An Analysis of the Asprox Botnet

Ravishankar Borgaonkar
Technical University of Berlin
Email: rbbo@kth.se

Abstract—**The presence of large pools of compromised computers, also known as botnets, or zombie armies, represents a very serious threat to Internet security. This paper describes the architecture of a contemporary advanced bot commonly known as Asprox. Asprox is a type of malware that combines the two threat vectors of forming a botnet and of generating SQL injection attacks. The main features of the Asprox botnet are the use of centralized command and control structure, HTTP based communication, use of advanced double fast-flux service networks, use of SQL injection attacks for recruiting new bots and social engineering tricks to spread malware binaries. The objective of this paper is to contribute to a deeper understanding of Asprox in particular and a better understanding of modern botnet designs in general. This knowledge can be used to develop more effective methods for detecting botnets, and stopping the spreading of botnets on the Internet.**

Index Terms—**Asprox, Bot, Botnet, Fast-flux networks, Malware, SQL injection.**

I. Introduction

The term 'bot' is used to denote a computer that is infected by malicious code which often exploits software vulnerabilities on the computer to allow a malicious party commonly denoted as 'botherder' to control the computer from a remote location without the user's knowledge and consent. A network of bots constitutes a botnet which is a potent general purpose distributed supercomputer. Botnets represent a very serious threat to the Internet security [1] because they can be used to launch massive attacks against which there are no effective mitigation techniques or strategies. Botnet architecture consists of a pool of bots, a C & C (Command and Control) server and a botherder. The C & C server is sometimes referred as the mothership of a botnet. The botherder controls the botnet and uses it for illegal purposes. However botnets can be sold or rented out, so the botherder is not necessarily be the creator of a botnet. Bots are controlled by sending commands from the C & C server using different protocols like IRC (Internet Relay Chat) protocol [2], HTTP (Hyper Text Transfer Protocol) [3], P2P (Peer to Peer Protocol) [4], and FTP (File Transfer Protocol) [5]. Botnets are used as a vehicle for online crimes, and there are several illegal business models for making profit from it [6]. For example, botnets can be used for DDoS (Distributed Denial of Service) attacks, spamming, phishing, simply as a computing resource for rent, and for stealing users' credentials (identities, passwords, banking details etc.).

In order for a botherder to set up the botnets, there must be a combination of incentive and exploitable vulnerabilities. Incentives can be in terms of financial gain or political motives. Exploitable vulnerabilities may exist in the Internet infrastructure, in the clients and servers, in the people, and in the way money is controlled and transferred from the Internet into traditional cash. Many security firms and researchers are working on developing new methods to fight botnets and to mitigate against threats from botnets [7], [8], [9].

Unfortunately, there are still many questions that need to be addressed to find effective ways of protecting against the threats from botnets. In order to fight against botnets in future, it is not enough to study the botnets of past. Botnets are constantly evolving, and we need to understand the design and structure of the emerging advanced botnets. Learning from their creative designs could provide us new ways of understanding the modern botherders' tricks. Analysis of the advanced botnet can be helpful for the botnet defenders to develop mitigation tools and techniques against the botnet threat. Further botnet analysis process often reveals existing vulnerabilities in the operating systems and in the different applications that need to be patched. In this paper, we analyze and describe the Asprox botnet. Recent botnets are designed for propagating through SQL (Structured Query Language) injection attacks, exploits advanced fast-flux networks as a stealth technique to make tracing and shutting down process of the botnet more difficult. Asprox is a type of botnet that has these properties. Initially, Asprox was used as a password stealing Trojan and later upgraded to send phishing scams. Then in the year 2008, the Asprox botnet was modified to include an SQL injection attack tool and from then was used to attack a large number of legitimate websites.

This paper will focus on the design and structure of the Asprox botnet. In particular, we will investigate the C & C structure used by this botnet, the communication protocols, the drive-by download technique for spreading malicious content, and the advanced fast-flux service network. Later we discuss the weaknesses in the Asprox design and potential threats that can be expected in the next generation of botnets.

This paper is organized as follows. Section II describes a brief history of botnets and discusses the current trends. The main Asprox botnet features are described in Section III. Section IV explains the unique infection and spreading method of this botnet that made it the most innovative botnet of the year 2008. Weaknesses and potential future architectural botnet threats are discussed in Section V. Conclusions are presented in Section VI.

II. Asprox and other Botnets

Eggdrop which was created by Robey Pointer in 1993, was the first botnet that used IRC (Internet Relay Chat) as the C & C server [10]. Later many variants of IRC bots like Agobot, GTbot, SDbot, Spybot infected the Internet. However, as stated by Bitdefender Antivirus Company [11], NetBus and BackOrifice2K Trojans were first distinct malware breeds that also contained botnet-like features.

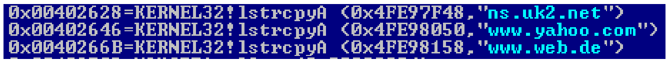
The Internet worms Lovesan, Sobig, Swen and Sobar represented a changing trend in virusology from mid 2003. These worms were used to exploit software vulnerabilities in MS windows, for connecting victims machine to the Internet, for DDoS attacks on websites, used for spammer techniques, and social engineering to distribute malware binaries. From 2003 to present we have seen many botnets with different architecture and features. Table I lists some well known botnets and their main features. The first Asprox variant appeared in 2003, and new advanced variants kept appearing until 2008 when it was a fast growing bot that infected a large number of hosts.

Botnets	Year	Infected Host	Architectural Features
Eggdrop	1993	–	IRC,First botnet
NetBus	1998	–	HTTP
BackOrifice2K	1999	–	IRC
Bagle	2004	–	HTTP
Spybot	2004	–	IRC
Strom	2007	85000	P2P ,fast-flux nw
Kraken[12]	2008	4,95,00	HTTP
Asprox	**2008**	**50,000[13]**	**HTTP, advanced fast-flux nw**
Conficker	2009	27,08,259[14]	P2P, fast-flux nw

TABLE I
History of Botnets

III. Asprox Botnet features

In this section, we describe important features of the Asprox botnet. Note that, we have dynamically and statically analyzed the malicious samples of Asprox botnet. We acquired these malicious samples and an analyzing tool (Norman SandBox Analyzer Pro) from Norman ASA [15] for our research purpose. Norman SandBox Analyzer Pro provides deep forensic analysis of executable code; in particular registers, memory, disassembled code, virtual hard disk, and network activity can all be closely monitored and manipulated in order to understand the full potential of the suspicious code. In order to analyze the malicious files of the Asprox botnet, it was executed in a Linux system (Ubuntu-8.04) using virtual machine environment. The environment includes a Sun's VirtualBox [16] application running Windows XP operating system. Figure 1 shows a snapshot of the malicious binary file that tries to connect to the Internet. However, while searching for other samples of the Asprox botnet on the Internet, we figure out that different names (given by various security companies) for the Asprox malicious samples makes a confusion. The same case we have been observed in the naming pattern of Conficker botnet. In future, we may need a unique naming standard or scheme for the bot samples that have to be found.



```
0x00402628=KERNEL32!lstrcpyA (0x4FE97F48,"ns.uk2.net")
0x00402646=KERNEL32!lstrcpyA (0x4FE98050,"www.yahoo.com")
0x0040266B=KERNEL32!lstrcpyA (0x4FE98158,"www.web.de")
```

Fig. 1. Asprox bot sample connecting to the websites

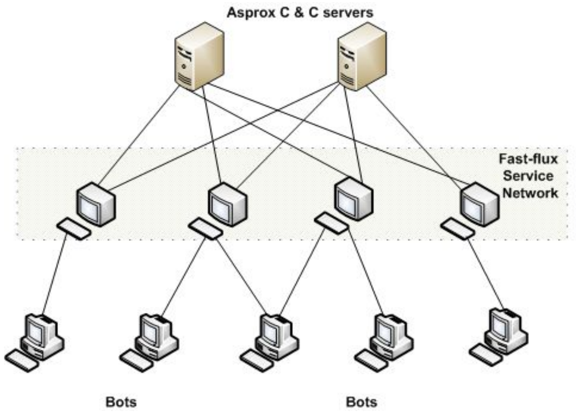
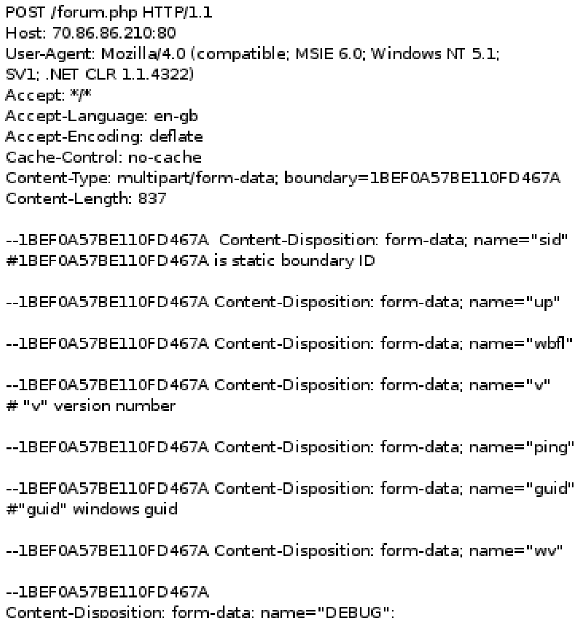


Fig. 2. Asprox Botnet Architecture

A. Centralized Command and Control Structure

Centralized command and control botnets follows traditional client-server paradigm. In the client-server architecture, clients requests contents or commands from a server. Centralized command and control structure botnets can be divided into being either a push-type or a pull-type botnet, depending on how bot herders send commands to the bots [17]. Asprox is a pull-type botnet in which the bot herder sets the command and relevant data in a file on the C & C server. The Asprox sample running on the bot machine tries to connect to some specific IP addresses. It sends authentication data in the form of forum-data post to the file '/forum.php' that resides on the server (having the specific IP addresses). Figure 3 shows the data part of the '/forum.php' file. Then the bot machine waits for further commands from the server and pulls a configuration file named COMMON.BIN from the C & C server. The COMMON.BIN file contains IP addresses of C & Cs, as well as the DNS related information and a malicious javascript file that is used to lure the users for drive-by downloads. The centralize architecture allow botherder to communicate with all bot machines instantly, compared to the peer-to-peer distributed structure. However once the C & C server goes offline, the centralize architecture might fail. To avoid of the service failure, Asprox uses advanced hydra fast-flux service network for providing high availability of the malicious content; thus protects the C & C server of the botnet. In section III-D, we discuss more about the fast-flux service networks. Figure 2 illustrates the centralized architecture of Asprox botnet.



```
POST /forum.php HTTP/1.1
Host: 70.86.86.210:80
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;
SV1; .NET CLR 1.1.4322)
Accept: */*
Accept-Language: en-gb
Accept-Encoding: deflate
Cache-Control: no-cache
Content-Type: multipart/form-data; boundary=1BEF0A57BE110FD467A
Content-Length: 837

--1BEF0A57BE110FD467A  Content-Disposition: form-data; name="sid"
#1BEF0A57BE110FD467A is static boundary ID

--1BEF0A57BE110FD467A Content-Disposition: form-data; name="up"

--1BEF0A57BE110FD467A Content-Disposition: form-data; name="wbfl"

--1BEF0A57BE110FD467A Content-Disposition: form-data; name="v"
# "v" version number

--1BEF0A57BE110FD467A Content-Disposition: form-data; name="ping"

--1BEF0A57BE110FD467A Content-Disposition: form-data; name="guid"
#"guid" windows guid

--1BEF0A57BE110FD467A Content-Disposition: form-data; name="wv"

--1BEF0A57BE110FD467A
Content-Disposition: form-data: name="DEBUG":
```

Fig. 3. forum.php post data

B. HTTP based Communication

Asprox botnet uses HTTP protocol for the communication between the C &C server and the bots. There are two types of web based botnets [18]. Asprox botnet is based on echo based botnet. In echo based type, bot announce their existence to the C & C by sending out a full URL (Uniform Resource Locator) to the web server. HTTP protocol is widely spread protocol over the Internet and most of the networks allow traffic on port 80. The HTTP protocol ensures existence of the bot to the C & C server. The vulnerable computer infected with Asprox binary frequently poll C & C servers via HTTP protocol. Figure 3 shows the pattern of the HTTP traffic between the C & C server and the bots.

As shown in the figure 3, Asprox bot uses port 80 as a outbound port, HTTP post static boundary ID, version number, and Windows guid. The bot replays the forum.php post data which is partitioned and tracked by GUID (Globally Unique Identifier). In addition, bot replays the post data for updating new C & C control servers list, new spamming or phishing campaigns related data, new binary version, and new fake AV(Antivirus XP2008) malware. Responses of forum.php contain stolen credentials of the user, bot's IP addresses, IP addresses of the C & C server, phishing page resources, and injected scripts [19].

C. SQL injection attack

SQL injection is a code injection technique used for maliciously exploiting applications that takes client supplied input data in the form of SQL statements. Attackers gain unauthorized access to a vulnerable database by supplying specially crafted string input that tricks the SQL engine to execute unintended commands. Figure 4 explains the SQL injection attack process. Asprox botherders used the trick to infect SQL server mostly serving .ASP (Active Server Pages) pages. Botherders automates the SQL attack vector to search potential SQL servers through Google search engine and then try to infect the server by inserting a malicious javascript file [20]. In 2008, infected machines started to download a SQL injection attack tool. A file named `msscntr32.exe` distributed by the Asprox botherders that act as a SQL injection attack tool. SQL injection attack is a web application attack vector that allows an attacker to alter the logic of running SQL query to run arbitrary commands on the vulnerable database server. The Asprox SQL injection tool first compromises a vulnerable website and then injects small javascript code into the server pages. The suspicious javascript code exploits application software vulnerabilities on the visitor's browser client that compromise the machine. Thus the compromised machine joins the Asprox botnet.

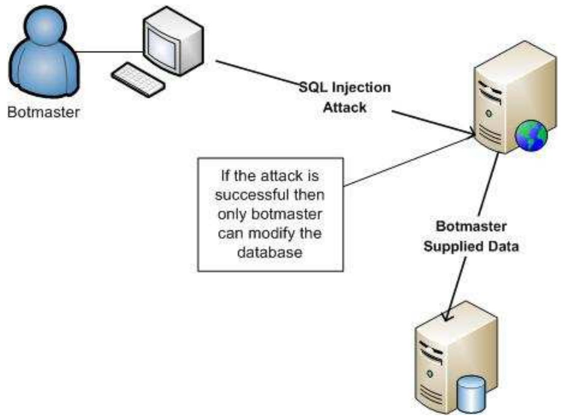
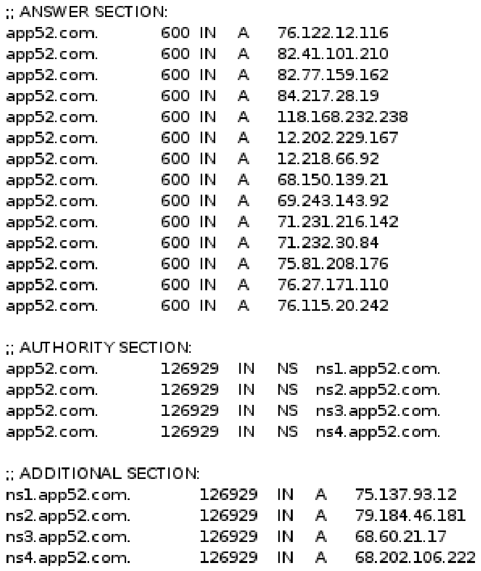

Botmaster
SQL Injection Attack
If the attack is successful then only botmaster can modify the database
Botmaster Supplied Data

Fig. 4. SQL Injection Attack

D. Use of Fast-flux Service network

Fast-flux is a technique in which A and/or NS resource records of a domain name changes rapidly and repeatedly in a DNS (Domain Name System) zone, thereby the location (IP address) of that domain changes rapidly when the domain name of an Internet host (A) or Name Server (NS) resolves. High-traffic websites use fast-flux technique to adapt addresses of their homepage according to internal and external network conditions, such as server load, outages, user location, and resource reconfiguration. However, cyber-criminals engaged in illegal activities (e.g. Phishing, Spamming, etc) use fast-flux technique to frustrate the efforts of investigators to locate and shut down their illegal operations. Storm botnet creators used such service networks first time effectively in 2007. Later Asprox botherders also utilized the fast-flux service networks in order to strengthen the botnet architecture. In particular, fast-flux service networks are networks of hijacked computer (that are part of a Botnet) systems with public DNS records that are constantly changing, with short time span [21]. The hijacked computers relay the illegal content from the botnet endpoint to a central server (or mothership of the botnet). The main aim of this technique is to provide high availability of the



```
;; ANSWER SECTION:
app52.com.          600  IN   A     76.122.12.116
app52.com.          600  IN   A     82.41.101.210
app52.com.          600  IN   A     82.77.159.162
app52.com.          600  IN   A     84.217.28.19
app52.com.          600  IN   A     118.168.232.238
app52.com.          600  IN   A     12.202.229.167
app52.com.          600  IN   A     12.218.66.92
app52.com.          600  IN   A     68.150.139.21
app52.com.          600  IN   A     69.243.143.92
app52.com.          600  IN   A     71.231.216.142
app52.com.          600  IN   A     71.232.30.84
app52.com.          600  IN   A     75.81.208.176
app52.com.          600  IN   A     76.27.171.110
app52.com.          600  IN   A     76.115.20.242

;; AUTHORITY SECTION:
app52.com.          126929   IN   NS   ns1.app52.com.
app52.com.          126929   IN   NS   ns2.app52.com.
app52.com.          126929   IN   NS   ns3.app52.com.
app52.com.          126929   IN   NS   ns4.app52.com.

;; ADDITIONAL SECTION:
ns1.app52.com.          126929   IN   A    75.137.93.12
ns2.app52.com.          126929   IN   A    79.184.46.181
ns3.app52.com.          126929   IN   A    68.60.21.17
ns4.app52.com.          126929   IN   A    68.202.106.222
```

Fig. 5. Asprox fast-flux service network

malicious contents by hiding location of the mothership (or in some cases, malware distribution server). Asprox uses fast-flux service networks to serve the content or commands to the bots globally. There are two types of fast-flux networks: single-flux network and double-flux service networks [21]. Asprox comes under the later that has an additional layer of protection by changing the IP address for the authoritative Namer Servers. Single-flux network only maps DNS records to IP address. Figure 5 shows an example of the double-flux service network where A and NS records of `app52.com` changes rapidly. In order to disrupt the double-flux service network, the particular domain name must be deactivated. However international border laws, different rules, and regulation of the domain name service providers restrict the deactivation process of such malicious domains.

However, fast-flux service network of the Asprox botnet differs from the typical double-flux service network. Main intention behind building such type of network is to maintain the best availability of the malicious content. The service network can be deactivated by shutting down the mothership of the particular botnet. However, in the Asprox botnet, the infected host downloads a list of IP addresses that belongs to the mothership. Therefore, by taking down a mothership from the network could not affect the communication of infected host with the end node (mothership); since the client has alternative IP addresses of the mothership to communicate that are also part of double-flux service network. Thus multilayer double fast-flux service network of the Asprox botnet hardens the efforts of law enforcement organizations and keeps the high availability of the malicious content. Figure 6 shows multilayer fast-flux network of the Asprox botnet, commonly referred as hydra-flux service network.

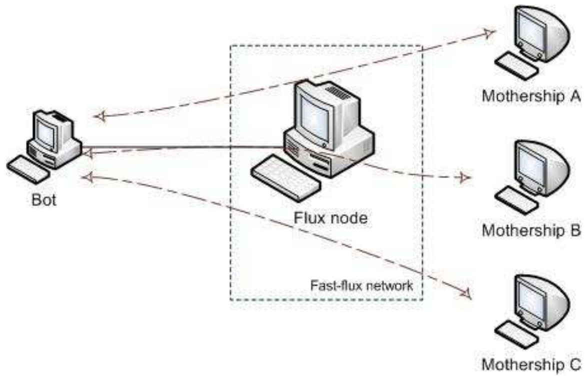


Fig. 6. Asprox hydra-flux service network

E. Use of Smart Social Engineering

The Asprox botherders fool the computer user into installing its malicious binary file. Botherder pretends such binary files as a real codec or software that needs to be installed. The Asprox botnet was responsible for spreading rogue Antivirus XP 2008 malware that was used for phishing and distribution of malicious bot files. For example, Botherders show 'spyware alert' message to the computer user and force to install (malicious) Antivirus XP 2008 antivirus . They use creative graphic images to lure the user. From the computer users IP addresses, botherder locates the location of the user and put curious messages, for example, *'powerful explosion burst in Oslo (place of the IP address) this morning that kills many people '*, and ask user to download latest flash player (which is bot's binary) to view the news. The naive computer user installs such malicious binaries in the form of various packages such as flash player and antivirus.

IV. Infection and Distribution Method

The Asprox botnet recruits new bots in a unique way, known as drive-by downloads method. As we discussed in the earlier section, SQL injection tool sends a query to `google.com` that search for the Microsoft IIS SQL server and the servers hosting mostly .ASP webpages. After receiving reply from `google.com`, SQL injection tool attacks on the potential vulnerable servers. If the attack is successful, the attack tool injects a javascript code containing a link for the malware hosting domain. Injected javascript redirects (the legitimate) website links to the server hosting malicious contents.

Figure 7 illustrates the infection method of Asprox botnet. In the figure 7, the infected machine gets new updates from the Asprox C & C server. The file named *"msscntr32.exe"* was responsible for the SQL injection attack. This attack is detailed as follows:

1) The infected machine sends queries to `google.com` using "msscntr32.exe" tool. In particular, the query searches for the websites hosting on Microsoft IIS SQL server and using .ASP pages.
2) The infected machines gets a reply from `google.com` containing a list of legitimate web servers including `website.com`.

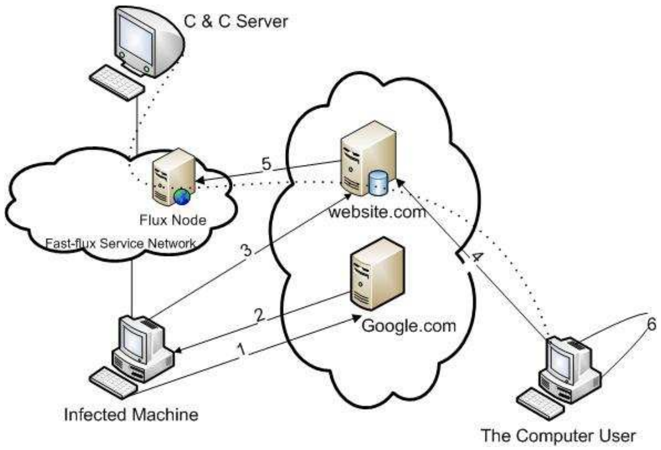


Fig. 7. Asprox Infection Method

3) The SQL injection attack tool attacks `website.com` using the SQL injection technique. If the server of `website.com` is vulnerable to this attack, then it injects a malicious code into the page of `website.com` by gaining the access of its server database.
4) The computer user tries to access the web service from `website.com`.
5) The request coming to `website.com` redirects automatically to a malicious server that hosts the website domain `malfluxdomain.cn`.
6) The fast-flux domain (server) `malfluxdomain.cn` prompts the computer user to install Asprox's malicious binary and become part of the Asprox botnet.

In the above attacking scenario, botherders changed their tricks. They first compromise legitimate websites to host the malware link rather than hosting malware on a newly registered domain name. Reason behind hosting malware link on the legitimate website could be

- The lack of security on old and popular legitimate websites [22].
- Since legitimate websites were old and running on older version of software with known vulnerabilities, thereby easy to compromise for botherders [23].
- In addition, user visits these websites frequently; thus no need to attract more users to download the Asprox malicious binary.

V. Possible Future Architectural Botnet Threats

In this section, we discuss about possible future architectural botnet threats that can be challenging to the Internet defense community.

Asprox botnet structure does not use strong cryptography. In the botnet architecture, authenticity and integrity of the bot commands is important. Some botherders use strong authentication and encryption mechanisms to protect the communication, however, these can be breakable. Botnet research community have not seen use of asymmetric cryptography in the botnet structure. Botherder can generate public/private key pair (of 2048 bits) and install the public key into bot's malicious binary. Thus botherders can able to sign the data using the private key. In the future, use of asymmetric cryptography can be challenging for botnet defenders.

Second potential feature in the Asprox architecture could be a Peer-to-Peer communication that overcome many of the problems of Asprox botnet having a centralized architecture, e.g., there will not be a single point of failure. Proprietary P2P module discussed in the design of Rambot botnet [24] can be more reliable and difficult to detect using advanced defense mechanisms.

Self-destruction function in the botnet can add an extra layer in the defense mechanism. Botherder could use such type of functions to destroy the users (bot's) operating system. Operating system of bot machine can be crashed by deleting registry entries in Windows and by cleaning the virtual memory. Researchers have seen such type of functions in the Zeus botnet [25]. However, crashing the operating system does not remove all the infection logs from the bot machine. The self-destruction process might force the user on (the infected machine) to reinstall new operating system, thereby botherder could try to block the user from submitting the malicious binary file to the Antivirus firm or the security research organization.

Tor, based on Onion Routing, can support anonymous communications over public networks by providing near real-time and bi-directional anonymous TCP connections that are resistant to both eavesdropping and traffic analysis attacks. Tor gives privacy to the user by adding perfect forward secrecy, congestion control, directory servers, integrity checking, configurable exit policies and a practical design for location hidden services via rendezvous point [26]. In future, we might see botherder using the Tor architecture features for setting up the botnets in order to be anonymous on the Internet and to harden the botnet traffic detection process.

IPv6 protocol can be misused to deliver a malicious binary file or to send instructions to the bots. Malware tunneling can be possible using the auto-configuration feature of IPv6 [27]. Tunneling commonly referred as a method of relaying private data over the public Internet. Botherders can configure the infected machine to allow IPv6 traffic and use this [28] novel approach to construct the covert channel that can be used for the malicious purpose. Though system administrators are aware of the IPv6 autoconfiguration feature, most firewall and IDS (Intrusion Detection Systems) are not configured to filter the IPv6 traffic.

VI. Conclusion

Botnet represents a very serious threat to Internet Security. Asprox combines two threat vectors- forming a botnet and generating SQL injection attack. In this paper, we have analyzed architecture of Asprox, the botnet having advanced features such as hydra fast-flux network, use of SQL injection attack tool, use of drive-by download method to recruit new bots, and smart use of social engineering tricks. However, use of potential future botnet architectural threats such as use of strong cryptography, use of self-destruction functions, use of onion routing technique or Tor architecture, and malware tunneling through IPv6 can be challenging for the botnet defense community. In future, network security design could be based on the different mechanism used by modern botnets.

Acknowledgement

The authors would like to thank Mr. Hans Christoffer Gaardls Hansen, virus analyst, Norman ASA for providing a tool called 'Norman SandBox Analyzer Pro' and for providing useful comments.

References

[1] Cooke E., Jahanian F., and McPherson D. 2005. The Zombie roundup: understanding, detecting, and disrupting botnets. In Proceedings of the Steps To Reducing Unwanted Traffic on the Internet Workshop, USENIX Association, Berkeley, CA, pp 39-44.

[2] J. Oikarinen and D. Reed. Internet Relay Chat Protocol. Network working group, request for comments, May 1993, RFC-1459. {Online} http://www.ietf.org/rfc/rfc1459.txt.

[3] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinte, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol. Network working group, request for comments, June 1999, RFC-2616. {Online} http://www.ietf.org/rfc/rfc2616.txt.

[4] G. Camarillo, Ed. Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability. Network working group, request for comments, November 2009, RFC-5694. {Online} http://tools.ietf.org/html/rfc5694

[5] J. Postel and J. Reynolds. File Transfer Protocol. Network working group, request for comments, October 1985, RFC-959.{Online} http://www.ietf.org/rfc/rfc959.txt.

[6] Ianelli N. and Hackworth A. Botnets as a Vehicle for Online Crime. {Online} http://www.cert.org/archive/pdf/Botnets.pdf, December 2005.

[7] Masud Mohammad M., Gao Jing , Khan Latifur, Han Jiawei, and Thuraisingham Bhavani. Peer to peer botnet detection for cyber-security: a data mining approach. CSIIRW '08: Proceedings of the 4th annual workshop on Cyber security and information intelligence research, 2008,978-1-60558-098-2,pages = 1–2,Oak Ridge, Tennessee, ACM, New York, NY, USA.

[8] Abu Rajab M., Zarfoss J., Monrose F., and Terzis A.. A multifaceted approach to understanding the botnet phenomenon. In Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement , IMC '06. ACM, New York, NY, 41-52.

[9] Felix C. Freiling, Thorsten Holz, Georg Wicherski. Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks, In Proceedings of 10th European Symposium on Research in Computer Security, ESORICS, September 2005.

[10] John Canavan. The Evolution of Malicious IRC Bots. From the proceedings of the VB2005 Conference.

[11] Bogdan Botezatu. Botnet:10 Years of Security Threats.{Online} http://www.malwarecity.com/blog/ botnet-10-years-of-security-threats-227.html. Last visited, April 2010.

[12] Pedram Amini. Kraken Botnet. {Online} http://dvlabs.tippingpoint.com/blog/2008/04/28/kraken-botnet-infiltration. Last visited, April 2010.

[13] Brian Prince. Phishers bite back with malware exploits linked to keywords. {Online} http://www.eweek.com/c/a/Security/Phishers-Bite-Back-With-Malware-Exploits. Last visited, April 2010.

[14] Conficker Working Group. {Online} http://www.confickerworkinggroup.org. Last visited, April 2010.

[15] Norman ASA. Norman Sandbox Analyzer Pro. {Online} http://www.norman.com/enterprise/all_products/malware_analyzer/norman _sandbox_analyzer_pro/en. Last visited, April 2010.

[16] Sun Microsystems. VirtualBox. {Online} http://www.virtualbox.org/wiki/VirtualBox. Last visited April 2010.

[17] G. Gu, J. Zhang, and W. Lee. BotSniffer: Detecting botnet command and control channels in network traffic. Proceedings of the 15th Annual Network and Distributed System Security Symposium NDSS08 (2008).

[18] C.A. Schiller. Botnet:Killer Application Botnets, 2007 , Syngress Publication.

[19] Dennis Brown. http://www.toorcon.org/tcx/18Brown.pdf. Last visited, November 2009.

[20] Symantec. White Paper: Web Based Attacks, February 2009. {Online} http://www4.symantec.com/Vrt/wl?tu_id=ZAW@123663933427236001. Last visited, April 2010.

[21] The Honeynet Project. Know Your Enemy: Fast-flux Service Networks, 2007. {Online} http://www.honeynet.org/papers/ff. Last visited, April 2010.

[22] SPAMFIGHTER Article. {Online} http://www.spamfighter.com/News-12417-Hackers-Target-Legitimate-Websites-to-Host-Malware.htm. Last visited, April 2010.

[23] Dan Raywood. SC Magazine Article. {Online} http://www.scmagazineuk.com/Legitimate-websites-are-hosting-most-of-the-web-based-malware-due-to-poor-security-measures/article/136883/ . Last visited, April 2010.

[24] R Hund, M Hamann, T Holz. Towards Next Generation Botnets. The fourth European Conference on Computer Network Defense, EC2ND 2008.

[25] Zeus Botnet. {Online} http://www.abuse.ch/?p=1327, Last visited, April 2010.

[26] R. Dingledine, N. Mathewson, and P. Syverson. Tor:the second-generation onion router. In Proceedings of the 13th conference on USENIX Security Symposium, pages 303-320, Berkeley, CA, USA, 2004.

[27] US cert Publications. {Online} http://www.us-cert.gov/. Last visited, April 2010.

[28] Janne Lindqvist. IPv6 Stateless Address Autoconfiguration Considered Harmful. In Proceedings of the Military Communications Conference, MILCOM 2006, Washington, D.C., USA, October 23-25, 2006.