Krzysztof CABAJ, Piotr GAWKOWSKI, Konrad GROCHOWSKI, Dawid OSOJCA

Warsaw University of Technology

doi:10.15199/48.2015.11.48

Network activity analysis of CryptoWall ransomware

Abstract. *The paper presents the analysis of the CryptoWall ransomware network behaviour. In this approach a HoneyPot technology as well as the automatic run-time malware analytical system called Maltester were used. We present the practical results of the analyses, technologies and tools used, and the gained experience with dynamic analysis of ransomware software in a dedicated environment. Most of the data was collected with the use of the HoneyPot infrastructure created and deployed in the network of the Institute of Computer Science WUT.*

Streszczenie. *Praca przedstawia analizę zachowania sieciowego złośliwego oprogramowania CryptoWall typu ransomware. W badaniach wykorzystano technologię HoneyPot oraz system automatycznej analizy działania złośliwego oprogramowania w czasie jego wykonywania o nazwie Maltester. Zaprezentowano zarówno uzyskane wyniki analiz, wykorzystane technologie i narzędzia jak i doświadczenia zebrane podczas analizy oprogramowania typu ransomware w autorskim środowisku analitycznym. Większość danych zebrana została z wykorzystaniem infrastruktury HoneyPot stworzonej i wdrożonej w sieci Instytutu Informatyki Politechniki Warszawskiej. (**Dynamiczna analiza aktywności sieciowej oprogramowania CryptoWall typu ransomware**).*

Keywords: Network Security, HoneyPot Systems, Network Attacks, Dynamic Malware Analysis.
Słowa kluczowe: bezpieczeństwo sieci, systemy HoneyPot, ataki sieciowe, dynamiczna analiza złośliwego oprogramowania.

Introduction

HoneyPot systems are known and quite widely used by security researchers for almost 20 years. They allow people from security community to spy attackers, identify new threats, ways of spreading etc. In order to conduct security research, there are various types of HoneyPot systems deployed for several years in the network of Institute of Computer Science in Warsaw University of Technology. Some of the experience gained are already published in [1, 2]. Preliminary works were mainly focused on automatic analysis of recorded meta-data of observed connections to our HoneyPot systems (e.g. detection of similar connections using frequent sets). For this purpose various data mining techniques were successfully utilized [3, 4]. The data about connection attempts to HoneyPots deliver many interesting information about the ways an exploit tries to infiltrate the victims computers. As the exploit is just a begin of the infection it is even more interesting to know what happens after this step and how the malicious software (like viruses or trojans) behaves [2, 5, 6]. To investigate it, HoneyPots can deliver not only data concerning connections but even more interesting pieces: samples of malware passed to the victims machines [7]

In the security field an assumption exists that there is no 100% secure software. Imperfections that can be used to provoke malicious actions on the victims machine are called vulnerabilities. They can be exploited through malicious documents sent by e-mail, accessing the infected web page or direct attack on the services served over the network (e.g. remote code injection through malicious network request). In this last scenario the infection does not require any user actions on the target machine to activate malicious code. Thus, it is more dangerous as the infection could be undiscovered for a long time with-out the proper system monitoring. Here arises the problem how to recognize any anomalies from normal operation [2, 5, 6]. As the network connectivity is widespread, one of the most important point of observation is machine's network activity.

This paper contains the analysis of the behaviour of one of the most recent ransomware called CryptoWall is given. It is based on black-box dynamic observations of network activity of the CryptoWall samples. The next section discusses static and dynamic malware analysis. Then, our custom environment for dynamic malware analysis, called Maltester, is presented. It is followed by presentation of practical results and gained experience and summarized at the end.

Static and dynamic program analysis

Generally speaking, analysis of the malware can be made in two ways: by analysing the malware's code statically (without malware execution) or dynamically – by its execution and observation.

Undisputedly, static analysis is much more safer for the researcher, nevertheless, the one has to be careful to not execute the sample by, for instance, accidental mouse clicking. Such analysis bases mostly on reverse engineering tools (like disassembler) to discover some basic blocks in the code, identify malware-related ones, recognize some specific and common schemas of the control flow or sequences of operations. That can be used to identify malicious code even if it is mutated version of the already known virus [8, 9]. However, malicious software uses some obfuscation techniques that severely aggravate this kind of analysis [8-11]. Moreover, malware functionality is not self-contained anymore – for example it can connect to some controlling nodes over the Internet, so sometimes only dynamic analysis (during the run-time) can reveal the true behaviour of the malware [11].

Properly secured sandbox is required to conduct dynamic analysis safely [5, 10, 11]. Here, the virtualization technology comes with great help. Typically, the malware sample is uploaded into specially crafted virtual machine. Execution of the sample is traced using monitoring software (e.g. *tcpdump*, system monitors) to collect the communication and detailed actions taken in the virtual machine (like system calls, disks operations, adding auto-run applications etc.). The scope of the collected data can vary on the types of tools installed on the testing environment. It is worth to note, that similar approach is commonly used in recent antivirus systems – the suspicious applications are at first executed within the sandbox and their actions are monitored for several seconds of execution. However, some malware simply delay their real activity just to deceive this detection mechanism. There're plenty of techniques used in the malware that prevent their monitoring and reverse engineering (e.g. detection of debuggers) [5, 10]. All this makes dynamic analysis definitely a challenge.

Dynamic analysis may provide several types of information, so, it is worth to distinguish two kinds of analyses – each require different techniques and have different goals: actions made by the malware within the infected machine and tracing its activity over the Internet. In the first case we can identify the scope of damage on the

infected host. In the second case, basing on the captured network activity, we can identify other hosts (over the Internet) involved in the malware infrastructure. They might be just new targets for the malware (attacked) or some hosts already taken over by the malware that serve some resources for it.

In this paper we present the Maltester environment (see next section) for dynamic analysis. However, we mainly focus on our experience with identification of the network activities of the CryptoWall samples.

Maltester environment

Maltester uses open source Xen hypervisor running at Linux Debian operating system. It consists of four virtual machines establishing an infrastructure depicted in Fig. 1 running in one physical server.

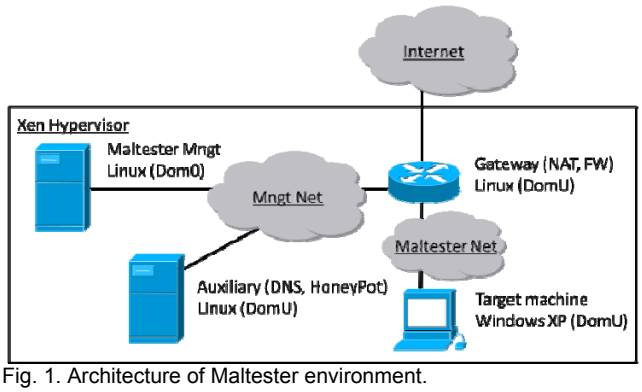

Fig. 1. Architecture of Maltester environment.

The management host is responsible for controlling the whole process of the analysis. When a user adds a new sample to the system (with remote call interface), an instance of the target machine is created from the snapshot state. This machine (currently running Windows XP) has a software that can interact with the management host. Thus, the malware sample is automatically transferred to this target system and executed. At this instant the dynamic analysis starts.

For the security reasons the target system (infected during analysis) is not directly connected to the Internet. Two dedicated virtual bridges are used. One for the management traffic and the second one for the traffic sourced from the target machine that may contain potentially hostile packets. Additionally, all the traffic is forwarded by specially configured gateway, running Debian Linux. This machine performs NAT translation as well as filtering of any known malicious traffic. It can also forward the traffic to other specialized services executed on auxiliary host (e.g. DNS proxy, dedicated HoneyPot). Moreover, gateway machine is responsible for recording of all the traffic generated by during the execution of malicious program (using *tcpdump*).

The analysis stops after a specified period of time: the management host freezes the state of the target and initiate the supplementary analysis (e.g. using *volatile* utility) and checking of the target's filesystem.

When the given sample is analysed for the first time it usually interacts with real servers provided by the attacker, for example with Command and Control (C&C) servers. However, multiple, short connections to the C&C servers, issued by frequent executions of same sample, can be identified by the attacker. Due to this fact in some experiments, as soon as we understand a part of the protocol used between the malware and C&C, we can easily develop an emulator, which can interact with running malware. These emulators are running at auxiliary machine (traffic to them is forwarded via gateway). Such utilization of both machines is further described with details.

CryptoWall

According to quarterly report for 2015 Q1 of the McAfee antivirus company there is 165% rise in the number of observed/detected ransomware samples [12, 13] (e.g. CryptoWall, CTB-Locker, TorrentLocker, BandarChor and Teslacrypt). The ransomware is a kind of malware that encrypts important personal files, like documents, spreadsheets etc. Decryption key (or a tool) is provided to the victim after paying ransom to attacker.

The most comprehensive and detailed analysis performed in Maltester environment at this moment concerns CryptoWall ransomware. During this research six CryptoSall samples were investigated – one was obtained from an infected machine in the faculty and five provided by the security community (*http://www.malware-traffic-analysis.net/*). From the user's perspective there are two known ways in which CryptoWall can infect machine: email spam with malware in attachment and infected web site with Angler Exploit Kit [13]. After successful exploitation CryptoWall malware contacts various servers over the Internet. As soon as the RSA 2048 bit public key for encryption is received, all important data files on victim's machine are being encrypted. In each directory encrypted by CryptoWall two additional files are created: HELP_DECRYPT.PNG and HELP_DECRYPT.URL. The first one contains a description of what happened with victim's files and where further information can be found. The second file contains a direct URL to the web page (shown in Fig. 2) where the victim is provided with his own personal code for further actions.

Fig. 2. Personal web-page for the CryptoWall victim.

To hinder detection and removal of this web-service it works as hidden Tor service [14]. That makes identification of physical machine hosting this service almost impossible. Moreover, ransom payment is made using Bitcoins (see Fig. 2). These two methods practically assure full anonymity for the attacker, and make difficult to track and stop his activity.

Analysing preliminary network traffic obtained by Maltester it was obvious that the whole communication is somehow encrypted. However, CryptoWall uses domain names instead of direct IP addresses. That means it needs a DNS service to resolve his peers. In order to get more control over the communication process the Maltester's firewall was specially configured and fake DNS service provided (executed on the Auxiliary Maltester host). It records, blocks or redirects (if needed) the DNS requests. Analysis of the recorded traffic revealed that the first action performed by the CryptoWall is victim's public IP address check using one of the publicly available services (all the

samples have three hardcoded address of such service: *ip-addr.es*, *myexternalip.com*, *curlmyip.com*).

In the next step CryptoWall tries to contact one of the hardcoded servers. In order to track all of them, the configuration of our fake DNS blocks these requests. Due to lack of response, after the timeout, the next domain was checked. Conducted experiments show that after 15 minutes all the domains were checked, and CryptoWall started to query the same addresses once again. In effect Maltester environment had collected a complete list of servers the malware tries to contact. It wouldn't be possible without fake DNS as the malware is satisfied with the first successful address resolving.

Having these addresses the Maltester was equipped with additional WebHP HoneyPot instance [3] (on Auxiliary host) and forwarded to it all the traffic directed to the previously obtained addresses. This Maltester configuration provided the ability to conduct further protocol analyses while blocking the malware communication over the Internet. Together with fake DNS ability to force malware to use all the servers, HoneyPot was able to automatically gather all the URLs used by CryptoWall. Sample of acquired URLs:

- /**wp-content/**themes/Avada-Theme/img3.php?q=**06cm24g496**
- /**wp-content**/themes/happykids/img1.php?o=**06cm24g496**
- /**wp-content**/themes/satellite_corp/img5.php?z=**06cm24g496**
- /blog/**wp-content**/themes/Chucky/img5.php?o=**06cm24g496**
- /**wp-content**/themes/twentyeleven/img5.php?w=**06cm24g496**

As can be easily seen, most of the machines served WordPress CMS (*wp-content* in the paths). This system has a long list of vulnerabilities. Additionally, users can add some plugins which may introduce new vulnerabilities. This suggested, that those machines might be infected as well and used in CryptoWall procedures against their administrator wishes. Manual analysis of the used URLs and their addresses revealed that one of them is a Polish web site, which allowed direct contact with its administrator. After that it became clear (with high confidence) that the machines the malware is connecting to joined CryptoWall infrastructure as victims.

The communication between the infected machine and the compromised servers uses the HTTP protocol but it is encrypted, so, we cannot say if the compromised servers are the core of the CryptoWall infrastructure or just a proxy service. Thankfully, the web administrator had provided the hostile script for further analyses. Its analysis shows that the machine only forwards the traffic to other machines. Due to this fact, later in this article we called these servers *proxy servers*.

The analysis of the PHP script proved that CryptoWall uses RC4 encryption algorithm, with random key for each connection session (PUSH parameter of the URLs shown in bold). With that knowledge, simple tool for decryption of malware communication was easily implemented. Decryption revealed simple text protocol (see Fig. 3). In the first connection, malware reports its unique identifier and currently used IP address. The communication is acknowledged. The second connection uses very similar request, however response is longer, and contains a Tor address of the (previously mentioned) ransom web-page, victim's personal code and a public key used for further encryption of the data. Further investigation shows that in this version of the CryptoWall ransomware public and private key are generated outside of the infected machines as well as outside of the proxy. In the third connection to the proxy, described previously PNG image containing information about the decryption process is provided. After successful downloading of this image the last connection from the infected victim acknowledges reception of all the data and communication stop.

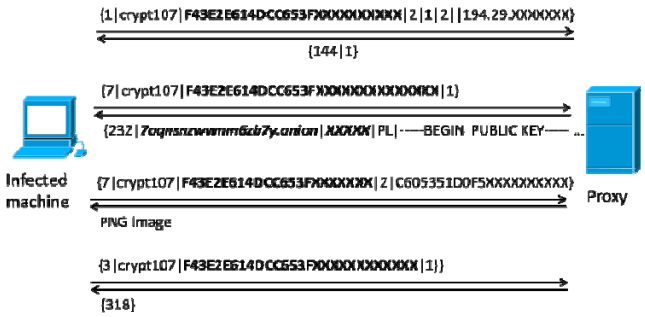

Fig 3. CryptoWall communication draft.

At this point of the analysis the overall discovered CryptoWall infrastructure can be drawn as depicted in Fig. 4. Two elements, introduced during our research, namely HoneyClient and HoneyProxy, are related to the Target machine in the Maltester (see Fig. 1) and the CryptoWall proxy machine executing our own HoneyPot-like fake PHP script. Thanks to the courtesy of the identified proxy machine administrator we were able to change the original CryptoWall's script to our own. It logs interaction with any infected machines contacting to this proxy and blocks sending the public key for encryption.

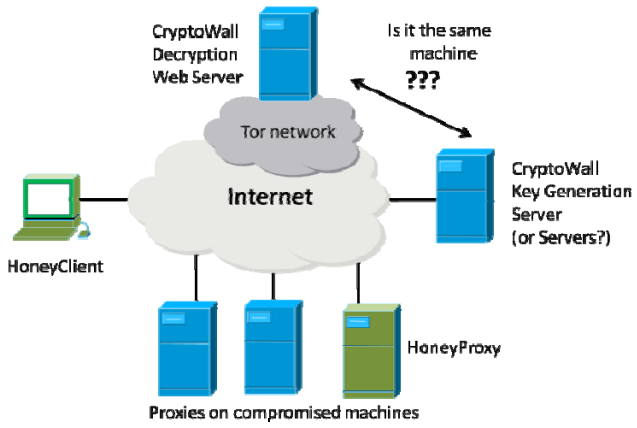

Fig. 4. CryptoWall infrastructure overview.

For reliability purposes each version (mutation) of the CryptoWall uses multiple proxy servers. Table 1 presents details concerning used proxies by the analysed samples. What is interesting, despite various proxies used and various samples of the malware, the same public key used for victim data encryption was always returned.

Table 1. Active proxies for various CryptoWall samples.

	Size (bytes)	Used proxies	Active proxies	Activity check date
Sample00	262656	29	11	2015.05.01
Sample01	221185	37	7	2015.05.18
Sample02	211642	26	10	2015.06.05
Sample03	204800	36	7	2015.06.16
Sample04	413696	29	19	2015.06.17
Sample05	272896	36	7	2015.06.18

Repeated experiments shown that the number of active proxies decays very slowly in a time frame of 40 days. What is troubling - only one of the samples lost all of its proxies and became harmless. Worth noticing is fact, that count of active proxies do not decrease monotonically - some proxies became reactivated. It might have numerous causes, like attacker changing proxy's DNS resolve address or some machine being restored from infected snapshot. Overall it proves low awareness of systems administrators, allowing malware to function undisrupted for months.

Having connection logs from the HoneyPot on the Maltester's Auxiliary host and HoneyProxy machine, the statistics on the geographical locations can be made. Figure 5 presents the distribution of the CryptoWall proxies. The number of unique proxy servers gathered from all 6 malware samples is 116 (compare with Table 1).

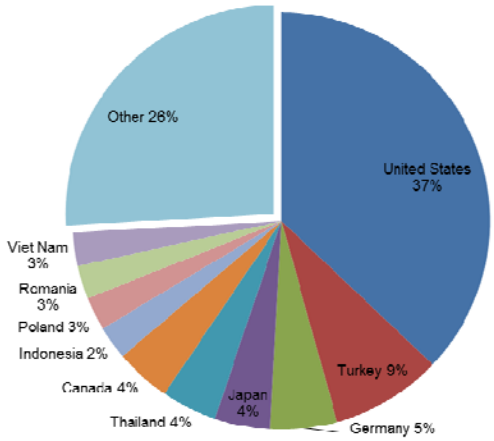

Fig. 5. Geographical locations of the CryptoWall proxies.

Figure 6 presents the geographical distribution of unique infected machines that contacted to our HoneyProxy machine. So far (after about a month) 30344 connections were observed from 1587 unique IP addresses. Daily 1145 connections is recorded on average.

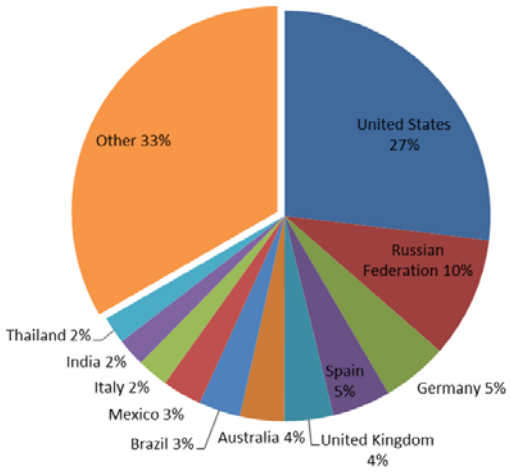

Fig. 6. Infected machines connecting to our HoneyProxy.

Summary

The presented analysis and its results proves the advantages and usefulness of dynamic analysis concept and Maltester environment. Identification of all the infected proxy machines playing an important role in the complex CryptoWall's infrastructure took about 15 minutes.

Having identified these machines and a basic knowledge of the ransomware's behavior and protocols used, it is quite easy to block its activity – the regained proxy can help to identify ransomware victims. The scale of the CryptoWall is terrifying as during a single month our HoneyProxy has identified 1587 machines. It is worth to note, that HoneyProxy administrator got an official warning note from the one of the Polish CERT teams about the infection a month after the regain. What is interesting, there were still some proxies - webservers - hosting the malicious script for 40 days at the moment of writing this publication.

Static analysis would never answer some questions, like how many machines used by the attacker is still responding at a given time. For this purpose Maltester had to be reconfigured several times in order to obtain pieces of information one by one. For example, when successful transmission via proxy was observed, the proxy's IP address was blocked and next dynamic analysis was executed to identify another proxy. So, only combining static and dynamic analyses can lead to the most valuables results.

REFERENCES

[1] Cabaj K., Gawkowski P., HoneyPot systems in practice, *Przegląd Elektrotechniczny*, 91 (2015), nr 2, 63-67

[2] Cabaj K., Grochowski K., Gawkowski P., Practical Problems of Internet Threats Analyses, in: Theory and Engineering of Complex Systems and Dependability, Advances in Intelligent Systems and Computing, 365 (2015), 87-96

[3] Cabaj K., Denis M., Buda M., Management and Analytical Software for Data Gathered from HoneyPot System, in: Information Systems in Management, 2 (2013), nr 3, 182-193

[4] Cabaj K., Visualization as Support for Web Honeypot Data Analysis, w: Information Systems in Management, WULS Press Warsaw, 4 (2015), nr 1, 14-25

[5] Ulrich Bayer, Andreas Moser, Christopher Kruegel, and Engin Kirda. "Dynamic analysis of malicious code," Journal in Computer Virology, vol. 2, pp. 67-77, 2006

[6] Lim, C.; Ramli, K. "Mal-ONE: A unified framework for fast and efficient malware detection", Technology, Informatics, Management, Engineering, and Environment (TIME-E), 2nd International Conference on, (2014), 1 – 6

[7] Baecher P., Koetter M., Dornseif M., Freiling F.: The nepenthes platform: An efficient approach to collect malware. In Proceedings of the 9th International Symposium on Recent Advances in Intrusion Detection (RAID'06) (2006)

[8] Mihai Christodorescu, and Somesh Jha, "Static Analysis of Executables to Detect Malicious Patterns," Univ. of Wisconsin, Madison, US. (2006)

[9] Xu, M., Wu, L., Qi S., Xu, J., Zhang, H., Ren, Y., Zheng, N.: A similarity metric method of obfuscated malware using function-call graph. *Journal in Computer Virology*, 9 (2013), Issue 1, 35-47

[10] Chen X., Andersen J., Mao Z.M., Bailey M., Nazario, J., Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware," IEEE International Conference on Dependable Systems and Networks, (2008), 177-186

[11] Moser, A.; Kruegel, C.; Kirda, E., Limits of Static Analysis for Malware Detection, Computer Security Applications Conference, (2007) 421 - 430

[12] McAffe Labs, Threats Report, May 2015

[13] Duncan B., "Increase in CryptoWall 3.0 from malicious spam and Angler exploit kit" http://dshield.org/forums/diary/Increase+in+CryptoWall+30+from+malicious+spam+and+Angler+exploit+kit/19785/

[14] Dingledine R., Mathewson N., Syverson P., Tor: the second-generation onion router. In Proceedings of the 13th conference on USENIX Security Symposium, 13 (2004), 21-21.

***Authors**: dr inż. Krzysztof Cabaj, Politechnika Warszawska, Instytut Informatyki, ul. Nowowiejska 15/19, 00-665 Warszawa, E-mail: K.Cabaj@ii.pw.edu.pl; dr inż. Piotr Gawkowski, Politechnika Warszawska, Instytut Informatyki, ul. Nowowiejska 15/19, 00-665 Warszawa, E-mail: P.Gawkowski@ii.pw.edu.pl; mgr inż. Konrad Grochowski, Politechnika Warszawska, Instytut Informatyki, ul. Nowowiejska 15/19, 00-665 Warszawa, E-mail: K.Grochowski@ii.pw.edu.pl; inż. Dawid Osojca, Politechnika Warszawska, Instytut Informatyki, ul. Nowowiejska 15/19, 00-665 Warszawa, E-mail: D.Osojca@ii.pw.edu.pl,.*

The correspondence address is:
e-mail: K.Cabaj@ii.pw.edu.pl