in this state, is cleaned. In this case, the filter rules corresponding to this host are no longer needed and are disabled;
  - *Remove last host traffic filters* (gS2_a2): this action occurs if the last quarantined host of the local network was cleaned. This implies changing the router to the Normal state;
  - *Add new host traffic filters* (gS2_a3): this action is performed when a new host is quarantined, activating new traffic filters for that host;
  - *Blocking all traffic* (gS2_a4): if the threat increased significantly and cannot be contained using only filters for malicious traffic, it can be necessary to activate more restrictive filters that block all traffic until the threat is controlled or eliminated. In this case, the router transits to the Blocking state.
- *Blocking state* (gS3): the router is in this state if one or more interfaces need to block all traffic. The device leaves this state by the influence of the following actions:
  - *Permit all network traffic* (gS3_a1): this action removes the filters that are blocking all traffic from one or more router interfaces. It is activated whenever the threats that previously implied the activation of these filters are definitively eliminated;
  - *Remove blocking all traffic* (gS3_a2): this action is activated if the generalized infection is controlled but not definitively eliminated. In this case, the rule that blocks all traffic in an interface is removed but the individual rules that filter malicious spam traffic for quarantined hosts remain active.

Fig. 13 presents the finite state machine corresponding to the router.

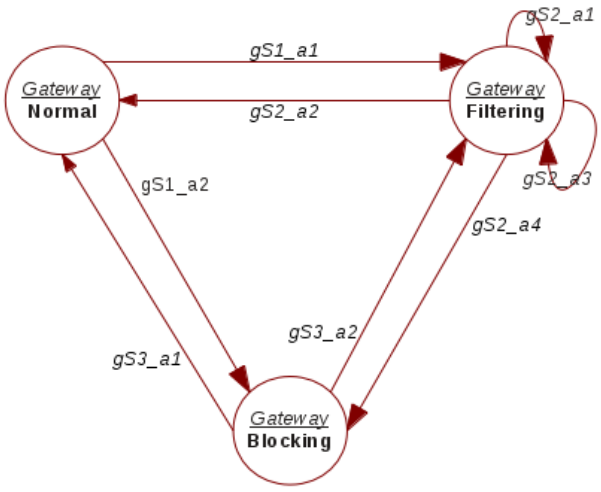

Figure 13. Finite state machine of the local gateway.

Considering that some of the actions taken to control and solve the host infection are implemented at the layer 2 device and/or the local gateway, Fig. 14 presents the interaction

between these three elements. The dashed lines represent transitions of an actor from one state to another caused by actions that occurred in another different actor. For example, the Host transits from the Infected to the Quarantine state by the effect of an action filtering malicious traffic that is applied in the Gateway through an action that adds host traffic filters.

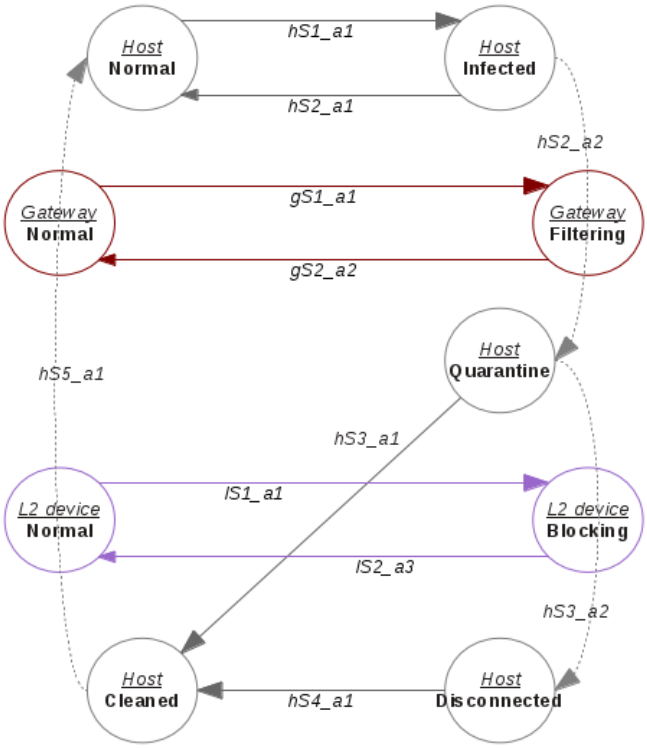

Figure 14. Interaction between host, router and layer 2 devices.

Following the previous analysis that considered the behavior of an individual host faced to a possible infection, it is now time to consider the states and behaviors of all local network as a set of hosts that can be infected with spam malware. Thus, we now identify the possible states, and associated actions, of the local network, when one or more local hosts are faced to a possible spam Botnet infection:

- *Normal state* (nS1): in this state, the network is working according to its baseline, without strange events originated by the presence of spam malware running on hosts. The transition to another states is affected by the following actions:
  - *Small Botnet Infection* (nS1_a1): if a Botnet infection is detected, with a small number of hosts transiting from Normal to the Infected state, the network changes from the Normal to the Low Infection state;
  - *Medium Botnet Infection* (nS1_a2): this action is performed when an infection suddenly affects a large number of hosts in the local network, transiting them from Normal to Infected state. As a result, the network state transits from Normal to Medium Infection state;
  - *Massive Botnet Infection* (nS1_a3): if an unexpected massive Botnet infection is detected, the network changes directly from the Normal to the Generalized Infection state;
- *Low Infection state* (nS2): a small number of infections on local hosts were detected but their impact in the overall network performance and security is not very significant. The

transitions that affect this state are:

- *Increased Botnet Infection* (nS2_a1): if the previously detected spam Botnet infection spreads to a significant number of hosts, the network changes from the Low Infection state to the Medium Infection state;
- *Recovery measures* (nS2_a2): the deployment of adequate recovery measures was able to eliminate the security threat, allowing the network to recover to the Normal state.

- *Medium Infection state* (nS3): in this state, a large set of local hosts is infected with spam malware and their activity impacts network performance.
  - *Remediation measures* (nS3_a1): the application of remediation measures, namely the filtering of malicious traffic by the router, results in an improvement of the network performance and transits the network to the Low Infection state;
  - *Increased Botnet Infection* (nS3_a2): the massive spread of the infected hosts forces the transition of the network state from the Medium Infection to the Generalized Infection state.
  - *Blocking all traffic* (nS3_a3): if the network performance is significantly degraded by the influence of the infected hosts, it can be necessary to immediately block all network traffic directed to outside, transiting the network to the Quarantine state.
- *Generalized Infection state* (nS4): a very significant number of hosts is infected, implying a big impact on the overall performance of the local network. The transitions from this state are affected by the following actions:
  - *Remediation measures* (nS4_a1): the deployment of remediation measures that confine the problem inside certain acceptable levels allow the network to return to the Medium Infection state;
  - *Blocking all traffic* (nS4_a2): if the generalized infection cannot be controlled within a certain limited amount of time, it can be necessary to transit the network state to Quarantine, blocking all network traffic directed to outside.
- *Quarantine state* (nS5): the previous detection of a generalized infection on local hosts (or an infection with a big impact on network performance) implied the quarantine of the network, blocking all traffic exchanged (in the router) with other IP networks. The transitions from this state are affected by the following actions:
  - *Remediation measures* (nS5_a1): the application of remediation measures can alleviate the degradation of network performance, allowing the transition to the Medium Infection state;
  - *Significant Remediation measures* (nS5_a2): in some cases, it may not be feasible to solve the complete infection in a short time, but the applied remediation measures can significantly recover the performance of the network and limit the damage of the infection. With this action, the network can move from Quarantine to Low Infection state.
  - *Recovery measures* (nS5_a3): the deployment of adequate recovery measures that definitively eliminate the threat, allowing the network to recover to the Normal state.

Figure 15 graphically represents the finite state machine of the local network, including the five states that were proposed and the transition actions between them.

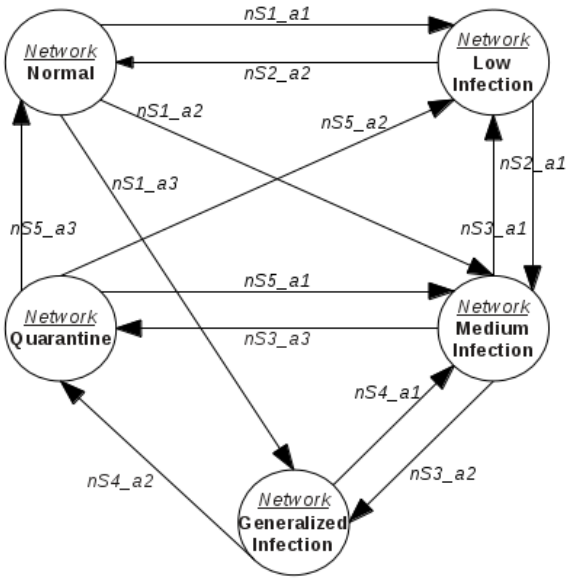

Figure 15. Finite state machine of the local network.

From this discussion, it is clear that the states and transition actions corresponding to the three identified actors are completely interrelated. Figure 16 tries to integrate the finite state machine of each element (local network as a set of hosts, layer 2 devices and local gateway), producing an overall picture of the system. The color scheme tries to give a visual idea of the network health, from the light (healthy) to the dark states (completely sick).

The knowledge of the real network state, influenced by the presence of Botnet activities, is fundamental to take the right decisions and apply the most effective countermeasures. This knowledge is only possible after inferring all the network model parameters from real and/or reliable network data.

4.2 From the inference of the model parameters to network management

In a first phase, network data reflecting normal activity and anomalous behaviors induced by the presence of different Botnet types should be collected, analyzed and correlated in order to understand which anomalies have occurred and how they can be characterized. The characterization of each anomaly should be as complete as possible, including the amount of data that is generated (alert messages, traffic amount on the different network links, anomalous information on log files, etc), the timing parameters associated to the anomaly (like, for example, the duration of its characteristic segments) and the transition probabilities between the different states that characterize the anomaly, among other relevant statistics. The data collection step should involve the deployment of laboratorial testbeds where the different security threats can be easily installed in a controlled environment, analyzed and characterized.

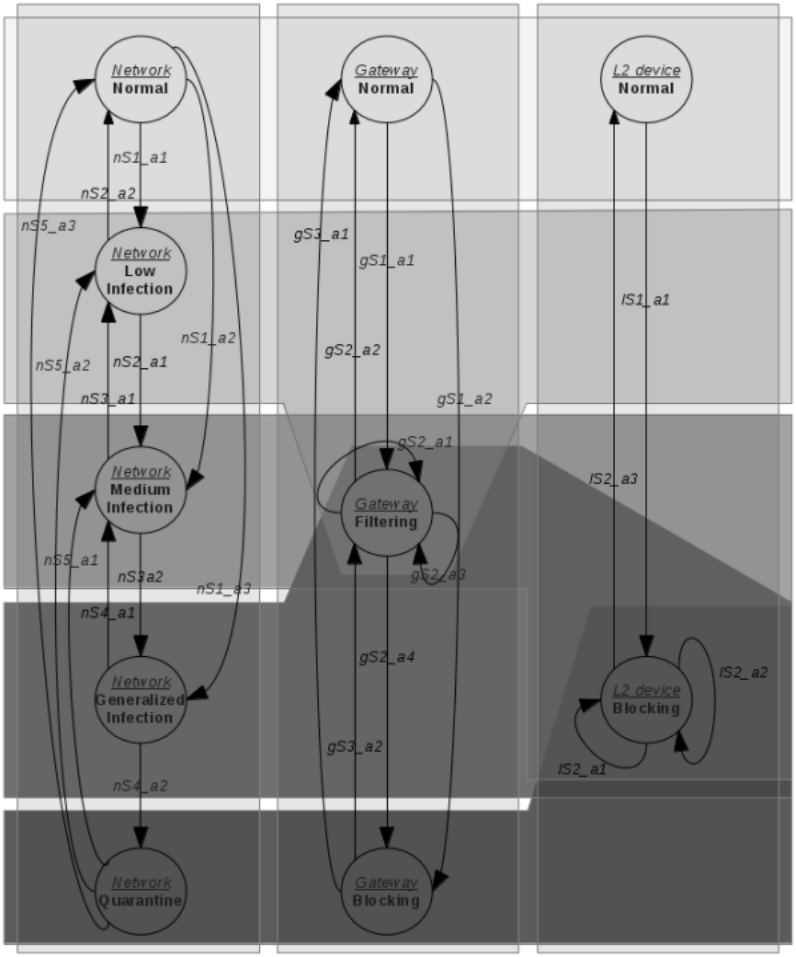

Figure 16. Finite state machine of the overall local network.

The network modeling framework is a multistage space state process able to model the number of error or alert messages and the different states of the network in terms of security threats. Each state is characterized by the type of generation process (deterministic, exponential or other) and its corresponding parameters. The dynamics of the state transitions are heterogeneous and can be ruled by deterministic or exponential processes that define the time of permanence in each state and the destination of the next transition. The modeling framework parameterization will agree with the assumption that state transitions can follow a deterministic or random distribution. State transitions are ruled in parallel by two (or more) parametric matrices that define, respectively, the next transitions after a deterministic amount of time and the probabilistic transitions after a random period of time. The probabilistic/random transitions can follow an exponential distribution (like happens in Markovian models) or any other distribution. The information generation processes associated with each state will also be parameterized by two (or more) vectors defining, respectively, the deterministic values and distribution function parameters for the rates and amount of alert messages generated.

The chain modulated nature of the modeling framework will allow the use of traditional mathematical tools to obtain the model resulting from the superposition of several models or

predict the network state at future time instants. The superposition of multiple models (corresponding to different independent networks or different network segments where a certain level of independence can be assumed) can be easily calculated using simple Kronecker sum and product operations. Besides, the chain nature of the resulting model will facilitate the prediction of future network states.

5. Conclusions

Among the different approaches that can be used to study the Botnet phenomenon, installing the appropriate malware, capturing the exchanged traffic and processing it in order to obtain relevant statistics that can be used to characterize each Botnet is of the most appropriate, being also immune to different types of restrictions. In this paper, we were able to characterize three relevant spam Botnets, Grum, Cutwail and Bobax, and the results obtained show that there are distinct features between them that can be explored to detect their activity. Having a systematic high value or abrupt peaks on the number of contacted peers was clearly an indication of a possible Botnet infection. Then, by looking to other statistics, such as the protocols distribution in the download or upload directions, the number of packets over time or the number of TCP packets with some flags (SYN, ACK, RST) set, it is possible to identify the specific Botnet that is active.

Since Botnet threats are becoming more sophisticated, new systematic models are needed to achieve significant improvements in fighting them. This paper proposed a network model that is able to describe all network states and the network dynamics in the presence of security threats, especially those originated from Botnets, being the first step for a more embracing objective, the development of an integrated framework that is able to identify threats and deploy appropriate counter-measures. All actors, dimensions, states and actions that need to be taken into account at each moment were defined, allowing the future development of appropriate inference procedures that can infer the different model parameters based on real data. Having the ability to model all network states (from a security perspective), events and transitions will be extremely important for network administrators and end users, helping them choose the most appropriate actions/countermeasures for each specific situation.

Acknowledgement

This research was supported by Fundação para a Ciência e a Tecnologia, under research project PTDC/EEA-TEL/101880/2008.

References

[1] S. Goodman and H. Lin, “Toward a Safer and More Secure Cyberspace”, Communications of the ACM, vol. 50, n. 10, 2007. http://doi.acm.org/10.1145/1290958.1290991

[2] J. P. G. Sterbenz, D. Hutchison, E. K. Cetinkaya, A. Jabbar, J. P. Rohrer, M. Scholler

and P. Smith, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines", Elsevier Computer Networks, vol. 54, n. 8, pp. 1245-1265, June 2010. http://dx.doi.org/10.1016/j.comnet.2010.03.005

[3] D. Plohmann, E. Gerhards-Padilla and F. Leder, "Botnets: Detection, Measurement, Disinfection & Defense", European Network and Information Security Agency (ENISA), 2011. Available at http://www.enisa.europa.eu/activities/Resilience-and-CIIP/networks-and-services-resilience/botnets/botnets-measurement-detection-disinfection-and-defence

[4] ITU Telecommunication Development Sector, "ITU Botnet Mitigation Toolkit", ICT Applications and Cybersecurity Division, Policies and Strategies Department, 2008. Available at http://www.itu.int/ITU-D/cyb/cybersecurity/projects/botnet.html

[5] M. Fossi, G. Egan, K. Haley, E. Johnson, T. Mack, T. Adams, J. Blackbird, M. Low, D. Mazurek, D. Mckinney and P. Wood, "Symantec Internet Security Threat Report: Trends for 2010 (Volume 16)", Symantec Corp, April 2011. Available at http://www.symantec.com/threatreport/archive.jsp

[6] S. W. Korns and J. E. Kastenberg, "Georgia's Cyber Letf Hook", 2009. Available at http://www.army.mil/article/19351/georgias-cyber-left-hook/

[7] N. Falliere, L. Murchu and E. Chien, "W32.Stuxnet Dossier, Version 1.4", Symantec Security Response, February 2011. Available at http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

[8] F. Leder, T. Werner and P. Martini, "Proactive Botnet Countermeasures: an Offensive Approach", The Virtual Battlefield: Perspectives on Cyber Warfare 3. pp. 211-225, 2009.

[9] M. Fossi, D. Turner, E. Johnson, T. Mack, T. Adams, J. Blackbird, S. Entwisle, B. Graveland, D. Mckinney, J. Mulcahy and C. Wueest, "Symantec Global Internet Security Threat Report: Trends for 2009 (Volume XV)", Symantec Corporation, April 2010. Available at http://www.symantec.com/threatreport/archive.jsp

[10] Honeynet Project and Research Alliance website (Online), 2012. Available at http://www.honeynet.org

[11] M. Bailey, E. Cooke, F. Jahanian, Y. Xu and A. Arbor, "A Survey of Botnet and Botnet Detection", Third International Conference on Emerging Security Information, Systems and Technologies, IEEE Computer Society, 2009. http://doi.ieeecomputersociety.org/10.1109/SECURWARE.2009.48

[12] M. Feily, A. Shahrestani and S. Ramadass, "A Survey of Botnet Technology and Defenses", Proceedings of the 2009 Cybersecurity Applications & Technology Conference for Homeland Security, IEEE Computer Society, 2009. http://dx.doi.org/10.1109/CATCH.2009.40

[13] Snort website (Online), 2012. Available at http://www.snort.org

[14] B. Saha and A. Gairola, "Botnet: An Overview", CERT-In White Paper CIWP-2005-05, 2005. Available at http://www.cert-in.org.in/Downloader

[15] G. Gu, J. Zhang and W. Lee, "BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic", Proceedings of 15st Annual Network and Distributed

System Security Symposium, 2008. Available at http://www.sciweavers.org/publications/botsniffer-detecting-botnet-command-and-control-channels-network-traffic

[16] M. Masud, T. Al-khateeb, L. Khan, B. Thuraisingham and K. Hamlen, "Flow-based identification of botnet traffic by mining multiple log files", Proceedings of International Conference on Distributed Frameworks & Applications, Penang, Malaysia, 2008. http://dx.doi.org/10.1109/ICDFMA.2008.4784437

[17] S. Charney, "Collective Defense: Applying Public Health Models to the Internet", IEEE Security & Privacy, vol. 10, n. 2, pp.1, 2011. http://dx.doi.org/10.1109/MSP.2011.152

[18] Z. Li and Q. Liao, "Botnet economics: uncertainty matters". In Managing Information Risk and the Economics of Security: 245-267. Springer, 2008. http://dx.doi.org/10.1007/978-0-387-09762-6_12

[19] Wireshark webpage, 2012. Available at http://www.wireshark.org

[20] Offensive computing webpage, 2012. Available at http://www.offensivecomputing.net

[21] Symantec Incorporation, "Symantec intelligence report: June 2011". Available at http://www.symantec.com/about/news/release/article.jsp?prid=20110628_01 .

[22] J. Stewart, "Top Spam Botnets Exposed", SecureWorks, 2008. Available at http://www.secureworks.com/research/threats/topbotnets/

[23] C. Landwehr, "Formal Models for Computer Security", ACM Computing Surveys (CSUR), vol. 13, no. 3, pp. 247–278, 1981. http://dx.doi.org/10.1145/356850.356852

Copyright Disclaimer