http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (23): 7152-7159, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.7152.7159

A Systematic Approach to Capturing and Analysing Botnets

[1,2,3]Jiang Wei and [4]Tian Zhihong
[1]College of Computer Science, Beijing University of Technology, Beijing, 100124, Peoples' Republic of China
[2]School of Computer, National University of Defense Technology, Changsha, 410073, Peoples' Republic of China
[3]Key Laboratory of Information and Network Security, Ministry of Public Security, Shanghai, 201204, Peoples' Republic of China
[4] School of Computer Science and Technology, Harbin Institute of Technology, Haerbin, 150001, Peoples' Republic of China

Abstract: In this study, a systematic approach to Capturing and Analysing botnets is presented. Our framework is a scalable and robust infrastructure and consists of four modules. Honeynet-based capture system is proposed that is the first step towards our framework, which can automatically and dynamically collect and analyze malware traffic over the Internet without supervision. Furthermore, a multidimensional analysis system is designed to analyze binaries captured in the capture system. In addition, we discuss our preliminary results and lessons learned from this work.

Key words: Network security, botnet, attack taxonomy

INTRODUCTION

Internet has been plagued by a variety of security threats over the past several years. One of the biggest threats to the Internet is the presence of botnets. Botnets are often used to launch various attacks, ranging from Distributed Denial-of-Service attacks to e-mail spamming, key-logging, click fraud and spreading new malware. Defending networks against botnet attacks is an urgent task. And it becomes a hot issue in network security and cyber crime research communities.

To prevent attacks causing by botnets, it is therefore possible to identify, infiltrate and analyse botnets and to stop it in an automated fashion. Having more information on botnets could help us to mitigate the threat botnets pose.

Honeypots and honeynets are effective detection and defense techniques at a reasonable cost and without false positives and hence there has been much recent research in this area. The Honeynet project deploys an architecture that consists of a Honeywall and the honeypot network. Honeywall performs access controlof outbound connections from the honeypots and captures network data. The network behind Honeywall consists of high-interaction honeypots without emulation. Chinese Honeynet Project presented a malware collection tool based on the high interaction honeypot principle called HoneyBow (Zhuge *et al*., 2007). Bailey *et al*. (2004) presented a globally distributed, hybrid, honeypot-based monitoring architecture which deploys low-interaction honeypots as the frontend content filters and high-interaction honeypots to capture detailed attack traffic for scalable network monitoring. Tang and Chen presented a novel "double-honeypot" detection system to effectively detect Internet worm attacks (Tang and Chen, 2005).

There have been several works (Rajab *et al*., 2006; Freiling *et al*., 2005) on botnet tracking using honeynets. In Rajab *et al*. (2006), the authors construct a multifaceted and distributed measurement infrastructure to clear the fog surrounding botnets. In Freiling *et al*. (2005), an approach is presented to DDoS attack prevention that is based on the observation that coordinated automated activity by many hosts needs a mechanism to remotely control them. In Dagon *et al*. (2005) introduced taxonomy of botnets to provide a response to botnets by degrading or disrupting them, which involved discovery and proactive attack to the botnet. In Paxton *et al*. (2007), the authors introduce their Honeynet-based bot analysis architecture which is the first step towards our Risk-Aware Network-centric Malware Detection and Prevention Framework.

Corresponding Author: Jiang Wei, College of Computer Science, Beijing University of Technology, Beijing, 100124, Peoples' Republic of China

Present study is focused on an in-depth understanding of the botnet behaviours and proactive defence against botnets. In this paper, we describe our framework in large-scale network which includes automatically capture bots, comprehensive analysis them and in-depth monitoring them by connecting with their command and control center.

Present study is different from the above research works in several aspects. First, we present a comprehensive framework to capturing and analysing botnets in large-scale network. Second, a multidimensional, longitudinal capture and analysis approach is presented. This approach integrates information learned from multiple data collection mechanisms. These work is the first step towards our malware detection and proactive defence Framework.

The rest of the study is organized as follows. Section II discusses background technologies. The CAMB approach is presented in Section III. In Section IV, we discuss our analysis method along with the preliminary results. Section V concludes this paper with a brief description of future work..

BACKGROUND TECHNOLOGIES

Relate definitions: Before further discussion, let's first introduce and explain a few botnet terminologies which are used in the paper. A Botnet is a network of compromised computers under the control of a remote attacker [10]. Botnets consist of botmaster, bot, bot client and command and control channel (C and C). Botmaster is the attacker who controls the malicious network. Bot is a compromised computer under the botmaster control (also called zombies, or drones). Bot Client is a malicious trojan installed on a compromised machine that connects it to the Botnet. Command and control channel (C and C) is the communication channel that the Botmaster uses to remotely control his or her bots.

Botnet evolution: In order to watch a single channel, Robey Pointer wrote a legitimate IRC bot, Eggdrop[10], in 1993, which was well-known and widely used of the time as non-malicious IRC bot.

However, the usage of bots is not limited to good purpose only. The first malicious bot is PrettyPark worm [10], appeared in 1999. The PrettyPark worm contained a limited set of features, such as the ability to connect to a remote IRC server, retrieve a variety of information about the system. And it makes use of IRC as a means to allow a botmaster to remotely control a large pool of compromised hosts.It also had a basic update mechanism which allowed it to download and execute a file from IRC.

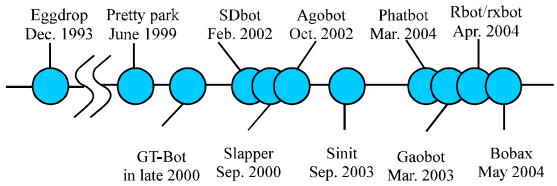

Fig. 1: Timeline of botnet evolution

Using some malicious scripts and a number of legitimate tools, GTbots began to appear in the wild in late 2000. It wasn't until the start of 2002 that the Sdbot began to appear. SDBot were found in the October, 2002 time frame. There are now hundreds of variants of this code that provide a wide range of capabilities. Phatbot is an early find peer-to-peer bot based on WASTE. Fig. 1 shows the time line of botnets evolution.

- **Infection mechanism:** An attacker may adopt any of the following techniques or a combination of the techniques for wide distribution of a particular bots
- **Web download:** Attackers can hack Web sites and install bots that infect surfers' vulnerable browsers
- **Mail attachments, Attachments:** E-mail attachments with mass-mailing worms can carry bots. Spam techniques simplify and enable fast spreading of such malware easily
- **Automatically scan exploit and infection:** The bots automatically infect the machines that have the backdoors or other vulnerabilities that the software was written to exploit via virus, worm, or Trojan horse components

Figure 2 illustrates a typical bot-infection process. Botnets usually commandeer new victims by remotely exploiting a vulnerability of the system running on the victim via an e-mail attachment, infected Web site, or other means.

When the infection has taken place, the victim executes a script and downloads bot binary from some location. The bot binary installs itself to the victim and it will start automatically when the victim is rebooted. The new bot will establish an IRC session with the server and join the command and control channel specified in the bot binary. Once the bot successfully joins the specified IRC channel, it automatically parses and executes the channel topic, which contains the default commands. Commands are sent to infected PC from bot master via IRC server (s). Infected PC executes commands by, for example, launching a distributed denial-of-service attack, sending mass spam mailings, or logging keystrokes.

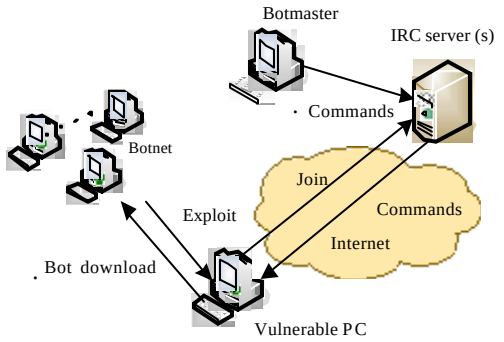

Fig. 2: Process of a typical botnet infection

Command and control (C and C) mechanism: C and Cs are important for botnets and are unlikely to change among bots and their variants. However, the C and Cs are the weakest link in the operational aspect of botnets. Therefore, understanding the C and C mechanism in botnets has great value for us defending against botnets. C and C communication mechanism can be identified three possible topologies (Cooke *et al*., 2005).

Centralized C and C model: The botmaster chooses a single compromises host to be the contacting point (C and C server) of all the bots. When a new computer is infected by a bot, it will join the botnet by initiating a connection to the C and C server. The centralized model has some advantages such as simple implementation and customization. However, the centralized C and C model has some significant drawback. For example, it will be detected and destroyed easier. Although this model hascertain drawbacks, most bots use the centralized C and C model such as AgoBot, SDBot and Zotob.

P2P-based C and C model: For drawbacks of centralized model, some botnet authors have started to build alternative botnet communication systems. And botmaster are shifting to P2P-based botnets. Compared with the centralized C and C model, the P2P based C and C model is much harder to discover and destroy. Botherder can send commands from any peer. However, it is a more complex job for designing p2p systems. And other peers can potentially take over the botnet Some variants of Phatbot have used P2P communication as a means to control botnets.

Random C and C model: In Cooke *et al*. (2005), Evan described a random C and C model. In random model, a bot will not actively contact other bots or the botmaster and would listen to incoming connections from its botmaster. This C and C model is potentially interesting to certain future types of botnets.

BOTNET CAPTURE AND ANALYSIS FRAMEWORK AND METHODOLOGY

We present capture and analysis's architecture and explain how its different components interact. The whole measure-ment setup is depicted in Fig. 2: With the help of nepenthes and honeynet, we collect samples of malware. These samples are then analysed with the help of Norman S and Box and CWS and Box which results in a behavior based analysis report.

As Fig. 3 illustrates, there are different components in Capture and Analysis framework. The first one is the malware capture and its goal is the capture of as many as possible malicious binaries. The second component is the malware analysis. In it the binaries captured on the first step are analysed in detail.

Our architecture was created to satisfy the following major requirements:

- Systematically and automatically collect and analyze malware traffic over the Internet without supervision
- scalable and robust infrastructure developing

Honeynet-based malware capture and analysis: Malware capturing and sharing is one of the most effective ways to discover new botnets to monitor and honeypots are a popular way to capture new malware samples such as bots.

In general, the more we capture malware, the more we can monitor botnets. In order to get more malware, we prepared two kinds of honeynet. One is a modified version of Nepenthes (Baecher *et al*., 2006), the other is honeynet and a download station is used.

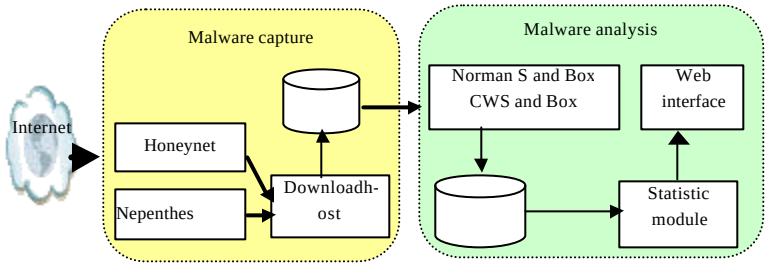

Fig. 3: Modules of the CAMB

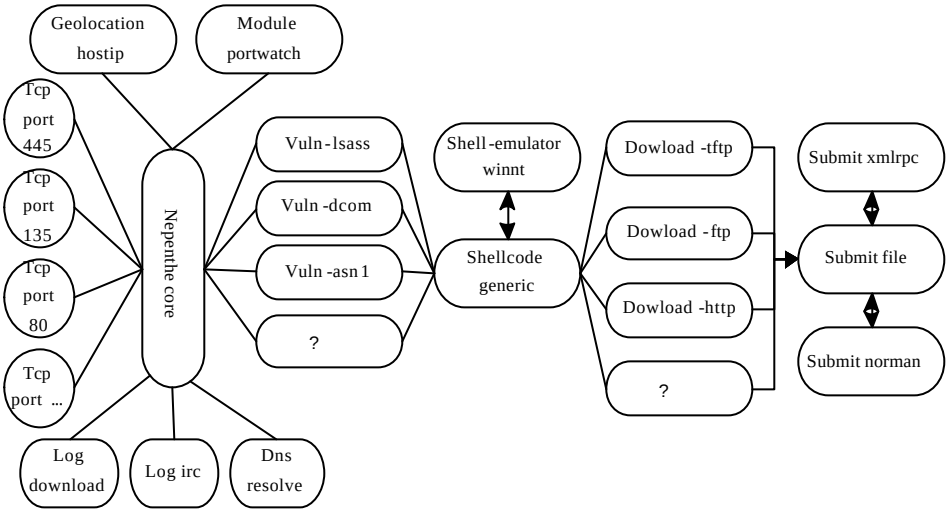

Fig. 4: Nepenthes architecture

Nepenthes platform is a popular and prominent malware capturing tool. It inherits the scalability of low-interaction honeypots yet in the while offers a high degree of expressiveness. Nepenthes runs on a UNIX server and provides enough emulation of common Windows services to incite most automated attacks. When an attack occurs, Nepenthes logs the malicious activities and attempts to download any binaries associated with the attack. In order to prevent excessive downloads from requesting the same URL multiple times, we disable the on-line download modules in nepenthes. What's more, the modification also helps avoid reflection attacks. For the retrieving of binaries a list of URLs to be downloaded is created and sent to a download host designed for this task. The schematic interaction between the different components is depicted in Fig. 4.

Nepenthes can get only malware whose vulnerability definition has been provided for users. Therefore, to complement the role of nepenthes, we make use of a honeynet consisting of two honepots. The primary reason for doing so is to ensure catching exploits missed by nepenthes. Our honeypots runs an unpatched version of Windows 2000 or Windows XP in a virtualized environment. This system is thus very vulnerable to attacks. It is located within the internal network of CAMBpus.

Malware analysis: The download host filters the entries in the received list and extracts the unique sources and URLs. Then the download host automatically download the binaries and stored in a MySQL database on the architecture, as well as the originating IP address.

IRC-based botnets often protect the server and channels with passwords to block unwanted visitors with the help the IRC protocol's capabilities. However, the malware sample itself will carry these credentials with it and analyzing the malware can discover these parameters. We can adopt automated mechanism to analyse and

Fig. 5: Part of a CWS and Box analysis of a malware sample

extract the sensitive information of a botnet from a given binary. With the help of honeypots and S and Box-like techniques, such as Norman S and Box or CWS and Box, we can automate this step to a certain degree. Malware that is distributed over the botnet is automatically downloaded and sent to a S and Box to trigger automated analysis of the malware and retrieve the results of the analysis. After this analysis, we are able to get IRC server information, domain name, port number and channel name, which helps us connect to IRC server of botnet. We can browse those collected data (such as binary sample statistic information and botnet parameters) on Web interface and configure connections to botnets.

The Norman S and Box simulates an entire computer and a connected network by reimplementing the core Windows system and executing the malware binary within the simulated environment. Norman S and Box daily receives thousands of suspicious files. All files submitted are being analyzed and as a subscriber you will get a list with information of Norman S and Box analysis over the past 24 h.

CWS and Box is an approach to automatically analyze malware which is based on behavior analysis: malware samples are executed for a finite time in a simulated environment, where all system calls are closely monitored. From these observations, CWS and Box is able to automatically generate a detailed report which greatly simplifies the task of a malware analyst.

The CWS and Box uses three strategies to fulll its three criteria:

- Dynamic analysis means that malware is analyzed by executing it within a simulated environment to achieve the automation of the analysis
- API hooking means that calls to the Windows API are re-routed to the monitoring software before the actual API code is called and grants the corrective ness of the overall process
- DLL code injection allows API hooking to be implemented while granting con_dence in the implementation and the correcteness of the reported results

Utilizing these three strategies the CWS and Box autmatically generates reports describing for example the following behaviors of the analyzed binary. File creation and modification, Windows Registry modification, DLL utilization, memory access, process creation and network connection.

Figure 5 represents part information of a CWS and Box analysis of a malware sample. This sample was uploaded to the CWS and Box Webinterface site (http://cwS and Box.org/).

PRELIMINARY RESULTS AND ANALYSIS

This section presents preliminary results of our CAMB framework. Based on the investigation of so called Virtual Honeynet, we deployed an experimental environment with some modifications as depicted in Fig. 6.

Our project installed a Nepenthes honeypot using version 0.20 running on Fedra Core 6. Over a period of 15 days, Nepenthes had collected 41 different samples as distinguished by the MD5 hashes of the binaries. The log files also indicate how and where each binary is obtained:

tail-1/var/log/nepenthes/logged_submissions
[20xx-xx-xxT17: 11: 53]xx.xx.xx.xx->xx.xx.xx.xx ftp: //blah: blah@xx.xx.xx.xx: 9044/ssdpsr.exe

Of these binaries, only portion were identified as malware by a particular antivirus product at the end of the 15 day period. We found that both Symantec AntiVirus and Rising Antivirus with the newest updates did not detect about 60% of the binaries from Table 1.

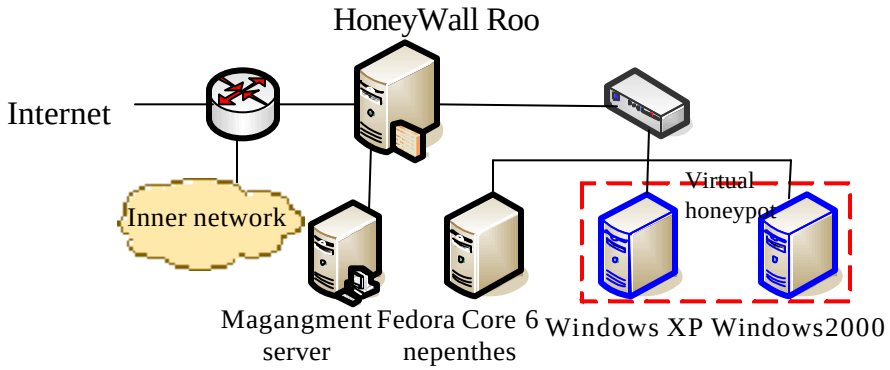

Fig. 6: Example of network topology

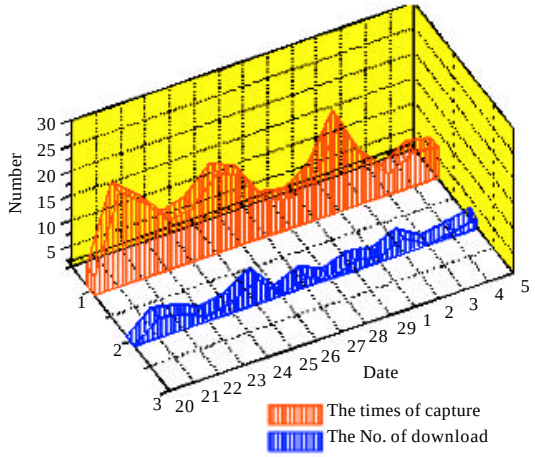

Fig. 7: Capture and download information of every day

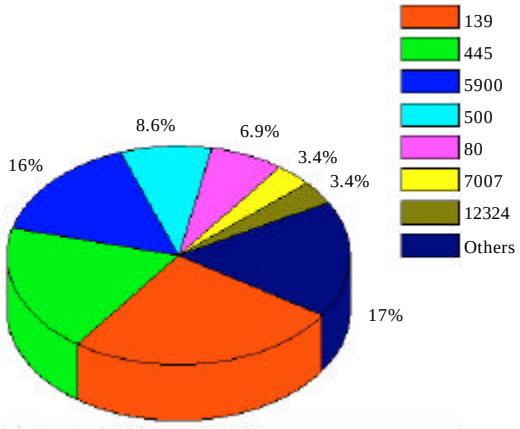

Fig. 8: Percentage of IRC server port types

With further analysis, we found that many binaries were IRC bots of one sort or another, like SDBot, Spybot and Mybot. The rest were worms such as Sasser and Korgo. The majority of binaries, whether classified, as worms or bots had some kind of IRC backdoor functionality.

Figure 7 shows the result of download and different binaries every day. It shows that there are many binaries of the same downloaded in Nepenthes.

Figure 8 depicts the ports use of the control server. IRC-based botnets usually use standard TCP port 6667 for IRC traffic. We can found that attacker more and more use some other ports such as 139 and 445 in the control servers. It is difficult to detect this kind of botnet by monitoring ports for defender.

Figure 9 shows Geographic location of the IRC server of the monitored botnets. The red area indicates the location of the IRC server.

Table 1: Detection information of two antivirus product

Antivirus engine	Total binaries	Detected binaries	Percent detected
Symantec antivirus	41	17	41.463
Rising antivirus	41	15	36.585

Figure 9 represents the national/regional distribution of the IRC server we tracked. The IRC server is mainly located in some countries such as America, Germany, China, Canada and other countries and regions.

From above long period monitoring and experiments result, we found that many different botmasters existed varying from novices to sophisticate. We also found that IRC is still the dominant protocol used for C and C communications and that its use is adapted to satisfy different botmasters' needs. Botmasters often hides their IP addresses. Sometime we can find the IP address because of careless of botmaster. We can locate it by IP address and make a further tracking. Many botnets have a short lifecycle.

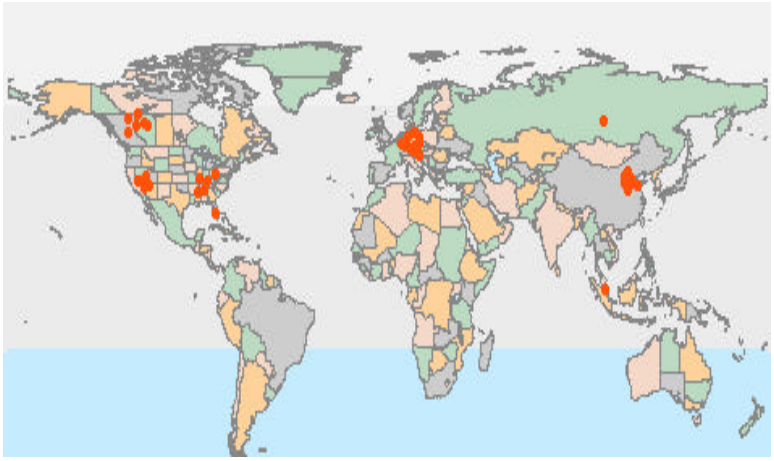

Fig. 9: Geographic location of the IRC server

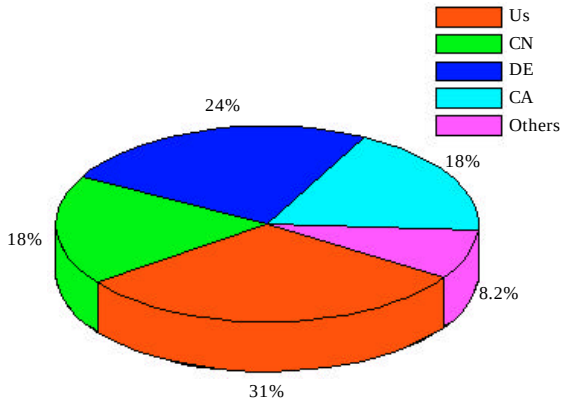

Fig. 10: National/Regional distribution of the IRC server

CONCLUSIONS

In this study, we presented a comprehensive framework for capture and analysis botnets in large-scale network. With the help of multidimensional, longitudinal and automated capture and analysis, we can efficiently detect botnets and extract the sensitive information that allows shutting down the botnet network. In addition, we discuss our preliminary results and lessons learned from this work.

Nevertheless, present study in botnets detection and analysis is still preliminary. Our approach has provided us with the analysis results necessary to move forward to our next steps which are proactive defence botnets. Several important research issues need to be further explored. In particular, in our future work (a) We would refine our capture and analysis framework; (b) We would investigate complete automated mechanism every step; (c) We verify and analyses the CAMB's ability to proactive defense against real-life attacks.

ACKNOWLEDGMENT

"The paper is supported by the National Science Foundation of China under Grant No. 61170262, the Hi-Tech Research and Development Program of China under Grant Nos. 2012AA012506,2012AA012901 and 2012AA012903, the Research Fund for the Doctoral Program (New Teachers), Ministry of Education of China under Grant No. 20121103120032, Humanity and Social Science Youth foundation of Ministry of Education of China under Grant No. 13YJCZH065, Opening Project of Key Lab of Information Network Security of Ministry of Public Security(The Third Research Institute of Ministry of Public Security) under Grant No. C13613, China Postdoctoral Science Foundation, General Program of

Science and Technology Development Project of Beijing Municipal Education Commission of China under Grant No. km201410005012, Research on education and teaching of Beijing University of Technology under Grant No. ER2013C24, Beijing Municipal Natural Science Foundation, Sponsored by Hunan Postdoctoral Scientific Program, and Open Research Fund of Beijing Key Laboratory of Trusted Computing."

REFERENCES

Baecher, P., M. Koetter, T. Holz, M. Dornseif and F. Freiling, 2006. The Nephenthes platform: An efficient approach to collect malware. Proceedings of the 9th International Symposium on Recent Advances in Intrusion Detection, September 20-22, 2006, Hamburg, Germany, pp: 165-184.

Bailey, M., E. Cooke, D. Watson, F. Jahanian and N. Provos, 2004. A hybrid honeypot architecture for scalable network monitoring. Technical Report CSE-TR-499-04, University of Michigan, Ann Arbor, MI., USA., October 27, 2004.

Cooke, E., F. Jahanian and D. McPherson, 2005. The zombie roundup: Understanding, detecting and disrupting botnets. Proceedings of the USENIX Workshop on Steps to Reducing Unwanted Traffic on the Internet, July 7-8, 2005, Cambridge, MA., USA.

Dagon, D., G. Gu, C. Zou, J. Grizzard, S. Dwivedi, W. Lee and R. Lipton, 2005. A taxonomy of botnets. Proceedings of 1st DNS-OARC Workshop, July 25-26, 2005, Santa Clara, CA., USA., pp: 1-16.

Freiling, F., T. Holz and G. Wicherski, 2005. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. Proceedings of the 10th European Symposium on Research in Computer Security, September 12-14, 2005, Milan, Italy, pp: 319-335.

Paxton, N.C., G.J. Ahn, R. Kelly, K. Pearson and B.T. Chu, 2007. Collecting and analyzing bots in a systematic honeynet-based testbed environment. Proceedings of the 11th Colloquium for Information Systems Security Education, June 4-7, 2007, Boston University, Boston, MA., USA., pp: 76-81.

Rajab, M.A., F. Monrose and A. Terzis, 2006. A multifaceted approach to understanding the botnet phenomenon. Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement, October 25-27, 2006, Rio de Janeiro, Brazil, pp: 41-52.

Tang, Y. and S. Chen, 2005. Defending against Internet worms: A signature-based approach. IEEE Comput. Commun. Soc., 2: 1384-1394.

Zhuge, J.W., X.H. Han, Y.L. Zhou, C.Y. Song, J.P. Guo and W. Zou, 2007. HoneyBow: An automated malware collection tool based on the high-interaction honeypot principle. J. Commun., 28: 8-13.