Computer Forensic Analisys of Some Web Attacks

Nataša Šuteva, Aleksandra Mileva
Faculty of Computer Science
University Goce Delčev
Štip, Republic of Macedonia
{natasa.suteva, aleksandra.mileva}@ugd.edu.mk

Mario Loleski
Forensic Department
Ministry of Interior of the Republic of Macedonia,
Skopje, Republic of Macedonia
mario_loleski@moi.gov.mk

Abstract—**Symantec Internet Security Threat Report 2014 is showing a horrified fact, that when an attacker looked for a site to compromise, one in eight sites made it relatively easy to gain access. Many attackers are arrested due to the evidences obtained by computer forensics. The victim machine usually gives some data, which are then used for identifying possible suspects, which is followed by forensic analysis of their devices, like computers, laptops, tablets, and even smart phones. In this paper, we use an attack scenario on the known vulnerable web application WackoPicko, of three types of attacks: SQL Injection, stored XSS, and remote file inclusion, usually performed by using a web browser. We use post-mortem computer forensic analysis of attacker and victim machine to find some artifacts in them, which can help to identify and possible to reconstruct the attack, and most important to obtain valid evidence which holds in court. We assume that the attacker was careless and did not perform any anti-forensic techniques on its machine.**

Keywords-Computer Forensics; SQL Injection; File Inclusion; XSS.

I. INTRODUCTION

Vulnerability scans of public websites carried out in 2013 by Symantec's Website Vulnerability Assessment Services found that 77 percent of sites contained vulnerabilities, and 16 percent of them were classified as critical vulnerabilities that could allow attackers to access sensitive data, alter the website's content, or compromise visitors' computers (Internet Security Threat Report 2014, [1]). The OWASP (Open Web Application Security Project) Top Ten 2013 [2] offers a list of the most critical Web application vulnerabilities, including different types of injection, broken authentication and session management, cross-site scripting, secure misconfiguration, etc. Many organizations lose their reputation or revenue, because of various hackers' attacks. Today, the cybercrime is a global problem, and the computer forensics is one way to combat it. Computer forensics prepares legal evidences and give answers to many questions of legal systems related to computers. Analyzed forensic images are the primary evidence.

We chose to investigate three types of attacks, SQL injection, stored XSS and remote file injection, which are usually conducted through a web browser. We are interested in what kind of post-mortem forensic artifacts can be found after performing attack on the attacker and victim machine. As a tested web application, we use known vulnerable WackoPicko [3], first introduced by Doupe et al. [4]. Also, we assume that the attacker did not perform any anti-forensic techniques (format, wipe etc.) on its machine. We are aware that conducted research is very platform specific, so our results holds for the dominant Apache web server and Backtrack 5 R3 attacker's machine. But similar artifacts can be also expected on other related attacker/victim platforms, too.

We showed that from the three types of attacks, remote file inclusion and use of shells leave many traces on both machines, most of them in log files on the victim and web history in the attacker.

After Introduction Section, Section II is devoted to attacking scenario, including a short description of vulnerable web application WackoPicko, and detailed description of three performed attacks SQL injection, stored XSS and remote file inclusion. In Section III we give a brief overview of performed forensic analysis of both machines, followed by discussion of the results and final conclusions.

A. Previous work

To our knowledge, there are no many papers for forensic investigation of web attacks. Andrade and Gan [5] investigate passive attacks for determination of vulnerabilities of Linux Ubuntu server, using Linux BackTrack 5 tools, including Metasploit, Nessus, Whatweb, Nmap, PHP-Backdoor and Weevely. They use netstat tool and server log files for forensic investigation of the attacks. Good forensics analysis of Linux RAM is given in [6]. Shulman and Waidner [7] show how digital signatures from DNSSEC can be useful in forensic analysis.

II. ATTACK SCENARIO

For the attack, we use virtual WM Ware machine with installed BackTrack 5 R3 and with IP address 192.168.60.159.

A. Vulnerable web application

The vulnerable WackoPicko application is a photo sharing and photo-purchasing site. Users of WackoPicko can upload photos, browse other user's photos, comment on photos, and purchase the rights to a high-quality version of a photo. It has 10 vulnerabilities accessible without authentication (reflected and stored XSS, reflected XSS behind JavaScript, predictable Session ID for admin, weak admin password, reflected SQLI, command line injection, file inclusion, unauthorized file exposure, and parameter manipulation), and 6 vulnerabilities

accessible after logging into the web site (multi-step stored XSS, stored SQLI, directory traversal, forceful browsing, logic flaw and reflected XSS behind Flash).

The web server hosting WackoPicko and used in our experiments was run on the OWASP Broken Web Applications Project virtual machine [8], which has numerous intentionally vulnerable applications (we ignore other applications). The following technologies are used: Apache 2.2.14 on Linux Ubuntu 10.04.1, PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch, and MySQL 5.0.67. The IP address of the victim server is 192.168.60.160.

B. Conducted attacks

We conducted three types of attacks on the WackoPicko, including:

- SQL Injection on the login form;
- Stored XSS on guestbook page; and
- Remote file inclusion with null byte injection.

SQL Injection on the login form is done by injecting a known string **1'or 1=1#**, which allow us to login without a password as Sample User.

On the guestbook page visitors can leave comments to every picture. These comment fields are not properly escaped, so we can use them for stored XSS attack. We insert the following script **<script>location=" http://www.ugd.edu.mk/ index.php/mk;"</script>** in the comment on the picture ugd.jpg (we uploaded this picture previously, and it is saved as 4 images on the server: ugd.jpg, ugd.128.jpg, ugd.128_128.jpg and ugd.550.jpg). Whenever user visits the guestbook page and try to see the targeted picture in full size, the attack is triggered and the JavaScript code will be executed, redirecting the visitor to the link www.ugd.edu.mk.

From WackoPicko, the admin page is vulnerable to file inclusion. We use remote file inclusion in two ways. In the first way, using the browser, we upload two public accessible shells b374k-shell.php [9] and c99shell.php [10] and one picture ugd.jpg, using the form for uploading pictures (through uploading, name of the scripts is changed to b374 and c99). We can run the shells using admin page and null byte injection, for example *http://192.168.60.160/WackoPicko/admin/index.php?page=http://192.168.60.160/WackoPicko/upload/b374/b374%00.php*. Using this shell, we have removed the folder Cart (**rm -rf cart**) and we have created new folder Natasa (**mkdir Natasa**). Using the shell c99.php, we have changed the page http://192.168.60.160/WackoPicko/test.php, by adding a new link to the page http://www.w3schools.com. We chose two different shells, because b374k-shell.php sends commands to the server as part of the POST request body, and c99shell.php sends commands as parameters in the URL.

The other way uses the known hacker tool Metasploit, (from Backtrack 5 R3) and its Reverse TCP Payload command, for generating the reverse shell payload /root/payload.php on the attacker's machine. The procedure for uploading and using this shell on the victim machine is the same as previous (saved with name pay). Using this shell, we have deleted the file error.php (rm error.php) and uploaded the file hack.html (upload /root/hack.html).

Usually, attackers use IP spoofing, but in this case we are not doing that, because we want to see the artifacts left on the attacker's machine, by performing some web attacks.

III. FORENSIC ANALYSIS

In forensics expertise there are three main phases: acquisition, analysis, presentation.

In the acquisition phase, the state of digital system, with all allocated and unallocated areas, is saved for later analysis. This copy is called an image. For preserving integrity of the image, the hash result of the image is calculated and saved. For making the image we used the free AccessData FTK Imager 3.1 [11].

The analysis phase takes the acquired data and involves examination of every piece of data from the evidence. We are starting with hypothesis for what kind of attack can be found. This phase includes search with keywords (names of files or folder that we assume that are produced from the attack). For analysis we are using two forensic tools - Autopsy (The Sleuth Kit, freeware) [12], and WinHex (freeware) [13].

In the last presentation phase, conclusions are made from the analysis and it is necessary to prepare a documentation that can be in a readable format for people who work in courts. This document is called forensic report. Clearly, the final part with forensic report is not done in this research.

A. Analysis of the victim server

The analysis of the victim image is done with the following forensic techniques:

- File system analysis
- Recovering of deleted files and folders
- Log file analysis
- Keyword search
- Overview and keyword search of the swap area

In terms of the file system analysis and recovering of deleted data, we can find that several new php scripts are present in the system, and that some files are recently changed. We can use their names in keyword search. We have examined the log files: error.log, access.log, mysql.log, and their versions with different extensions, for example, log.1, or log.1.gz. Again, we assume that we are dealing with careless attacker that did not change the log files. From the log files analysis, we can first determine that the attack on this server is performed from the IP address 192.168.60.159 and it runs on 32 bit version of Linux operating system with Mozilla Firefox 21.0 web browser. Additionally, we obtained traces that show the use of b374 and c99 shells, and even the modification of the file test.php with c99 shell (see Table I) from 192.168.60.159. In mysql.log file one can find the following two traces

*140904 14:59:32 565 Query SELECT * from `users` where `login` like '-1'or 1=1#' and `password` = SHA1(CONCAT('', `salt`)) limit 1*

605 Query INSERT INTO `comments` (`id`, `text`, `user_id`, `picture_id`, `created_on`) VALUES (NULL,'<script> location=\"http://www.ugd.edu.mk/index.php.mk;\"</script>', '12', '17', '2014-09-03 06:35:16')

The first one, show us how the attacker's SQL query is processed and the second tells us the creation date and time for stored XSS attack, and user_id of the user which created it.

From the log files found in the location presented before, a keyword list can be prepared. In our research, the keyword list is consisted of the following terms: %00.php, natasa, b374, hack.html, c99, www.ugd.edu.mk, hack.php, test.php, Sample User, pay, ugd.550.jpg, error.php, Cart, and href="http://www.w3schools.com/".

The useful results from the keyword search with WinHex were the same as those found in log files.

An enormous interest was expected to be the swap file (which had 400MB of size) and inside that file were not found any hits, which could be further treated as clues/traces for the interest in the analysis. Or in other words, the SWAP area has proven as a useless target for the task.

B. Analysis of the attacker's machine

The analysis of the suspect's image is done with the following forensic techniques using Autopsy:

TABLE I. PART OF A KEYWORD SEARCH ON THE VICTIM MACHINE WITH COMMENTS, PRESENTED ALSO AS ENTRIES IN ACCESS.LOG

No.	Search hints	Name	Path	Modified	Accessed	Inode Modification	Comment
1	[192.168.60.159 - - [03/Sep/2014:07:19:26 -0400] "GET /WackoPicko/admin/index.php?page=http://192.168.60.160/WackoPicko/upload/b374/b374%00.php HTTP/1.1" 200 1773 "-" "Mozilla/5.0 (X11; Linux i686 on x86_64; rv:14.0) Gecko/20100101 Firefox/14.0.1"	access.log	\var\log\apache2	03/09/2014 16:22:35 +2	03/09/2014 12:35:24 +2	03/09/2014 16:22:35 +2	Successful use of the script b374.php with null byte injection and GET request from IP address 192.168.60.159
2	192.168.60.159 - - [03/Sep/2014:07:19:46 -0400] "POST /WackoPicko/admin/index.php?page=http://192.168.60.160/WackoPicko/upload/b374/b374%00.php& HTTP/1.1" 200 1792 "http://192.168.60.160/WackoPicko/admin/index.php?page=http://192.168.60.160/WackoPicko/upload/b374/b374%00.php" "Mozilla/5.0 (X11; Linux i686 on x86_64; rv:14.0) Gecko/20100101 Firefox/14.0.1"	access.log	\var\log\apache2	03/09/2014 16:22:35 +2	03/09/2014 12:35:24 +2	03/09/2014 16:22:35 +2	Successful use of the script b374.php with null byte injection and POST request from IP address 192.168.60.159
3	192.168.60.159 - - [03/Sep/2014:08:40:45 -0400] "POST /WackoPicko/admin/index.php?page=http%3A%2F%2F192.168.60.160%2FWackoPicko%2Fupload%2Fc99%2Fc99%00.php&act=f&f=test.php&ft=edit&d=%2Fowaspbwa%2FWackoPicko-relative_urls-git%2Fwebsite%2F HTTP/1.1" 200 3595 "http://192.168.60.160/WackoPicko/admin/index.php?page=http%3A%2F%2F192.168.60.160%2FWackoPicko%2Fupload%2Fc99%2F[redacted]%00.php&act=f&f=test.php&ft=edit&d=%2Fowaspbwa%2FWackoPicko-relative_urls-git%2Fwebsite%2F" "Mozilla/5.0 (X11; Linux i686 on x86_64; rv:14.0) Gecko/20100101 Firefox/14.0.1"	access.log	\var\log\apache2	03/09/2014 16:22:35 +2	03/09/2014 12:35:24 +2	03/09/2014 16:22:35 +2	Successful use of the script c99.php with null byte injection and POST request from IP address 192.168.60.159. From the URL, one can see that file test.php is edited (ft=edit).
4	192.168.60.159 - - [03/Sep/2014:10:01:12 -0400] "GET /WackoPicko/admin/index.php?page=http://192.168.60.160/WackoPicko/upload/pay/pay%00 HTTP/1.1" 200 26 "-" "Mozilla/5.0 (X11; Linux i686 on x86_64; rv:14.0) Gecko/20100101 Firefox/14.0.1"	access.log	\var\log\apache2	03/09/2014 16:22:35 +2	03/09/2014 12:35:24 +2	03/09/2014 16:22:35 +2	Successful use of the script pay with null byte injection and GET request from IP address 192.168.60.159

TABLE II. SOME RESULTS OF THE FILE SYSTEM ANALYSIS AND HASH COMPARING USING FTK IMAGER AND AUTOPSY

Victim machine file system		Attacker machine file system	
Location	**MD5 Hash**	**Location**	**MD5 Hash**
\[root]\owaspbwa\WackoPicko-relative_urls-git\website\hack.html	c12d9b7961e6dccf246eecfd50b4d165	root/hack.html	c12d9b7961e6dccf246eecfd50b4d165
\[root]\owaspbwa\WackoPicko-relative_urls-git\website\upload\c99\c99	83f83bb415b98d41fbcae9193b47c984	root/Desktop/c99shell.php	83f83bb415b98d41fbcae9193b47c984
\[root]\owaspbwa\WackoPicko-relative_urls-git\website\upload\pay\pay	946d436048ca2e3c2f271c2b035a7774	root/payload.php	946d436048ca2e3c2f271c2b035a7774
\[root]\owaspbwa\WackoPicko-relative_urls-git\website\upload\ugd\ugd	d5105a4fd856aa1eee786bc10fd35898	root/Desktop/ugd.jpeg	d5105a4fd856aa1eee786bc10fd35898
\[root]\owaspbwa\WackoPicko-relative_urls-git\website\upload\ugd\ugd.550.jpg	8b2ca7498c6f2dc140695e9031211fca	root/.mozilla/firefox/nq474mcm.default/Cache/8/9E/769E1d01	8b2ca7498c6f2dc140695e9031211fca
\[root]\owaspbwa\WackoPicko-relative_urls-git\website\upload\ugd\ugd.128.jpg	bfcb3fc8420481c5be8fd9609291ef56	root/.mozilla/firefox/nq474mcm.default/Cache/5/10/43DABd0	bfcb3fc8420481c5be8fd9609291ef56
\[root]\owaspbwa\WackoPicko-relative_urls-git\website\upload\b374\b374	cc8d0f697435783610a4c17278e3c51c	root/.mozilla/firefox/nq474mcm.default/Cache/1/64/268A0d01	cc8d0f697435783610a4c17278e3c51c

TABLE III. SOME RESULTS FROM LOG ANALYSIS AND INTERNET HISTORY FILES FROM ATTACKER'S MACHINE WITH AUTOPSY

Content	Program	Source File	Date Accessed	Tags	Comment
http://192.168.60.160/WackoPicko/admin/index.php?page=http://192.168.60.160/WackoPicko/upload/b374/b374%00.php	FireFox	root/.mozilla/firefox/nq474mcm.default/places.sqlite	2014/09/03 07:19:20	Mozilla History	Using of the script b374.php on IP address 192.168.60.160.
http://192.168.60.160/WackoPicko/admin/index.php?page=http%3A%2F%2F192.168.60.160%2FWackoPicko%2Fupload%2Fc99%2Fc99%00.php&act=f&f=test.php&ft=edit&d=%2Fowaspbwa%2FWackoPicko-relative_urls-git%2Fwebsite%2F	FireFox	root/.mozilla/firefox/nq474mcm.default/places.sqlite	2014/09/03 08:40:40	Mozilla History	Using of the script c99.php with null byte injection on IP address 192.168.60.160. From the URL, one can see that file test.php is edited (ft=edit).
Name> b374_included URL> 192.168.60.160	FireFox	root/.mozilla/firefox/nq474mcm.default/cookies.sqlite			This is a sample artifact found at the browser's cache folder which represents a cookie from the attacked site with already applied the intrusion script b374

- File system analysis and hash comparing
- Log file analysis and internet history files
- OS and third party software artifacts/CLI history overview
- Keyword search
- Preparation of forensic report

According to the report of the incident and the previous analysis on the server, the forensic expert can have a starting point for answering the question - Is the suspect the attacker itself? We actually have the attacker's machine, and the following results are found:

File system analysis and hash comparing

In this case, a complete file search activity is done in which every file and folder is hashed. From the attacked side, the forensic expert can find names and hash values of the files which one thinks that are produced from the incident, and see if some of those values can be found in the suspect's image. If so, there is a starting point for determining that this is the actual attacker. In this case, we have found that the file content and their hash values are appropriate for both sides, so we have a starting point. From Table II, one can see that the shells used in the attack, are the same files on both machines, and also

uploaded files ugd.jpg and hack.html are the same. Additionally, one can see that created versions of the picture ugd.jpg on the victim machine can be found in the cache of the web browser Mozilla Firefox on attacker machine. This type of forensic analysis is proven to be very useful.

We can conclude that careless attacker that saves and use shells for an attack, lives evidences for their existence on both machines.

Log file analysis and internet history files

The mentioned activity in general is focused on locating internet artifacts which can be found in web browser temporary storage and browser's history files. In this case, several URL records and cached content is found, which proves that the user was visiting the attacked site. Since our attacking methods are conducted through the interface of the web browser, this analysis can prove the act of the third attack, the methodology (used scripts) and the files that were the object of the intrusion. Sample results are given in Table III.

Because web browser is the primary way of performing selected attacks, it is normal that we have found evidence in Internet history, again, if the attacker is careless.

OS and third party software artifacts

A useful forensic artifacts are found in the bash history file (Table IV). Apparently the forensic expert can conclude that the user has cleaned the other history of the shell activity, or the shell was not used for other attacks. Also, additional interesting forensic artifact can be the IP address of the suspect's machine, but in this case, the operating system of the suspect machine was using an automatic assigned IP address from the virtual machine. Therefore, the file at the location etc/network/interfaces in this case does not prove anything.

If the attacker had used the command prompt for carrying the attack, the bash_history file keeps the records of executed commands (if it is not erased by the attacker). So, for example, if the attacker uses the Metasploit console, every command will be recorded in the bash_history file.

TABLE IV. INTERESTED RESULT FROM OS ARTIFACT ANALYSIS

File name	Location	Content
.bash_history	root/ bash_history	msfpayload php/meterpreter/reverse_tcp LHOST=192.168.60.159 LPORT=4444 R >/root/payload.php

Keyword search

From the keyword search approach we can obtain additional information, some of them are presented in Table V (for keyword hack.html).

IV. DISCUSSIONS

From the obtained results, we can summarized several results. First, remote file inclusion and use of shells leave many traces on the attacker's and the victim's machine. If the shell carries out commands through a POST request, its successful use is documented (without content of POST body) in victim log files and in the attacker files places.sqlite and possible in the cookies.sqlite. Even more, if the shell carries out commands as parameters in the URL, they are documented in the same way, and even the attack can be reconstructed correctly. If the attacker use Metasploit console, he leaves traces in a batch_history file on its machine.

TABLE V. SOME RESULTS FROM KEYWORD SEARCH ANALYSIS ON ATTACKER'S MACHINE

File location	Preview
/img_image.E01/opt/metasploit/postgresql/data/pg_xlog/000000010000000000000002	/root/«hack.html« -> /root/«hack.html« session_uploady
/img_image.E01/root/.mozilla/firefox/nq474mcm.default/places.sqlite	160/WackoPicko/«hack.html« http://192.168.60.160/WackoPicko/u
/img_image.E01/root/.mozilla/firefox/nq474mcm.default/sessionstore-1.js	160/WackoPicko/«hack.html«","ID":20,"docshellID":8,"docIdentifier":20
/img_image.E01/opt/metasploit/postgresql/data/pg_xlog/xlogtemp.3440	/root/«hack.html« -> /root/«hack.html« session_uploady

Examples of our SQL injection and stored XSS attacks use the body of the POST request, so they do not leave any traces in the victim log files nor the browser history files. The script from the stored XSS attack is saved in the backend database and it leaves traces in the mysql.log file, which includes a time stamp and user_id which identifies the user who create this entry. Also SQL injection login leaves traces in the mysql.log file. It is difficult to connect the traces from mysql.log file with the attacker machine.

V. CONCLUSIONS

Using post-mortem computer forensic analysis of attacker and victim machine, we found several artifacts on them, for scenario of three types of attacks: SQL Injection, stored XSS, and remote file inclusion with a null byte injection. Careless attacker that saves and use shells for his attack, leaves evidence on both machines. On the attacker's machine, traces were found in the browser's history files, browser' temporary storage, and bash_history_file. On the victim's machine, traces were found in the file system and in the log files. These artifacts can help to identify and sometimes to reconstruct performed attacks, and even more, they can represent a valid evidence for the court.

REFERENCES

[1] Symantec, "Internet Security Threat Report 2014", April 2014.

[2] Open Web Application Security Project, "OWASP Top Ten Project" [Online]. Available: http://www.owasp.org/index.php/Category: OWASP Top Ten Project. (Access Date: 1 September 2014)

[3] WackoPicko [Online]. Available: https://github.com/adamdoupe/WackoPicko/archive/master.zip.. (Access Date: 1 September 2014)

[4] A. Doupe, M. Cova and G. Vigna, "Why Johnny Can't Pentest: An Analysis of Black-box Web Vulnerability Scanners". In C. Kreibich, M. Jahne (Eds.) *Proceedings of the 7th International conference on Detection of Intrusions and Malware, and Vulnerability Assessment - DIMVA'10*, pp. 111-131, Springer Berlin Heidelberg 2010.

[5] J. J. B. Andrade, and D. Gan, "A Forensics Investigation into Attacks on Linux Servers", University of East London, Cybercrime, Cybercrime, Security and Digital Forensics Conference, University of East London, May 14-15, 2012

[6] J. M. Urrea, "An Analysis of Linux RAM forensics", MSc thesis, Naval Postgraduate School, Monterey, USA, March 2006

[7] H. Shulman, and M. Waidner, "Towards Forensic Analysis of Attacks with DNSSEC", 2014 IEEE Security and Privacy Workshops: International Workshop on Cyber Crime, May 2014

[8] Open Web Application Security Project (OWASP), OWASP Broken Web Applications Project. [Online]. Available: https://www.owasp.org/index.php/OWASP_Broken_Web_Applications_Project#tab=Main

[9] b374k-shell.php. [Online]. Available: https://code.google.com/p/b374k-shell/. (Access Date: 1 September 2014)

[10] c99shell.php. [Online]. Available: http://www.4shared.com/file/-sHx3aFm/c99shell.html?locale=en. (Access Date: 1 September 2014)

[11] FTK Imager 3.1. [Online]. Available: http://accessdata-ftk-imager.software.informer.com/3.1/. (Access Date: 1 September 2014)

[12] Autopsy 3.1.0. [Online]. Available: http://www.sleuthkit.org/autopsy/. (Access Date: 1 September 2014)

[13] WinHex. [Online]. Available: http://www.x-ways.net/winhex/. (Access Date: 1 September 2014)