Testing Data Set for Analyzing Behaviors of Malicious Codes

Youngsoo Kim, Jungtae Kim and Ikkyun Kim
Cyber Security Research Laboratory
Electronics & Telecommunications Research Institute
Daejeon, Korea
e-mail: {blitzkrieg, jungtae_kim, ikkim21}@etri.re.kr

Abstract— Cyber targeted attack has sophisticated attack techniques using malwares to exploit vulnerabilities in systems and external Command & Control (C&C) servers are continuously monitoring and extracting data off a specific target. Since this attacking process is working continuously and uses diverse malicious codes and attacking routes, it is considered to be difficult to detect in advance. The paper proposes an indirect analytical method based on the Testing Data Set (TDS) that includes various malware behaviors for detecting cyber attacks. Especially, the proposed TDS contains both network and host dataset by running recently collected malwares in a secure testbed environment for collecting specific behaviors of the malware infections and activations. Such a combination of the self-generated datasets provides a valuable information source for the malware behavior analysis.

Keywords-Malicious Code; Behavior-based Analysis; Testing Data Set; Host-based Malicious Behavior; Network-based Behavior

I. INTRODUCTION

Recently, a common phenomenon of recently increasing cyber attacks tends to focus on a specific target with preparations for a long period. Such complicated and stealthy attacks, known as the Advanced Persistent Threat (APT), normally begin with a customized malware penetration for a target system infection in order to collect internal information of the target system. The malwares in the infected system continuously communicate with the Command & Control (C&C) servers to transmit the internal system information to hackers that was collected in advance. Then, the hacker can control the target system as well as other surround servers and resources by transferring additional malwares. Finally, the hacker achieves an intended purpose, such as the destruction of a target system, or to take the financial gains, with acquired controls over the target system and terminates the attack without leaving any evidences for the traces.

Many researches and studies are on-going to find and prevent such sophisticated APT attacks in advance that are found to be the main cause of the recent cyber incidents. Since most of the APT attacks begin with a malware infections, therefore studies for the detection of anomalies in the system caused by APT is also being actively conducted. Generally, those analysis methods used to predict the anomalies are being investigated through either the static or dynamic analysis of malwares. Also, there is an indirect analysis method, which is based on the data generated in a secure environment, for the malware infections and activations.

The paper proposes an indirect analytical method based on the Testing Data Set (TDS) that includes various malware behaviors. For the conventional Network TDS, it is not easy to get a useful dataset due to the different types of network attacks happening in the heterogeneous environments. Also, the Internet Server Providers (ISP) are not able to publically open the network TDS, that are collected for traffic analysis for the management purposes, due to the subscriber privacy policy. In case when the dataset is available, the network payload that contains personal and classified information such as IP address is removed or modified, which becomes an improper dataset for security analysis. Table I presents various types of the network TDS that are used for the attack analysis [1]-[5]. In case of the Host TDS, which obviously contains behavioral information for PC users including application usages and processes details, it was not easy to find a proper dataset to analyze. Consequently, the proposed testbed generates self-generated TDS which contains both network and host dataset by running collected malwares. The self-generated combination of both Host and Network TDS provides a valuable information source for the malware behavior analysis before the collected malware patches can be available to prevent collection of active behaviors of the malware.

TABLE I. NETWORK TDS

NAME	KDD CUP 1999	MIT LINCOLN LAB	NLANR	CAIDA	SONY MAWI
ATTACKS	DOS, BACKDOOR, BUFFER OVERFLOW	DOS, DDOS	SLAMMER, CODE-RED	WITTY	SLAMMER , WITTY
PACKET SIZE	HEADER LENGTH	NO LIMIT	HEADER LENGTH	N/A	96 BYTES
USAGE	PERFORMANCE EVALUATION FOR IDS		TRAFFIC COLLECTION, ANALYSIS		
COLLECTING TIME	1999	1998~ 2000	2001	2001	1999~ PRESENT
AMOUNTS (MB)	17	150~ 200	25~40	1	100~150
FORMAT	PCAP	PCAP	TSH	TEXT	PCAP

 ISBN: 978-1-61208-444-2

Especially in the past, it was common to use a Virtual Machine (VM) to evaluate the runtime malware behaviors. As malwares were, discovered recently, programmed to circumvent the VM environment, therefore we apply a system reboot software to roll-back the system into an initial clean state when every time malwares are executed.

The rest of the paper consists of three sections. It begins with the Section II, which contains a proposed TDS generation environment in details including a network configuration with testbed components, detailed contents of the collected host and network TDS. And a method to collect the recent malwares and setup Host PCs to maximize the malware activation rate as well as a method of collecting and storing data generated by the actual executions of malicious codes will be described in the Section III. Finally, Section IV concludes a paper after reviewing and evaluating the resulting TDS.

II. TDS GENERATION ENVIRONMENT

We generated normal and abnormal data separately to get the TDS. The normal data could be a collection of host data generated from computers, e.g., name of process or triggering time of a specific event, while users doing their normal job with computers, without infection of malwares. It also includes the network traffic data generated from devices linked with network, e.g., source IP address, destination port, or starting time of a specific session, simultaneously with the above host data.

An abnormal data could be generated after activating malicious codes. We have gathered diverse malicious codes in advance and executed them automatically to get an abnormal data sets which includes both host data and network traffic data.

A. Network Architectures and Components

The TDS generation environment proposed has a different approach to get the both normal TDS and abnormal TDS. The following Figure 1 depicts network architecture of our testbed for collecting the normal and abnormal TDS. This testbed is designed as similar as to a general network architecture in real world. It includes general components, e.g., user computers, a web server, switches, routers, and a database server. It also contains a notebook for an attacker, C&C servers, a PMS (Patch Management System) server, and redirection server for support advanced attacks. To make this testbed to be the same as real enterprise network architecture, we include commercial security devices, e.g., IPS (Intrusion Prevention System), WAF (Web Application Firewall), and UTM (Unified Threat Management).

To collect TDS by generating two types of data, we need the following components including User Computers (UC), Host Data Collector (HDC), Network Data Collector (NDC), Network Traffic Collecting Device (NTCD), Servers for Network Drive (SND), and Malware Crawler (MC). We installed a specialized software on the Host Data Collector (HDC) at UCs and activated it to get the TDS of user PCs.

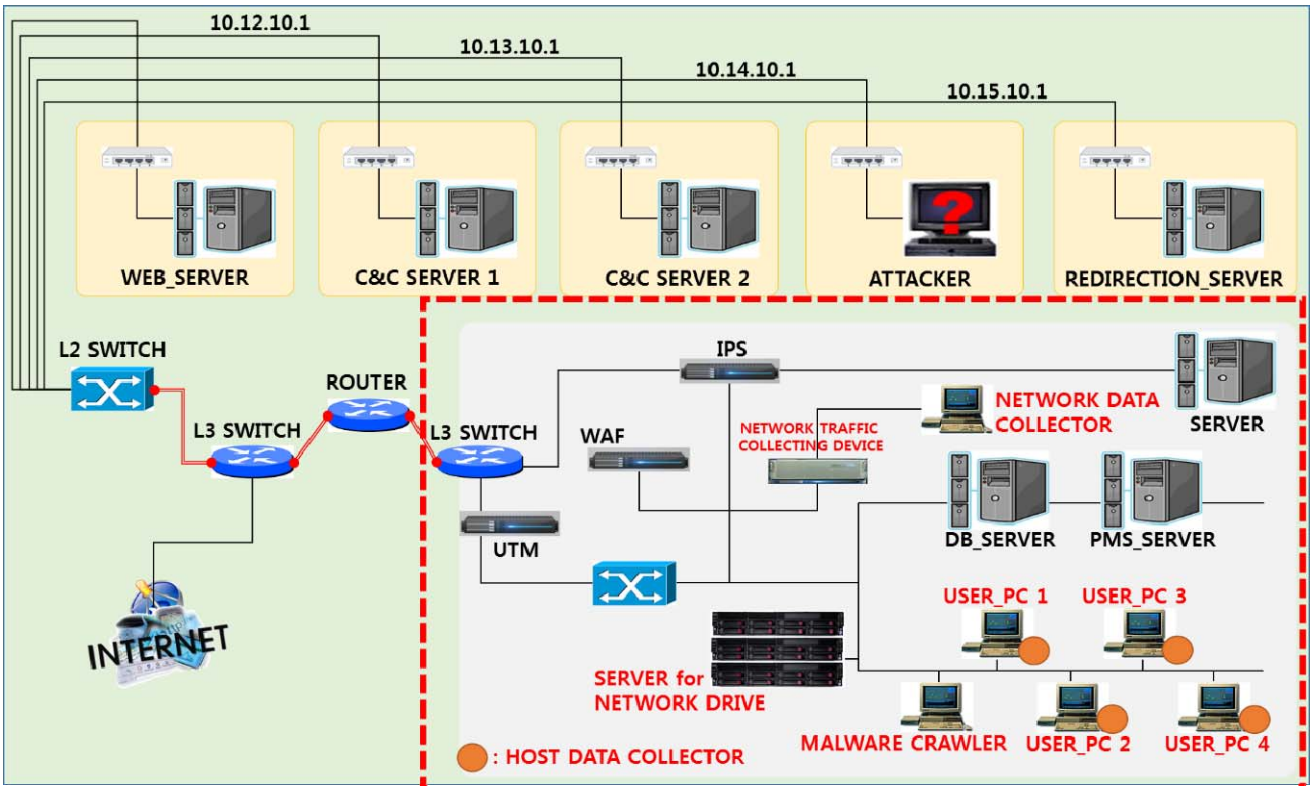

Figure 1. A Testbed Configuration of the TDS Generation Environment.

The UCs must be connected with the SND for transporting the collected host TDS. The HDC is an agent program for logging host data including the triggering time of a specific event, event user, process name, process identity, parent process identity, threat identity, event number, event name, API name, path of a specific process, parameters for calling APIs, etc. The logged data will be saved into a form of binary file (.DAT) once in a minute.

For the Host TDS, a set of binary files will be stored in a local directory temporally and moved to a specific folder of the network drive. We can select APIs for logging and filter some processes not to be saved selectively. The NDC collects and stores network data from the NTCD that tapped into a local switch. The network data includes Connection Start time, Connection End tine, Source IP Address, Destination IP Address, Source PORT, Destination PORT, Protocol, Inbound Flow Bytes, Outbound Flow Bytes, Numbers of Inbound Packets, Numbers of Outbound Packets, Service Name and Service Provider information and etc.

The NTCD includes DPI (Deep Packet Inspection) engines and flow collector which is based on multicore-processors. It can control asymmetric traffics using clustering technologies, support to set up controlling policies and to subscribe server-based session statistics [6]. The SND is a kind of storage server for the hosts TDS, which is stored in local directory of user's computer temporally. We can set up this network drive at the Windows Explorer of UCs. The MC is automated programs that create copies of malwares from some malware sample sites, e.g., *Vxvault* or *Malshare*. Resulting logs for visiting web sites are stored at *Mongodb* and malware information can be stored at *MySQL* [7][8].

B. Contents of TDS

The Testing Data Set (TDS) contains many useful data for the security analysis. We selected some items primarily for analyzing status of the host computer system and networks, e.g., malware behaviors or abnormal network flows. But, it can be applied to many fields for security analysis, since it includes the real case normal and abnormal host and network data which can be correlated in the both time and IP addresses. For example, an identified malware process with a PID, local IP and port information can help to find overall network connectivity of the malware to the external networks.

Following Table II and Table III show a set of collected items of each host and network data respectively [9]. Firstly, the collected items of the host data includes an Index Number of Event, Triggering Time of Event, Event User, Process Name, Process Identity, Parent Process Identity, Threat Identity, Event Number, Event Name, Windows API Name, Process Path, and Parameter of Calling APIs.

Also, the collected items of network data includes a Connection Start time, Connection End time, Source IP Address, Destination IP Address, Source PORT, Destination PORT, Protocol, Inbound Flow Bytes, Outbound Flow Bytes, Numbers of Inbound Packets, Numbers of Outbound Packets, Service Name and Service Provider information.

TABLE II. COLLECTED ITEMS OF HOST DATA

Collected items	Description
Index	Sequence number of a specific event
Time	Triggering time of a specific event
User ID	Event user
Process Name	Process name
PID	Process identity
PPID	Parent process identity
TID	Thread identity
Event Number	Event Number
Event Name	Event Name
API Name	Name of Windows API
Path	Path of a specific process
Parameter	Parameters for calling APIs

TABLE III. COLLECTED ITEMS OF NETWORK DATA

Collected items	Description
startTime	Starting time of a specific session
endTime	Terminating time of a specific session
Duration	Duration of a specific session
srcIp	Source IP address
destIp	Destination IP address
srcPort	Source port number
destPort	Destination port number
tcpFlag	TCP flag
protocol	Protocol name
inpkts	The number of input packets
outpkts	The number of output packets
inbytes	The amount of bytes for input packets
outbytes	The amount of bytes for output packets
service	Service name
device	Device name,
Sp	Name of service provider,
Status	Normal (0) / Abnormal behavior (malware name)

 ISBN: 978-1-61208-444-2

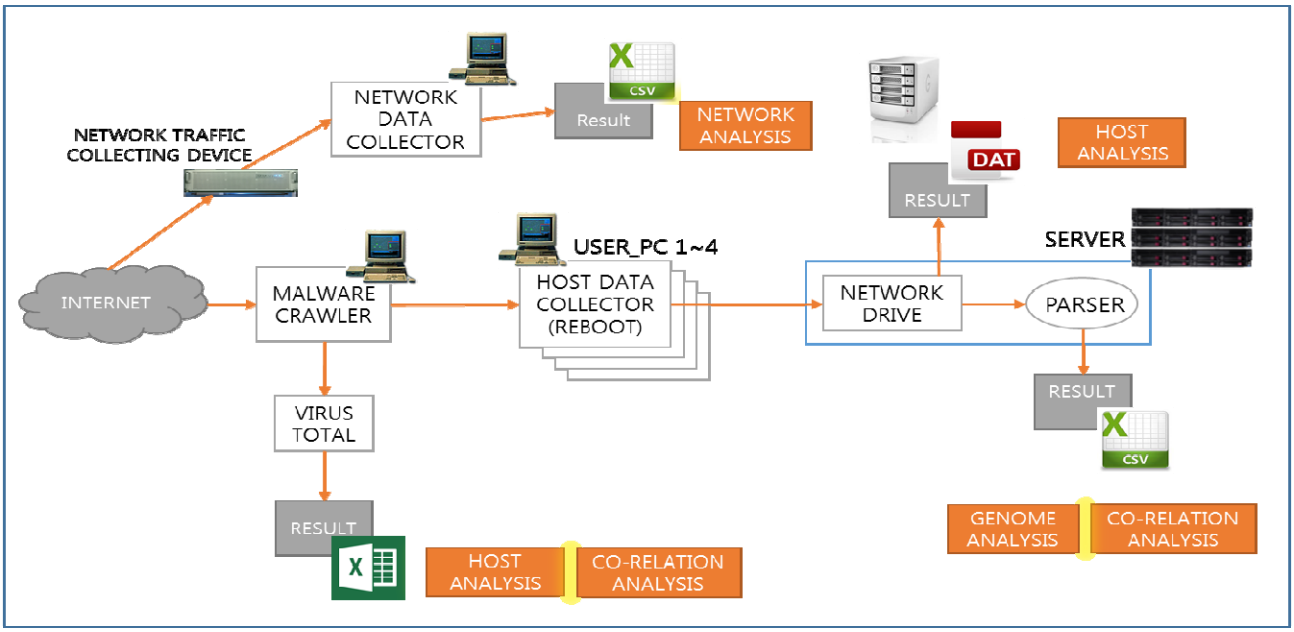

Figure 2. A Conceptual Structure for Collecting of Abnormal Data.

III. METHOD OF COLLECTING TDS

A. *Collection of Recent Malwares*

The malwares, recently founded, should be collected and managed to generate an abnormal data continuously. The MC creates copies of malwares from the malware sample sites including the *Vxvault* and *Malshare*, and stores them. It also manages malware related information using a database. The resulting logs for visiting the web sites are stored at the *Mongodb* and malware's information can be stored at the *MySQL*.

TABLE IV. TABLE STRUCTURE FOR STORING MALWARES' INFORMATION

Field	Type	Description
no	int(11)	Sequence number
type	text	Name of malware sample site
reg_time	text	Stored time
malware_name	text	Name of malware
download_link	text	Download URL
status	text	Status (Wait/Fetched/Completed/stop)
analysis_ip	text	IP address of performing analysis
start_time	text	Starting time of analysis
end_time	text	End time of analysis

The above Table IV shows a detailed information of the collected malware information with the DB table structure. After the MC copies the collected malwares from the malware sample sites, it calculates the hash values of those collected malwares and removes duplicated malwares by comparing each hash values. We use the SHA256 hash algorithm for hashing [10].

When a malware is registered for the first time, the initial status information is "Wait", and the client program in the HDC copy the target malware to activate, resulting status changed to the "Fetched" state. The status of malwares can be changed as "Completed" in case of finishing analysis or activation, and as "Stop" in case of being interrupted because of some errors.

B. *Collection of Abnormal Data*

Figure 2 depicts a conceptual structure for collecting the abnormal data. The HDC programs installed at 4 UCs are activated to get the TDS for host data. It copies malwares those status are in "Wait" state from the MC and activates them one by one. The logged data will be made into a form of binary file (.DAT) in every minute. The Host TDS, a set of binary files, will be stored in local directory temporally, and moved to a specific folder of the network drive.

The binary files are generally used for a host data analysis, but should be transformed to the CSV file format using a parser for the cyber-genome analysis and co-relation analysis. We set up 4 UCs with different environmental settings, as shown in the Table V. We selected applications for installation by referencing the CVE (Common Vulnerabilities and Exposures) lists [11].

The NTCD is positioned at a connection point between an external and internal network in order to detect malicious

 ISBN: 978-1-61208-444-2

behaviors which occur from the outside, and collects the network traffic data generated between the external and internal network.

The 20 data factors which are collected by the network data collector includes Connection Start time, Connection End time, Source IP Address, Destination IP Address, Source PORT, Destination PORT, Protocol, Inbound Flow Bytes, Outbound Flow Bytes, Numbers of Inbound Packets, Numbers of Outbound Packets, Service Name and Service Provider information.

TABLE V. DIFFERENT SETUPS OF 4 UCS

	User PC_1	User PC_2	User PC_3	User PC_4
OS	windows 7 SP1 32bit		windows 7 SP1 64bit	
Installed Applications	IE 10	IE 11	IE 10	IE 11
	Flash Player 14	Flash Player 15	Flash Player 16	Flash Player 17
	Acrobat Reader 10	Acrobat Reader 11	Acrobat Reader 10	Acrobat Reader 11
	.NET			
	Google Chrome 43.X			
	SDK			
	Silver Lite			
	MSOffice 2003 SP2 / MSOffice 2007 SP3 / MSOffice 2013 SP1			
	HWP 2010/2014, 7zip, Nateon, Alftp, Mplayer, Notepad, Putty, MS Outlook, Outlook Express, cmd, telnet, utorrent, gzip, vim, Wordpad, Kakaotalk, Facebook, windows media player, GOMaudio, Google Drive, gimp, Filezilla, Smplayer, Xmplay, pnotes, Naver Streaming Service, Stickies, Cpu-z, Freecommander, Apache, Unitaller			

After it collects the network flow data from the edge routers and it sends the data which should be included in payload defined UDP packet to NDC for a network topology processing. NDC makes and stores network TDS locally as a format of CSV file. It can be used for network analysis. By inputting a hash value of a specific malware to VirusTotal site, MC gets some diagnosis results that over 40 vaccine companies have [12]. It can be referred to host analysis or co-relation analysis.

For the evaluation purpose, we have collected and generated a TDS by executing 3,392 malwares during two weeks periods. Each host collector is configured to collect the process information for 5 minutes until malwares were properly executed, then the notification of malware process terminations were forwarded to the NDC.

For recursive testing of heterogeneous malwares, the Comback 7.0 was used to support automatic reboot of the system to its original state after collecting the malware process information [13]. For the network TDS, the collected binary data is converted to the CSV file of 434 MB for analysis, and 6.7 TB were collected for the binary type host TDS.

The host TDS were also converted to a CSV file type with a specific parser developed for easy data analysis to be used in the Cyber Gene and Correlation analysis. Additionally, the hash values of malwares names from the VirusTotal were generated as an Excel format for analysis.

Following Figure 3 and 4 represents a sample host and network TDS collected respectively.

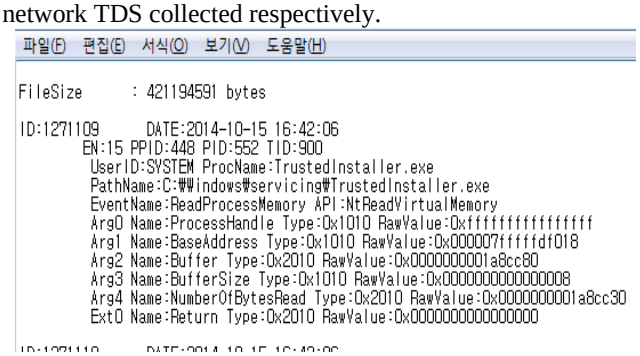

Figure 3. Host TDS

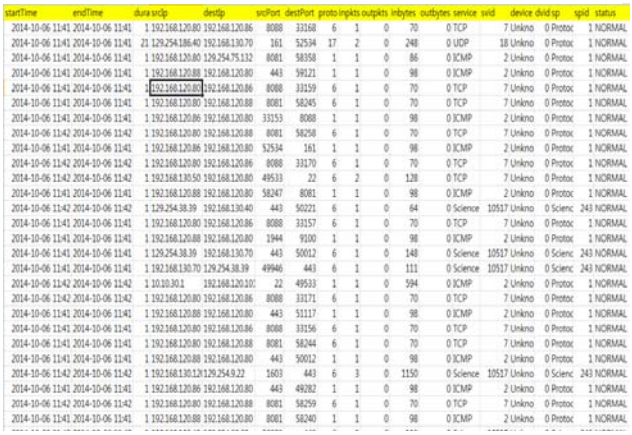

Figure 4. Network TDS

IV. CONCLUSION AND FUTURE WORK

To cope with the increasing advanced cyber attacks and to overcome the limitations of the conventional Network TDS, the paper proposed an indirect analytical method for detecting advanced cyber attacks (APT) by proposing a collection method of the Testing Data Set (TDS) that includes various malware behaviors. To do so, a testbed was designed to suit with the real network environments and various types of recent malware were collected for evaluating the resulting dataset collected.

The self-generated dataset collects predefined 12 and 17 detailed components information of the Host and Network TDS respectively. The combination of both Host TDS and Network TDS provides a valuable information source for the malware behavior analysis. For the future works, a TDS collection for the known malware with behavior information will provides a useful insight for malware analysis which helps to create categorized datasets based on the different types of the malware behaviors.

 ISBN: 978-1-61208-444-2

ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No.B0101-15-1293,Cyber-targeted attack recognition and traceback technology based on the long-term historic analysis of multi-source data).

REFERENCES

[1] The UCI KDD Archive, "KDD Cup 1999 Data," available at http://www.ics.uci.edu/~kdd/databases/kddcup99/kddcup99.html [retrieved: Oct, 2015]

[2] Lincoln Laboratory Massachusetts Institute of Technology, "MIT Lincoln Laboratory – DARPA Intrusion Detection Evaluation Data Sets," available at http://www.ll.mit.edu/IST/ideval/data/data_index.html [retrieved: Oct, 2015]

[3] NLANR Measurement and Network Analysis Group, "NLANR PMA," available at http://pma.nlanr.net [retrieved: Oct, 2015]

[4] Cooperative Association for Internet Data Analysis, "Cooperative Association for Internet Data Analysis (CAIDA)," available at http://www.caida.org [retrieved: Oct, 2015]

[5] MAWI Working Group, "MAWI Working Group Traffic Archive," available at http://tracer.csl.sony.co.jp/mawi/ [retrieved: Oct, 2015]

[6] PacketLiner EL480, http://sysmate.com [retrieved: Oct, 2015]

[7] Vxvault, http://vxvault.net/ViriList.php [retrieved: Oct, 2015]

[8] Malshare, http://malshare.com/ [retrieved: Oct, 2015]

[9] D. Moon, H. Lee, and I. Kim, “Host based Feature Description Method for Detecting APT Attack,” Journal of The Korea Institute Of Information Security & Cryptology, Vol. 24, No. 5, Oct. 2014, pp. 839-850, ISSN: 1598-3986.

[10] S. Lee, D. Choi, and Y. Choi, “Improved Shamir’s CRT-RSA Algorithm: Revisit with the Modulus Chaining Method,” ETRI Journal, Vol. 36, No. 3, Jun. 2014, pp.469-478, ISSN: 1225-6463.

[11] Common Vulnerabilities and Exposures, http://www.cvedetails.com/ [retrieved: Oct, 2015]

[12] Virus Total, https://www.virustotal.com/ [retrieved: Oct, 2015]

[13] Comback, http://www.wowcomback.com/comback/combackIntro.asp [retrieved: Oct, 2015]

 ISBN: 978-1-61208-444-2