Proceedings of the Federated Conference on Computer Science and Information Systems pp. 981–989
DOI: 10.15439/2016F490
ACSIS, Vol. 8. ISSN 2300-5963

Developing malware evaluation infrastructure

Krzysztof Cabaj, Piotr Gawkowski, Konrad Grochowski, Amadeusz Kosik
Institute of Computer Science
Warsaw University of Technology
ul. Nowowiejska 15/19
00-665 Warsaw, Poland
Email: {K.Cabaj, P.Gawkowski, K.Grochowski, A.Kosik}@ii.pw.edu.pl

***Abstract*—Malware evaluation is a key factor in security. It supposed to be safe and accurate. The contemporary malware is very sophisticated. Usually it uses complex distributed infrastructure an investigation of which is a very challenging task. In the paper, the development of the testbeds toward malware and its infrastructure evaluation is presented. Based on the real-life experience with the subsequent CryptoWall generations analysis, the MESS evaluation system is introduced. A rich set of analytical results is discussed. A new methods of visualization for malware artefacts analysis are given.**

I. Introduction

In the last decade the main motive of attackers' actions was associated with money. In the previous years the most precious treasures were credit cards numbers or data used for accessing to e-banking systems. However, it is worth mentioning that reaction to these threats from financial organizations made such attacks harder. From the last few years, more and more popular are attacks that lock victim's computers and demand some ransom for enabling access to the infected machines. Due to a ransom request, any malware used during these attacks is called ransomware. Reports prepared by antivirus companies show a huge increase in this kind of attacks in the last two years. For example, McAfee shows that only in the first quarter of the 2015 year, the number of observed ransmoware samples rose by 165% [1]. Symantec shows even more horrifying data - accordingly to its report number of ransomware which encrypts files in the hard drive rise almost 45 times, from 8274 samples observed in 2013 to the 373342 in 2014 [2].

At the end of March 2015 our security group had to clean-up an infected machine in the Institute of Computer Science. That malware sample (CryptoWall 3.0) was then examined using dynamic analysis. Performed analysis revealed very interesting behavior concerning the network activity of the ransomware. After the infection the sample contacts attacker's Command and Control server using a list of prior infected Web servers. We called this kind of a Web server a *CryptoWall proxy*. What should be emphasized, these servers are innocent victims, too. Analysis of few more CryptoWall samples showed that these lists of proxies contain many infected servers, and these lists are centrally managed by attackers. Detailed description of this network activity and results from its initial analysis can be found in [4].

During continuation of research many samples of CryptoWall family were soon gathered. Manual analysis of all samples soon became almost impossible. So, the ARTA system (Automatic Ransomware Traffic Analyzer) was develop and deployed. It consists of several modules. The ARTA has a dedicated subnet with a set of HoneyPots and a DNS redirecting the whole traffic to these HoneyPots. The malware sample is executed within the dynamic evaluation system Maltester (see [4]). Moreover, the whole system is remotely controlled with dedicated the Web application. Results and practical experience gathered with ARTA is presented in [9].

The advantages of using ARTA are really great. However, two things are missing (even if the lack of their presence should not be considered as drawbacks). First of all, in ARTA, the whole network traffic was enclosed within the HoneyPots. It is a safe solution and allows to identify the basic (i.e. initial) communication made by the malware. However, there is no further knowledge about the liveness of external parts of malware infrastructure (i.e. nodes it tries to communicate with). Also subsequent communication of the sample is not known (due to limitations of HoneyPots). Secondly, the Maltester is not designed to provide information about detailed actions taken by the sample within the target system. Maltester allows sample execution and comparison of system state only after the sample evaluation is finished. It is an environment for evaluation of malware in a Xen-based virtualized host by state comparison (only network traffic is monitored on-line from the outside).

There are some ready-to-go solutions, like Cuckoo Sandbox [10] - due to its popularity and availability it is common to meet malware samples that detect being executed in environment like this. Another popular system, Anubis (exposed as a service in web application) is no longer available as its developers has created their own company with malware analysis services. That is why it is important to build own solutions. Moreover, as the malware dynamically evolve, the evaluation infrastructure must be open for fast developing. So, we decided to develop and implement our own solutions.

In this paper we present the Malware Evaluation Support System - MESS - an environment based on Hyper-V virtualization that uses on-line, on-site monitoring of the malware activity. Comparing to Maltester, it delivers not only information about changes made by the malware but also how the execution proceeds. One of the main advantage on

978-83-60810-90-3/$25.00

MESS is the ability to remotely control the analysis process including the security settings of the network traffic. This capability was used in long-term experiment with the rich set of Cryptowall ransomware samples. One of the goals was to identify and observe the life-cycle of the so-called *proxy servers* being a vital part of the Cryptowall infrastructure. For the analysis we also propose a graph-based method that, in our opinion, facilitates identification of the most interesting artefacts of malware infrastructure. In the paper the MESS test-bed, methodology and results of different kinds of analysis are presented.

In the next section the basic differences between different kids of malware analysis are discussed. Section III generally presents the most contemporary malware type - ransomware. In more detail the behavior of a CryptoWall family ransomware is described. Then, the insights into the MESS test-bed are given in section IV followed by the description of experiments automation (section V). The paper presents the obtained results in section VI.

II. Malware analysis problems

Analysis of malware can be conducted in several ways. First of all, statically - with the analysis of the de-assembled/decompiled code. Generally, such analysis can be very effective. However, in the case of malicious software it can be challenging or even impossible: to cheat anti-virus scanners and to obstruct such analysis, the malware usually implements several obfuscating techniques [5], [6], [7]. For example, some techniques introduce dynamic code modifications (upon execution - decryption using XOR or ROT13 on some code blocks, garbage instructions overwritten with NOPs, return statements without previous calls).

In dynamic analysis, the black-box model is assumed - the examined application is analyzed through its behavior. Such analysis requires malware execution along with some dedicated monitoring utilities [6]. Gathered logs are then used to investigate actions taken by the malicious code on the host. Here, two main problems arise: how to safely execute malicious software and how to identify malicious behavior. Virtualization may address the problem of maltware sand-boxing. Another advantage is scalability - different malware samples can be examined in parallel and the guest system can be efficiently prepared for the evaluation using snapshot for state recovery. On the other hand, there is a risk of virtualization hypervisor disclosure [7]. One of the simplest way to identify virtualization is to check if the hard disk contains any user activity related files (e.g. changed desktop background, web browser temporary files). More sophisticated one is checking in the system registry for CPU information and verification of the number of threads with the declared by the CPU manufacture.

Creating an environment for malware evaluation a key issue is to assure security of other IT resources in the neighborhood. As the contemporary malware often requires Internet connection to be fully operable, the connectivity limitations may also significantly limit the analysis (e.g. unavailability of command-and-control servers, downloading of other malware). At the same time it is obvious that during our experiments we would like to protect our infrastructure as well as limit possible attacks made by the executed malware. So, in particular, the ports 25 and 587 should be blocked to not allow spamming over the Internet.

Dynamic analysis can be made on-line - the malware execution is monitored while its execution - or off-line - the analysis is based on the comparison of the system state before and after the execution.

In the second case, the analysis is mainly focused on changes in the file-system and system registry (i.e. new, deleted, renamed files and directories, registry entries). The main advantage is low probability of analysis disclosure but the temporal analysis is very limited. In fact, only the network traffic can be analyzed in details as it can be gathered from the outside of the host.

The most detailed information can be collected with the real-time monitoring of malware execution on the host itself. In this case the probability of monitoring disclosure by the malware is very high. Especially using debuggers can be easily identified - it interfere with the execution much more than other monitoring utilities [5], [7], [8].

III. CryptoWall Family Ransomware

The first generation of ransomware only locks access to the computer, preventing logging to the machine. For many skilled users these threats can be easily overcome. In the most severe cases full system reinstallation is needed. However, all user's data stored in the infected machine can be restored. Due to this fact, shortly, a second generation of ransomware become popular which works in more hostile fashion.

In its second generation the malware encrypts various types of files associated with user precious data generated by, for example, word processors, spreadsheets or games - yes, some ransomware encrypts games' saves files. As in most cases the ransomware uses modern encryption algorithms, like AES (Advance Encryption System), the decryption without the key is almost impossible. The first malwares of this generation utilized symmetric-key algorithms, which use the same key for encryption as well as for decryption. In effect, this key could be extracted from its poor implementation (key was not deleted after encryption of the whole data) or during its transfer from the victim's machine to the attacker.

The one of the most sophisticated ransomware family is called CryptoWall which uses an asymmetric-key encryption algorithm. Such algorithm uses two separate keys: public - used for encryption and private - used for decryption. In this situation both keys are generated somewhere in the Internet and only the public key used for encryption of users' data is transferred to the infected machine. Private key used for decryption never appears in the victims' machine. Considering that this malware uses 2048 bit RSA asymmetric-key algorithm, decryption of victim's data without the private key is unfortunately impossible. Detailed analysis of various ransomware families can be found in [3].

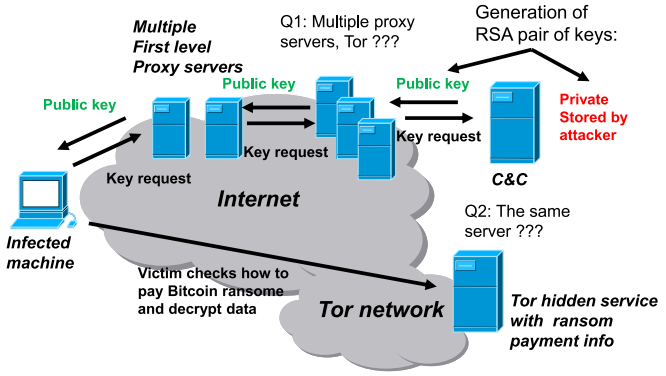

Fig. 1. CryptoWall infrastructure overview

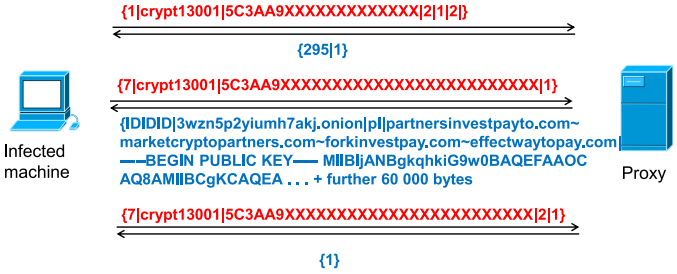

Fig. 2. CryptoWall 4.0 communication overview

The CryptoWall family, due to utilization of asymmetric cryptography, was one of the most sophisticated ransomware in the 2015 year. Usage of this type of cipher has big advantage in contrast to previous ransomware families that utilize symmetric cryptography - the key required for decryption is not present in infected machine at any moment of ransomware activity. However, the one disadvantage of this approach is a need of contact between a victim and an attacker's command and control server, which generates asymmetric key pair and provides a public key used for data encryption.

At the end of January 2015 a new version was observed - CryptoWall 3.0. This version uses infected web servers for hosting proxy script that hinder the location of attackers' command and control server. Detailed description of used communication protocol was presented in [4]. Fig. 1 presents CryptoWall infrastructure.

At the beginning of November 2015, the CryptoWall 4.0 came out. Despite the similar protocol (at first glance), we failed to decrypt its communication for more than a month. Fortunately, due to attackers' mistake, at 6th of December, one of the proxy servers, instead of the execution of the malicious proxy script, was sending its copy. Analysis of the script source code revealed that (in comparison to the previous versions) it has new four lines of code. This part of the code removes some random bytes added by the attacker at the beginning of the encrypted data (within the communication protocol). Probably, this change is introduced for hindering CryptoWall 4.0 activity from detection by Intrusion Detection System. The previous version uses messages that have almost the same length in particular communication phase during communication with proxy. Analysis of decrypted messages revealed a second change in comparison to the CryptoWall 3.0 - the protocol used is simplified. Instead of five transmissions, CryptoWall 4.0 exchanges only three. Decrypted communication of this CryptoWall version is presented in the Fig. 2.

The first transmission (message exchange) informs the attacker that a new machine is infected. This message contains the message of type one (see the first number in sent request - red color), the name of the used campaign (e.g. crypt13001 - see Fig. 2, the machine unique identifier and the encoded description of Windows operating system version. The next transmission (message with the type of 7) is responsible for downloading the public key and the personalized image presented to the victim. The last transmission confirms reception of all the data needed for the encryption process.

IV. Malware Evaluation Support System

Malware Evaluation Support System (called MESS later on) is a system toward dynamic monitoring in real-time of malware sample execution. Contrary to Maltester, the data collection is made on-the-target machine during the runtime.

A. MESS architecture

It consists of several components, as depicted in Fig. 3:

- Executor - responsible for malware sample execution and on-site monitoring tools. It resides on a target system on which the malware sample is executed - a virtual machine VMx
- NAT/Firewall - responsible for recording and filtering the network traffic to and from the Executor systems from/to the Internet
- Controller - responsible for coordinating the whole MESS infrastructure and interaction with the user
- Supervisor - responsible for controlling the Executors through the hypervisor management API.

All these components can be located on a single physical machine or on multiple virtualization servers. In the first, simplest scenario, the Controller and the NAT/Firewall can be located on a single virtual machine along with a set of Executor - separated virtual machines.

The Supervisor (in order to manage the virtual machines) has to be located within the physical host system. In more complex infrastructure (as in Fig. 3) MESS scales-up with the number of physical virtualizators (each requires its own Supervisor) and their virtualization capabilities (on each physical virtualization host a set of virtual machines VM1-VMx with the Executors can be used in parallel). Theoretically, MESS can utilize several different hypervisors (if a proper Supervisor component is available), however, at the moment only Microsoft Hyper-V is supported.

Within the MESS three networks are defined. The NAT/-Firewall machine has to have three network interfaces. The first one is connected to the Internet. The second one serves as a communication channel with the Controller. The third one

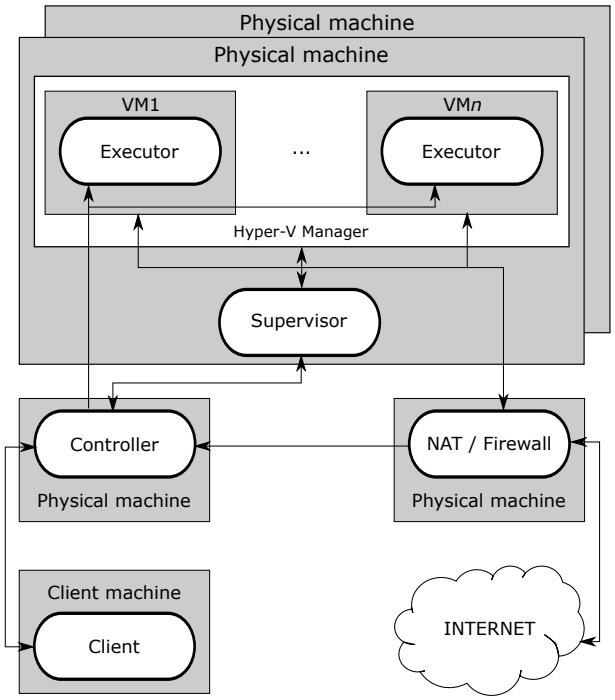

Fig. 3. MESS Architecture

serves the Internet connectivity for the target virtual machines with the Executors. However, due to security restrictions, that network is filtered. Executor's machine is not allowed to communicate with local network as well as with physical LAN (university network in our case). As some ports are commonly used to spread malware (e.g. through automatically sent spam) these ports are also disabled (e.g. 25, 587). During the analysis the settings of the firewall can be changed.

B. Usage scenarios

Typical usage scenario consists of several steps. First of all, a malware sample has to be executed within specially crafted virtual machine. That means, that a set of user-level applications with typical vulnerabilities and a desired set of operating system updates should be installed. A set of system services should be running as well as some user-level emulators (to emulate users' activity). After preparation of such environment, the virtual machine should be frozen in the snapshot. Later on, it will be rolled back to this state before each sample analysis.

A user has to choose at which target system snapshot he wants to conduct the analysis, as well as the malware sample and its filename at which it should be saved in the target machine. He can also pass a set of tools and scripts (in PowerShell) to be executed upon:

- before actual sample execution - some additional preparation tasks
- before restart of the virtual machine - sometimes an analysis might require to restart the machine
- before the end of the analysis - before the gathered results of monitoring utilities are prepared for sending to the user

If the chosen virtual machine is ready, the Supervisor rolls its state to the one saved in the chosen snapshot and runs it. Then, all the informations (scripts, additional monitoring tools are the malware sample) are transferred to the Executor component on the target machine. The monitoring tools are started and finally, the sample is executed - the actual analysis begins. During the analysis any user actions are not required, however, the user may request target machine restart, can interact with the target system (e.g. with remote desktop or console) or preview the network activity. The option to restart the machine is MESS unique feature. Some malware samples, do not start the main activity during the first execution - they just setup itself within the system. In such case, the system has to be restarted to observe the malware. In MESS, the target system will continue the analysis after such restart.

The sample analysis can end in several ways. In normal scenario the MESS is requested to stop the analysis. Then, the Executor on the target system stops the monitoring tools, executed proper user-defined scripts and the results (from embedded monitoring tools as well as the user's) are provided to the MESS as a ZIP file. That file is downloaded by the MESS and sent to the user. After this step, the target virtual machine is stopped and rolled back to its initial state, and ready for the next analysis. Sometimes the user may be not interesting in the results or some unusual circumstances occur (e.g. the sample become a part of DDoS attack or do other unpredicted actions). Then, the analysis is interrupted without result preparation and downloading.

By default, MESS uses Process Monitor tool from the Windows Sysinternals suite to register all the actions made by the sample [11]. It can trace events like system registry accesses (reads, writes of keys and values), filesystem operations and thread/process management. Moreover, the tool provides a rich GUI functionality for further data analysis in off-line. For dumping the network traffic, the Wireshark is used directly on the target system. However, the whole traffic can also be registered on the NAT/Firewall machine.

C. Component integration

Communication between MESS components is implemented with REST approach and XMLRPC protocols (both using HTTP beneath). The REST (Representational State Transfer) is used between the Supervisor(s) and the Controller. It was chosen because of its simplicity, very simple implementation on both, the client and the server side. Moreover it is independent to the implementation technology. It is very important aspect, as the Supervisors (as mentioned earlier) can be located on different kind of operating systems and environments.

Communication between the Controller and the Executors is implemented with XMLRPC (XML Remote Procedure Call). Functionally, it allows to implement remote-like procedure calls. All the parameters and results (even collections or binary files) are passed between the client and server sides very

convenient way. The external interface Controller (from the user side) is also implemented with XMLRPC. Thanks to that, the MESS can be easily integrated with external management systems or scripts.

In order to implement these quite complex tasks, the MESS components consists of subcomponents. For example, the Executor consists of a dedicated system service for keeping the analysis context between system restarts, HTTP service to provide convenient way of analysis result downloading and user-level application for sample startup.

D. Remarks

After several months we gained a lot of experience in using MESS. One of the biggest challenge is to properly prepare the target system environment and trim the monitoring tools. From the one side, the one is interesting in many details of the actions taking place but most of them are not anyhow related to the analysed malware activity. The target system or user-related applications may generate a lot of actions, because, for instance, automatic update services, background tasks etc. It has to be stressed that the Process Monitor can provide very rich set of data. That is great, but the result file might be huge. The proper filter may significantly limit the size of gathered data. We faced these problems as we missed (by oversight) that the Opera browser within the target system was left in the snapshot in the state just before the update. In effect, soon after each analysis, the Opera was starting update downloading and installation. That introduced additional traffic (ca. 40MB) and a lot of system actions (registry and filesystem related).

To properly conduct an analysis the target system should be evaluated in different configurations. Using only updated system might be "too good" for the analysed malware. On the other side, using "too old" version of the system (like WindowsXP, or very outdated version) might be useless (unusual configuration) or suspicious for the malware sample (see section II.

As the MESS utilize Hyper-V technology, there is a risk that the malware sample will detect the presence of the hypervisor. Typically, such sample simply terminates the execution without any malicious actions. Generally, we have met such situation only for a few samples. Only one sample has detected Hyper-V. It was stressfully analyzed in Maltester then. The other sample has detected the Xen hypervisor of Maltester. In this particular case, the sample executed in MESS, after some operations suggesting hypervisor detection procedure, failed to detect Hyper-V and was successfully analysed. These cases proves that both solutions are complementary.

V. Experiment Automation

Manual analysis of each available sample in attempt to retrieve all active servers utilized by it proves to be tiresome and time consuming process. Fortunately preliminary results of manual analysis helped developing tools which used available experiment environment capabilities to automate the process.

Algorithm 1 briefly presents automated experiment procedure. For clarity timeouts calculation and detection in lines 5 and 9 are not shown. Algorithm contains single procedure, repeated for each malware sample. Each sample is analyzed as long as it tries to connect to new servers. During preliminary analysis of malware it was detected that CryptoWall communicates with its servers in semi-random order. It starts to repeat that order after three attempts to connect to each server. That lead to condition used to detect completeness of the sample analysis (line 5). To mitigate potential transient communication problems, configurable timeouts where also applied in that condition, forcing the timed out experiment to be repeated for the given sample. To ensure high quality of gathered data, each active server was detected using separate execution of the sample in the controlled environment (line 6). After executing sample, experiment controller was monitoring network communication for occurrence of incoming malware server response or until some timeout passed (line 9). If response from server was received, it was added to the experiment results, and communication with IP address of that server was blocked for future experiments with the same sample, forcing that sample to try to stimulate another server (lines 10-13). Nevertheless reason for stopping sample run (line 9), all malware requests sent by this sample were gathered (lines 14-18) and used for experiment completeness condition (line 9). For next sample run, all network traffic blocks were lifted (line 4). Procedures used for detection of malware requests and responses were prepared during manual analysis.

Algorithm 1 Acquiring list of active malware servers.

```
1:  for all Sample ∈ MalwareSamples do
2:    Sample.Servers ← ∅
3:    Sample.ActiveServers ← ∅
4:    Unblock all network traffic
5:    while ∃ S ∈ Sample.Servers : S.Retries < 3 do
6:      Launch Sample in controlled environment
7:      repeat
8:        Monitor In-coming and Out-coming traffic
9:      until ∃ P ∈ In : IsMalwareResponse(P)
10:     if ∃ P ∈ In : IsMalwareResponse(P) then
11:       Block network traffic to and from P.SourceIP
12:       Add P.SourceIP to Sample.ActiveServers
13:     end if
14:     for all R ∈ Out do
15:       if IsMalwareRequest(R) then
16:         T ← Sample.Servers[R.TargetIp]
17:         Increment T.Retries
18:       end if
19:     end for
20:   end while
21: end for
```

VI. Cryptowall infrastructure analysis

As was described in the section III, communication to hinder detection infected machine connects to the command and control server via so called proxy servers. These servers are

hacked by the attacker and special proxy script was installed into web server. To raise the chance that an infected machine successfully download public key, each sample of CryptoWall malware contains hard-coded list of multiple proxy servers. The sample tries to connect in a sequence at the beginning of the infection. Our initial analysis reveals that various samples have the same list of proxy servers. Due to this fact we decided to cluster analyzed samples using proxy list as unique group identifier. Our analysis concerns almost 360 samples taken from openly available sources:

- blog `malware-traffic-analysis.net`
- `malwr.com`
- `reverse.it` services

To manage the collected samples during our research, they can be additionally tagged (beside a number) to be easily identified (*cw3-Feb* - for a sample of CryptoWall 3.0 obtained in February, *cw3-Mar*, etc.).

The samples use 59 unique proxy lists; average proxy list contains almost 40 unique URL, however maximal observed number of URL was 70. During our research we detect more than 2000 unique URLs, hosted in 1945 domains. Detailed analysis concerning domains and addresses is presented later in this chapter. Initial analysis of gathered data was performed with the help of graph theory. Data is used for generation of graphs, which represent connections between proxy lists and used domains. In the constructed graph two types of vertexes are introduced - blue and green. The blue vertexes represent name of proxy list. The green vertexes represent domains. Connection between vertexes indicates that this particular proxy list contains URL which is provided by server in this domain. Analysis of constructed graphs can reveal interesting patterns rapidly and help the person, who is performing analysis of gathered data. Analysis of samples from the beginning of the 2015 shows, that each new proxy list uses completely independent set of domains. Plotted graphs from this period are rather simple, especially in comparison to more complex plots from the end of the year. Fig. 4 presents sample graph of this type, in our security team called "flower".

However, from the middle of the 2015 year we started to observe more connections between various proxy lists. Sample graph of this type is presented in the Fig. 5.

As can be seen in the presented image there are many domains which are associated with two or even with three distinct proxy lists. What is interesting, some domains are used both for samples of CryptoWall version 3.0 and 4.0. This is an evidence that this malware is operated by one group of attackers. Moreover, in our opinion this reuse of domains can be sign that attackers have some problems with hacking or buying new machines, which hosts proxy script for various complains. Additionally, due to vast amount of data, rapid finding of interesting domains which should be investigated in the first row is very important. For this purpose graphs constructed in this fashion, can be beneficial, too. The most interesting domains are those, which connects two or more proxy lists. Shouting down such proxies we can eliminate the broadest spectrum of malware. These domains can be easily

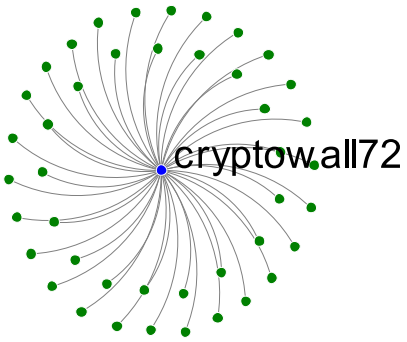

Fig. 4. Simple graph of member domains (green vertexes) associated with given proxy list (blue vertex)

automatically found, they are a green vertexes which degree value is greater than one. Additionally to the static analysis of all domain contained in each malware sample, we performed analysis which reveals information how many proxy servers in given instant provides access to the command and control servers. Accordingly to generally published information infected servers are easily detected and rapidly shut down by administrators. Our research confirms the first part of this statement. Unfortunately, we cannot agree with its second part. The most long lived proxy server observed during our research, allows access for victims to the C&C server for 11 weeks and 1 day. What is alarming, such servers are common. Fig. 6 presents how many servers in given proxy servers list associated with for CryptoWall 3.0 still allows access to the attackers C&C.

We executed samples in controlled environment, provided by the Maltester and MESS dynamic analysis systems. After successful reception of the public key form the C&C server we marked this server as alive, block its IP address in firewall and execute another analysis. Due to manual method we could perform no more than one check of given proxy list in a week. Because achieved in such way data was very valuable we decided to develop and deploy automatic system which could perform analysis more frequent. Details of the system were presented in the section IV. In the plot two instants are very interesting: the one in the middle of the September and the second just before the end of the December (both are marked in the figure with red arrows). In these two instants almost at the same time all proxy servers stopped forwarding the traffic to the command and control server. Due to the fact that these servers are placed in various countries such simultaneous actions with high probability was performed by the attackers. The first situation confirms our assumptions, because almost immediately many new samples of CryptoWall 3.0 come out with completely new proxy servers list. The second event marks the end of the CryptoWall 3.0 activity. After this we

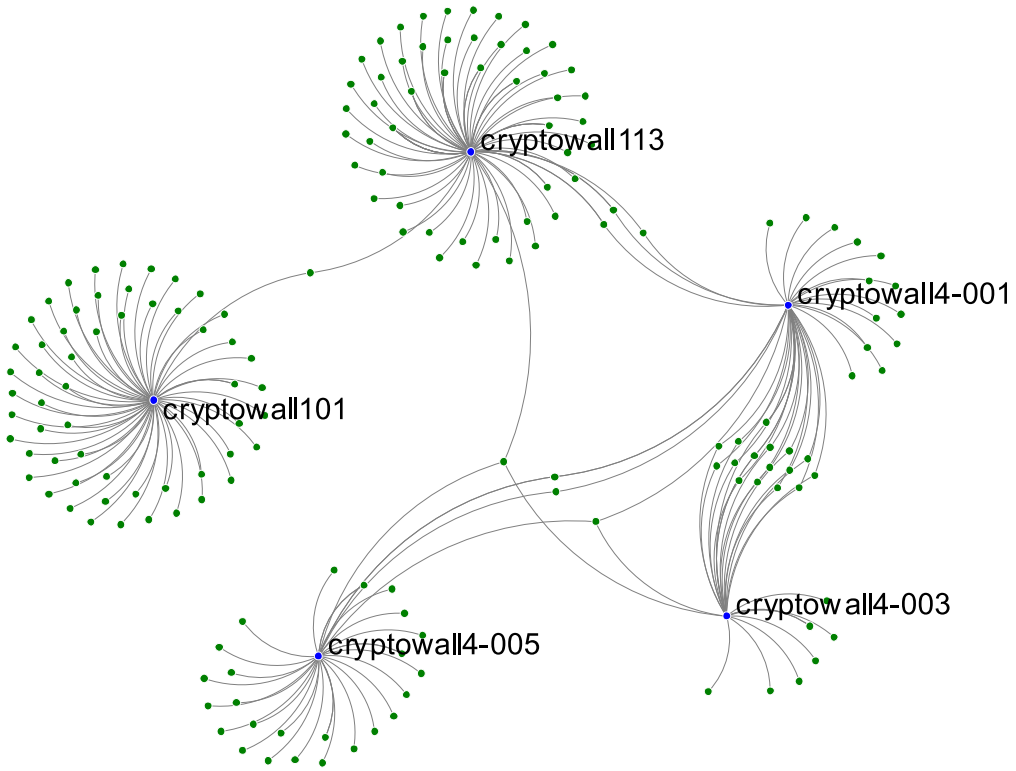

Fig. 5. Complex graph of member domains of proxy lists and its associations, observed at the end of 2015

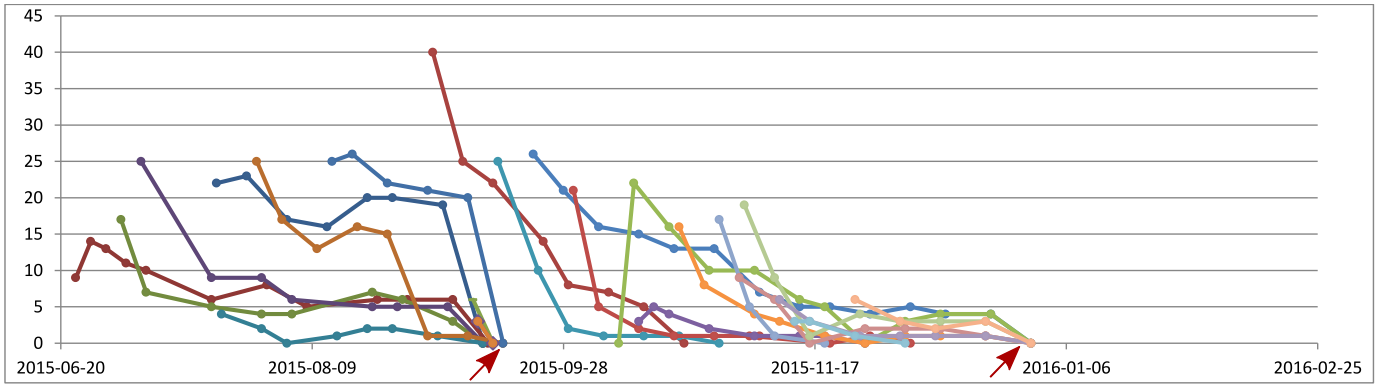

Fig. 6. Number of active CryptoWall proxy servers during 2015

have never observed responding proxy servers for CryptoWall 3.0 samples. However, the new threat came out - CryptoWall 4.0.

In addition we investigated the domain IP addresses and countries of origin of the infected proxy servers. The second aspect is very important, because it specifies to which national CERT or law enforcement agency (LEA) report concerning detected hostile activity should be provided. At the beginning, this aspect of analysis seems to be very simple. We have at least two source of such information: top level domain or geolocalization information associated with IP address. However, our research shows that this information can be inconsistent. We observed numerous examples, when country associated with DNS top level domain was other than country provided by the geolocalization database. To eliminate errors in gelocalization database, we investigated real localization of a server using `traceroute` utility program. In all such situations, information provided by the geolocalization database was accurate - the last few routers observed in the output have country top level domain associated returned country.

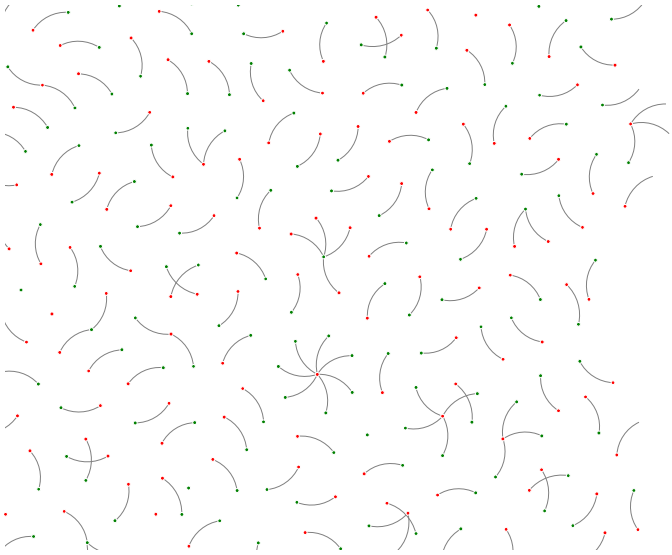

Fig. 7. Examples of graphs presenting associations between domains (green vertexes) and used IP address (red vertexes)

The good thing from this situation is that we could provide information concerning infection, both: to country of top level domain as well as to country where physically server is located. The vast amount of detected domains and IP addresses cannot be investigated all in short time. Due to this fact we introduces method which shows the most interesting data, which should be investigated in the first place. For this purpose we construct custom graph which have two types of vertexes - red and green. The green vertex represents detected domain. The red vertex represents associated with detected domain IP address. Connection between green and red vertexes determines that this domain is resolved to this particular IP address. Fig. 7 presents sample visualization of graphs, constructed in described manner.

Presented plot at first look is completely illegible. The most of presented graphs have only two vertexes, and these simple irrelevant ones hinder interesting knowledge. The first step in analysis of such data is removal of all irrelevant graphs. In data recorded during analysis of CryptoWall there are 1286 such graphs. Remaining ones consist of more than two vertexes. What is interesting, in remaining graphs only three have all vertexes which degree is greater than one. They are presented in the Fig. 8 (left). In remaining graphs there is always one vertex with degree greater than one and all other vertexes have degree equal to one. These graphs can be divided into two categories, depending on the type of vertex, which have degree greater than one. Two types of such graphs are presented in the Fig. 8 (right).

The first type of graph, with green vertex with degree greater then one, represent domains that have multiple IP addresses. The second one in contrast have multiple domains which are hosted in one IP address. The latter one is very promising from security perspective. In such situation, disabling this one address, can stop all domains hosted on it. Our research

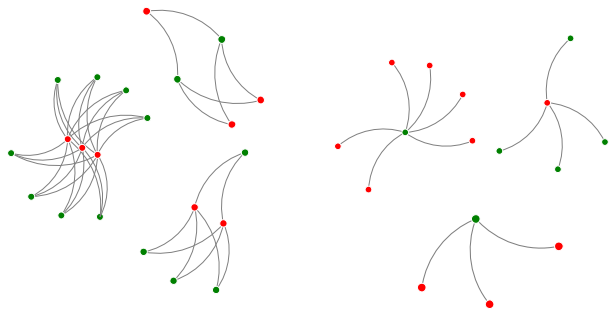

Fig. 8. Some of the most interesting graphs: with all vertexes of degree greater than one (left). Graphs with only one vertex of degree grater than one (right)

shows that attackers from the middle of the 2015 start reusing this same domains. In effect shutting down such domain can protect not only this one analyzed complain, but some before unknown, too. The first type of graphs can be useful, too. Because this same domain is hosted on various IP addresses, they are managed by on organization. In effect one contact with responsible person, can deactivate all of them. Results of our research lead to methodology which can be used for automatic prioritization of received date. In the first step graphs concerning domains and used IP addresses are constructed. In the second step degrees of all vertexes are calculated. These, which all vertexes have degree one, are removed from data presented to the person performing analysis. In effect, the most interesting from the security perspective domains or IP addresses can be easily detected and presented to the security officer in the first row, which can speed up whole process of finding and disabling hostile machines in the Internet.

VII. Conclusion

Starting on March 2015 the CryptoWall ransomware become a point of the authors interest. In order to evaluate its behaviour a rich set of experimental runs were conducted. To make these analysis safe and effective a special toolset and methods are needed.

Based on previous experience with malware analysis, the new, dynamic on-line analysis system, called MESS, was proposed. It proved to be an effective solution toward automated analysis, in particular, in data collection upon the malware behaviour within the operating system. The Hyper-V virtualization is quite effective approach in our case. There is a need to operate on different platforms in order to avoid hypervisor detection. In this sense, the MESS is complimentary to other similar systems.

Analysing the nowadays malware it has to done in two domains: actions within the target system and actions (i.e. communication) over the Internet. In the second domain, it is important to discover and investigate the infrastructure associated with the malware. Sometimes, to restrain the spread and side effects of malware, the easiest way is to identify and limit connectivity to its proxies or Command&Control servers.

In the paper we present the methodology of identification of CryptoWall proxies as well as propose new graph-based method for more effective analysis. The sad true is that the proxy servers, even when identified, are very hard to be turned off or cleaned. In several cases the authors successfully contacted proxy administrators (7 in Poland). However, the obtained life-time of the set of CryptoWall proxies is not optimistic. Further work will concentrate on the analysis of new malware families, like Locky or TeslaCrypt.

ACKNOWLEDGMENT

We would like to thank Wojciech Mazurczyk PhD., Dsc. for his assistance with finding and labeling samples of CryptoWall ransomware provided in the malwr.com service.

REFERENCES

[1] McAffe Labs, *Threats Report,* May 2015, www.mcafee.com/us/resources/reports/rp-quarterly-threat-q1-2015.pdf
[2] Symantec, *Internet Threat Report,* April 2015, www4.symantec.com/mktginfo/whitepaper/ISTR/21347932_GA-internet-security-threat-report-volume-20-2015-social_v2.pdf
[3] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge and E. Kirda, "Cutting the gordian knot: A look under the hood of ransomware attacks," *DIMVA 2015, 12th Conference on Detection of Intrusions and Malware & Vulnerability Assessment,* July 9-10, 2015, Milan, Italy, http://dx.doi.org/10.1007/978-3-319-20550-2_1
[4] K. Cabaj, P. Gawkowski, K. Grochowski, and D. Osojca, "Network activity analysis of CryptoWall ransomware", *Przegląd Elektrotechniczny,* Vol 91, No 11, 2015, http://dx.doi.org/10.15199/48.2015.11.48
[5] E. Skoundis and L. Zeltser, *Malware. Fighting Malicious Code,* Pearson Education Inc. ; 2004.
[6] U. Bayer, A. Moser, Ch. Kruegel and E. Kirda, "Dynamic analysis of malicious code," *J. in Comp. Virology*, vol. 2, 2006, pp 67-77., http://dx.doi.org/10.1007/s11416-006-0012-2
[7] X. Chen, J. Andersen, Z.M. Mao, M. Bailey and J. Nazario, "Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware," in IEEE Int'l Conf. on Dependable Systems and Networks, 2008, pp. 177-186., http://dx.doi.org/10.1109/DSN.2008.4630086
[8] P. Ferrie, *The "Ultimate" Anti-Debugging Reference*, 2011 http://anti-reversing.com/Downloads/Anti-Reversing/The_Ultimate_Anti-Reversing_Reference.pdf
[9] K. Cabaj, Management System for Dynamic Analysis of Malicious Software, Information Systems In Management, 2015
[10] Cuckoo Sandbox website, https://www.cuckoosandbox.org, May, 2016
[11] Process Monitor website, https://technet.microsoft.com/pl-pl/sysinternals/bb896645.aspx, May, 2016