Department of Electrical and Computer Engineering

"Master of Engineering in Telecommunications & Information Security (MTIS)"

ECE 567 – "ADVANCED NETWORK SECURITY"

- **Project Report: Part 2 - Defense Strategies**

• **Name:** Nelson Gomez	• **Name:** Gabriel Naranjo
• **Student No.** V01072284	• **Student No.** V01052227
• **Email:** nelsongb@uvic.ca	• **Email:** gabrielnaranjoor@uvic.ca

April 04, 2025

➢ **PART 2: DEFENSE STRATEGIES**

Phase 1: Intrusion Detection (11%)

1.1 Explain briefly the generic attack scenario associated with the selected vulnerability (2 paragraphs maximum); a graphical sketch (in addition of the explanations) is required. The attack scenario graph should allow a lay person to understand how the vulnerability works without executing a specific exploit code (2.5%).

Based on the Nessus report and the CVSS scores, a specific group of vulnerabilities was selected to identify the best option for this part of the project. The vulnerability 11356 - NFS Exported Share Information Disclosure is marked as exploitable by Metasploit, and is associated with CVE-1999-0170, CVE-1999-0211, and CVE-1999-0554. After searching for these CVE IDs in Metasploit, it was confirmed that CVE-1999-0170 can be exploited using the module auxiliary/scanner/nfs/nfsmount.

The first step is to understand the basic concept of NFS. NFS (Network File System) is a protocol that allows two devices to share files or directories over a network. In this case, the target device named UranusZ allows access to certain files without any user validation or authentication.

An attacker or intruder can discover the names of files or directories that are exposed and vulnerable. In some cases, it may even be possible to copy or modify those files.

- **Attack Scenario Graph**

The attack scenario is described by the following graphical sketch:

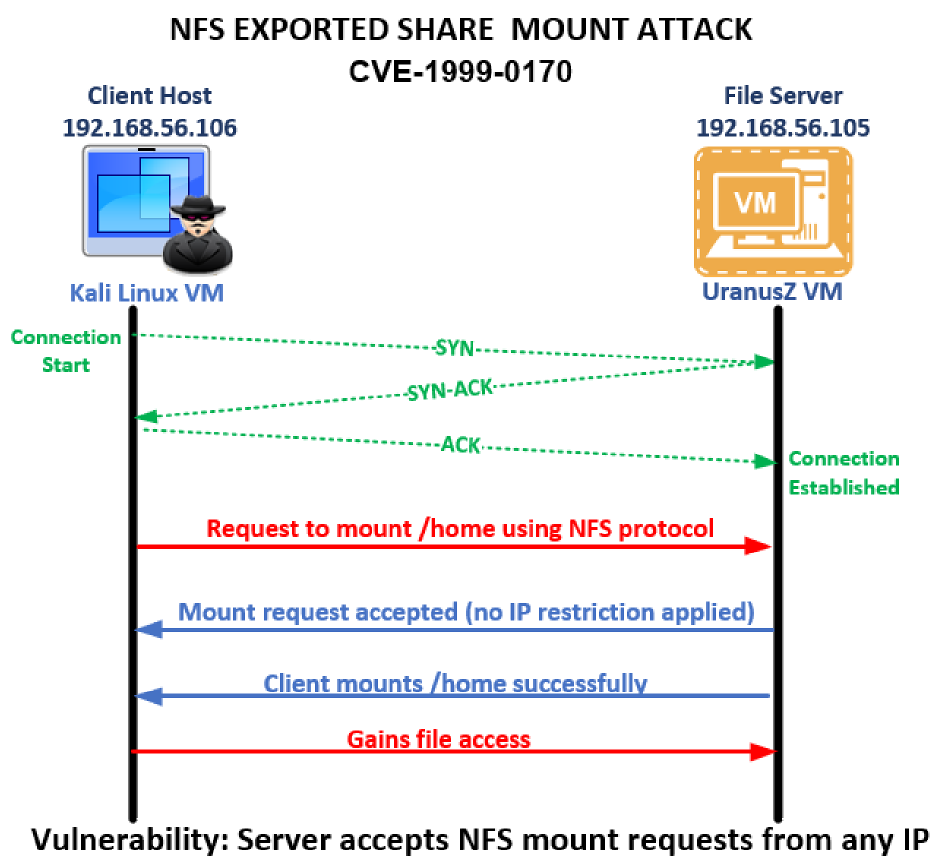


Figure. Graphical Sketch: NFS Unauthenticated Mount Attack

This attack exploits a misconfigured NFS server that exports the /home directory without IP-based access controls. As illustrated in the diagram, the attacker (client) establishes a standard TCP connection and transmits a mount request via the NFS protocol. Due to the server's default acceptance of requests from any IP address, the attacker successfully mounts the shared directory. This grants the attacker unrestricted, unauthenticated access to the directory's contents, creating a significant security risk by exposing potentially sensitive files.

Exploitation Process

1. Using the auxiliary/scanner/nfs/nfsmount module available in Metasploit, the following messages were exchanged between the attacker and the victim.

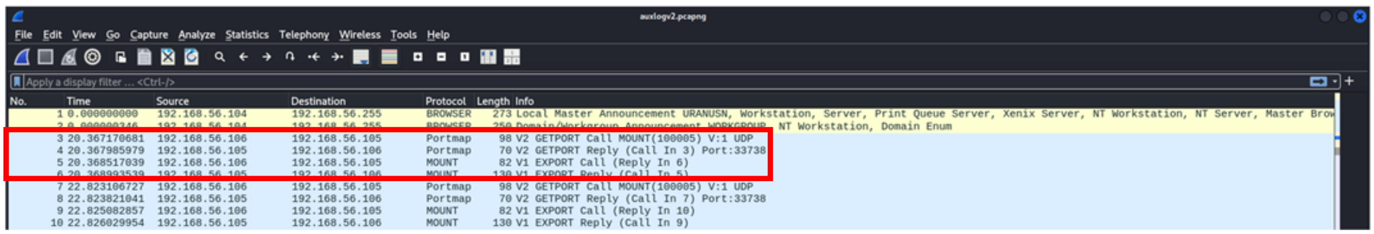


Figure. Message Flow auxiliary/scanner/nfs/nfsmount.

In this phase, the attacker's host sends a GETPORT request to the victim to ask which port the mount service is using. The victim replies with the information: Port 33738. After that, the attacker sends an EXPORT request to ask which volumes (directories) are available to be mounted. The victim responds with the following information:

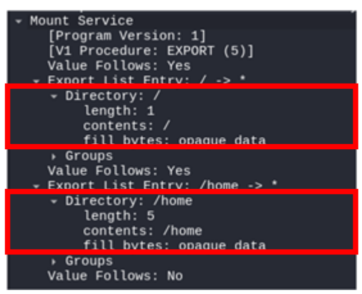
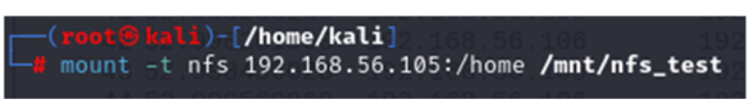


Figure. Volume information replied from Victim Host.

The same information can also be obtained using the command **showmount -e 192.168.56.105**. The main difference between using Metasploit and the direct command lies in the protocol: the first uses UDP, while the latter uses TCP. With these details, the attacker completes the reconnaissance phase, having collected enough information to proceed with the attack. The next step is to execute the attack using the mount command, which is explained in the following section.

2. Executing the mount command (exploit phase):



The following exchange takes place between the attacker and the victim: NULL CALL and NULL REPLY confirm basic connectivity between both hosts. EXCHANGE_ID is used to define the identity of each device in the NFS communication. CREATE_SESSION, RECLAIM_COMPLETE, and SECINFO_NO_NAME are involved in establishing and configuring the NFS session.

PUTROOTFH and GETATTR are used to access the root directory and request its attributes. Afterwards, ACCESS and GETATTR are used again to access the /home directory and retrieve its attributes.

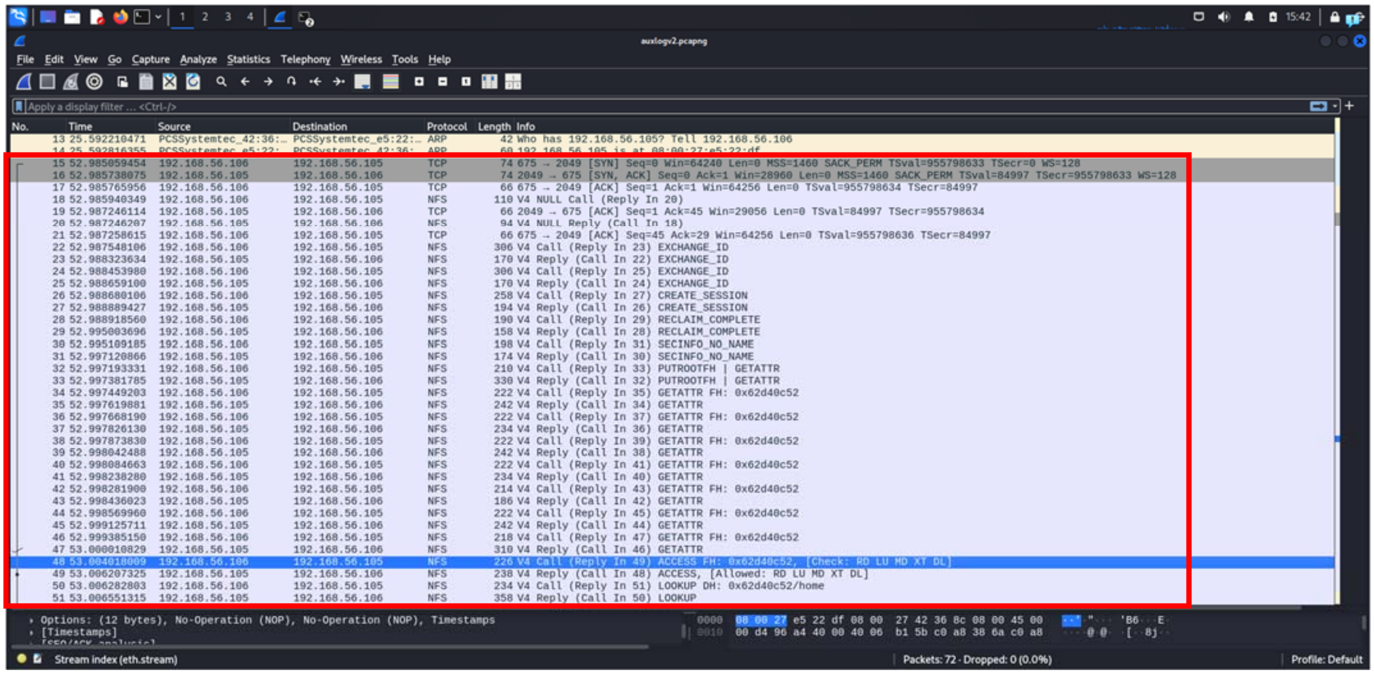


Figure. Mount messages flow between attacker and victim.

1.2 Define new Snort rules (as many as you think are necessary) to detect the attack and add these rules to the snort rule set. Justify the rationale for the rules. Make sure your Snort rules do not over-fit the attack scenario (5%).

Based on the Wireshark traces, the following rule was designed. The HOME_NET variable was configured to include the IP addresses of UranusR, UranusZ, and UranusN. All other IP addresses are considered external traffic.

- alert udp !$HOME_NET any -> 192.168.56.105 111 (content: “|00 01 86 a5|”; msg:“RPC and MOUNT detected from Outside by UDP”;sid:10001;)

  This rule detects the use of the auxiliary/scanner/nfs/nfsmount module in **Metasploit** when executed from a host outside the LAN. Traffic from the defined LAN hosts does not trigger the alert, as it is considered trusted. Port 111 is associated with this reconnaissance stage, and the content value |00 01 86 a5| is used to identify the use of the MOUNT protocol:

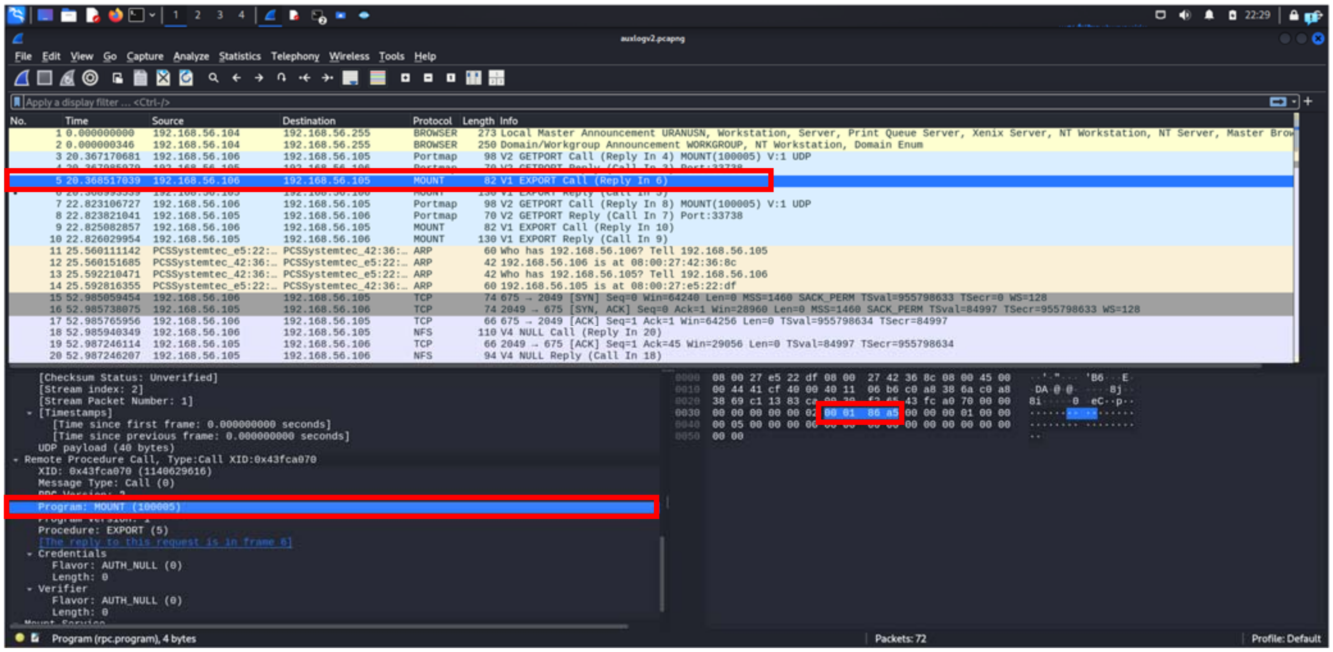


Figure. Content value to identify MOUNT.

- alert tcp !$HOME_NET any -> 192.168.56.105 111 (content: “|00 01 86 a0|”; msg:“RPC and PORTMAP detected from Outside by TCP”;sid:10002;)

  This alert detects the use of the **showmount** command from a host outside the LAN network. Traffic from internal LAN hosts does not trigger the alert, as it is considered trusted. Port 111 is associated with this reconnaissance stage, and the content value |00 01 86 a0| helps identify the use of the PORTMAP protocol.

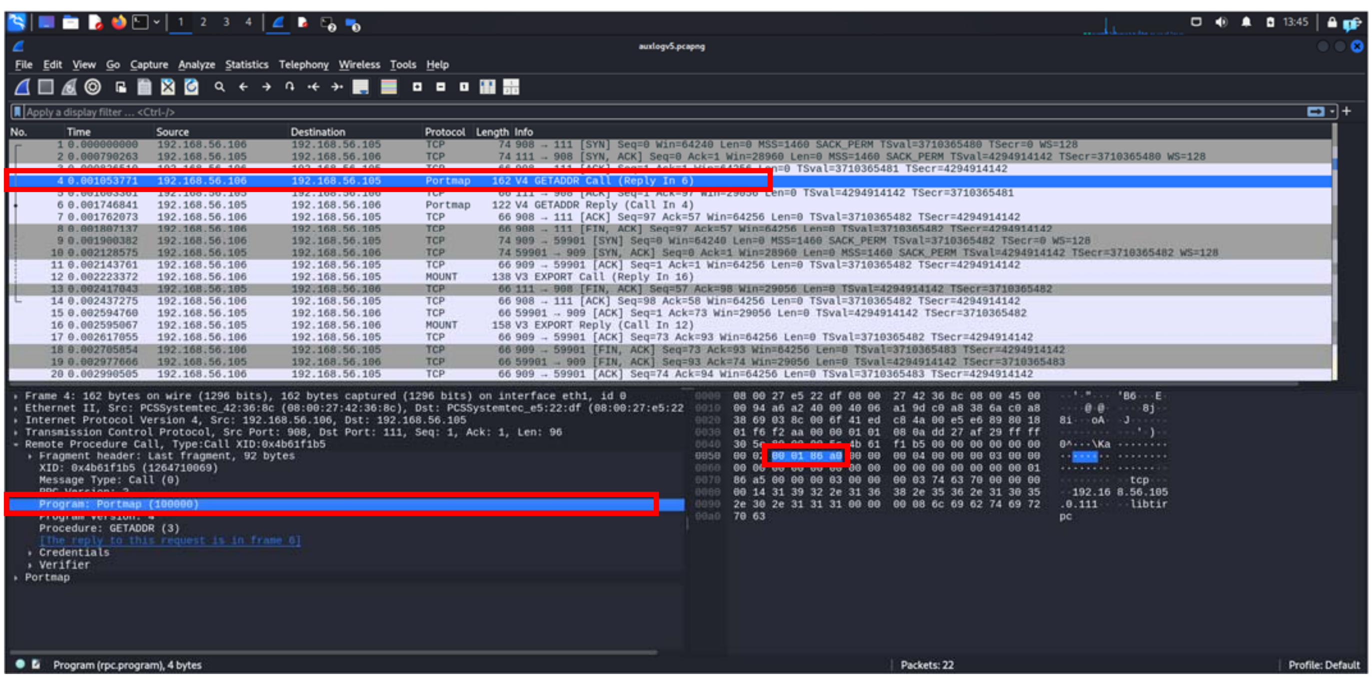


Figure. Content value to identify PORTMAP.

* alert tcp !$HOME_NET any -> 192.168.56.105 2049 (content:"|00 01 86 a3|"; msg:"NFS detected from Outside by TCP";sid:10003;)

  This alert detects the use of the **mount -t nfs** command from a host outside the LAN network. Traffic from internal LAN hosts does not trigger the alert, as it is considered trusted. Port 2049 is associated with this reconnaissance stage, and the content value |00 01 86 a3| helps identify the use of the NFS protocol.

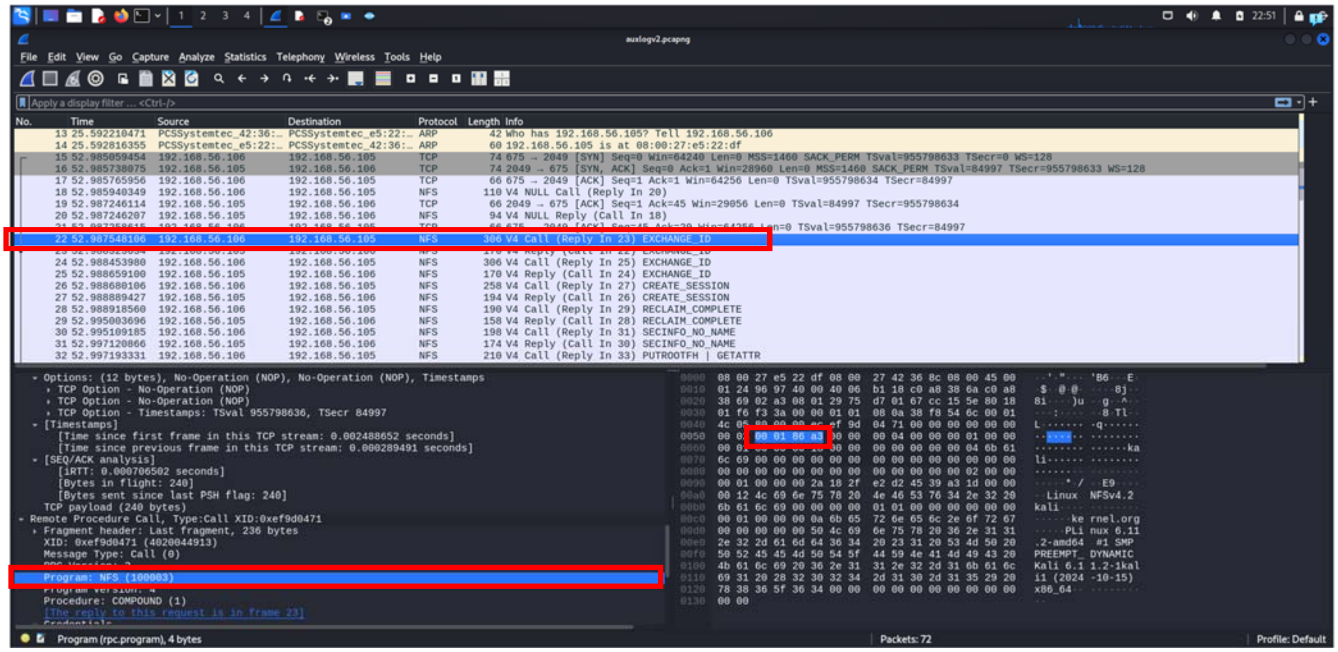


Figure. Content value to identify NFS.

1.3 Run Snort on uranusR in intrusion detection mode. And then execute using Kali the selected attack against your selected target machine (1%).

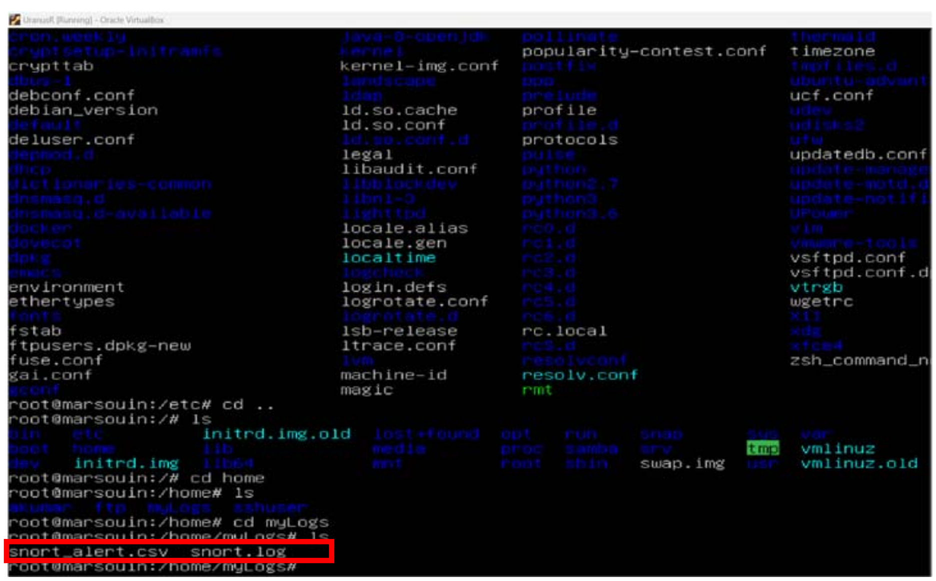


Figure. Snort execution evidence on the UranusR host.

Using root privileges, we transferred the snort_alert.csv file to Kali. Then, by using a shared folder with the personal laptop, we accessed the results. The output of Snort logs is as follows:

04/01-17:30:08.715253	1	10001	0	RPC and MOUNT detected from outside by UDP	UDP	192.168.56.106	42329	192.168.56.105	111	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x62
04/01-17:30:08.715253	1	579	8	RPC portmap mountd request UDP	UDP	192.168.56.106	42329	192.168.56.105	111	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x62
04/01-17:30:08.716960	1	1924	6	RPC mountd UDP export request	UDP	192.168.56.106	36945	192.168.56.105	43685	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x52
04/01-17:30:24.563320	1	10002	0	RPC and PORTMAP detected from outside by TCP	TCP	192.168.56.106	656	192.168.56.105	111	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xA2
04/01-17:30:24.563320	1	1266	10	RPC portmap mountd request TCP	TCP	192.168.56.106	656	192.168.56.105	111	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xA2
04/01-17:30:24.564221	1	574	8	RPC mountd TCP export request	TCP	192.168.56.106	657	192.168.56.105	56079	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x8A
04/01-17:30:34.993267	1	579	8	RPC portmap mountd request UDP	UDP	192.168.56.104	621	192.168.56.105	111	08:00:27:1B:2F:3F	08:00:27:E5:22:DF	0x62
04/01-17:30:34.995849	1	574	8	RPC mountd TCP export request	TCP	192.168.56.104	621	192.168.56.105	56079	08:00:27:1B:2F:3F	08:00:27:E5:22:DF	0x8E
04/01-17:30:49.824221	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x6E
04/01-17:30:49.825016	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x132
04/01-17:30:49.825378	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x132
04/01-17:30:49.825560	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0x102
04/01-17:30:49.825902	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xBE
04/01-17:30:49.830413	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xC6
04/01-17:30:49.830593	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xD2
04/01-17:30:49.830821	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xDE
04/01-17:30:49.831013	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xDE
04/01-17:30:49.831243	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xDE
04/01-17:30:49.831419	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xDE
04/01-17:30:49.831598	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xD6
04/01-17:30:49.831814	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xDE
04/01-17:30:49.833074	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xDA
04/01-17:30:49.835524	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xE2
04/01-17:30:49.836537	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xEA
04/01-17:30:49.836823	1	10003	0	NFS detected from outside	TCP	192.168.56.106	846	192.168.56.105	2049	08:00:27:42:36:8C	08:00:27:E5:22:DF	0xEA

1.4 Analyze the Snort alerts log generated after the attack and discuss the result in terms of false positives and false negatives (in principle the snort configuration must successfully alerts on all suspicious packets, while not raising alerts on legitimate traffic) (2.5%).

The Snort alert log shows all the stages involved in executing the attack. At the same time, we executed the showmount and mount commands from UranusN. That traffic is highlighted in orange, but it did not trigger any alerts, as it is considered trusted traffic within the LAN network.

Alerts 10001 and 10002 correspond to the reconnaissance stage, during which the Kali host (192.168.56.106) scanned UranusZ (192.168.56.105). Alert 10003 is related to the exploitation phase, where the mount command was used from the Kali host to UranusZ. This traffic is highlighted in green. These alerts are considered true positives, as they correctly identify a real attack attempt.

Phase 2: Intrusion Prevention - using Firewall (8%)

2.1 Define the IPTables rules to prevent the attack selected in Phase 1 and provide rationale for each of the rules. You should minimize false negatives and false positives so that a legitimate client is allowed access, but a client that attempts the selected attack is blocked (4%).

Solution:

2.1.1 IPTables Rules definition

To mitigate the NFS mount attack associated with CVE-1999-0170, we implemented a firewall rule set directly on the target machine (uranusZ – 192.168.56.105). This host exports a vulnerable NFS share (/home) and was previously accessible to any client due to an insecure wildcard export (*) in the NFS configuration.
The firewall configuration was designed with a principle of least privilege, allowing access only to the ports involved in the attack vector (111 for Portmapper and 2049 for NFS) and only from trusted internal IPs. This minimizes exposure while preserving system functionality for legitimate clients.

In this configuration, the allowed hosts are:

- uranusN (192.168.56.104) – authorized internal NFS client.
- uranusR (192.168.56.151) – the IDS host, which may require passive access or monitoring capabilities.

All other sources, including the attacker (Kali – 192.168.56.106), are denied access to these ports. This service-level filtering avoids the common mistake of blocking ports for all hosts, as advised in the course guidelines.

The IPTables rules applied on uranusZ are as follows:

IPTables Rules	Description
iptables -A INPUT -s 192.168.56.104 -p tcp --dport 2049 -j ACCEPT	Allow NFS (TCP) from uranusN
iptables -A INPUT -s 192.168.56.104 -p udp --dport 2049 -j ACCEPT	Allow NFS (UDP) from uranusN
iptables -A INPUT -s 192.168.56.104 -p tcp --dport 111 -j ACCEPT	Allow Portmapper (TCP) from uranusN
iptables -A INPUT -s 192.168.56.104 -p udp --dport 111 -j ACCEPT	Allow Portmapper (UDP) from uranusN
iptables -A INPUT -s 192.168.56.151 -p tcp --dport 2049 -j ACCEPT	Allow NFS (TCP) from uranusR
iptables -A INPUT -s 192.168.56.151 -p udp --dport 2049 -j ACCEPT	Allow NFS (UDP) from uranusR
iptables -A INPUT -s 192.168.56.151 -p tcp --dport 111 -j ACCEPT	Allow Portmapper (TCP) from uranusR
iptables -A INPUT -s 192.168.56.151 -p udp --dport 111 -j ACCEPT	Allow Portmapper (UDP) from uranusR
iptables -A INPUT -p tcp --dport 2049 -j DROP	Deny NFS (TCP) from all others
iptables -A INPUT -p udp --dport 2049 -j DROP	Deny NFS (UDP) from all others
iptables -A INPUT -p tcp --dport 111 -j DROP	Deny Portmapper (TCP) from all others
iptables -A INPUT -p udp --dport 111 -j DROP	Deny Portmapper (UDP) from all others

Table. IPTables rules applied on uranusZ

2.1.2 Justification of the rules to prevent CVE-1999-0170 (NFS Misconfiguration Attack):

- **For minimizing False Positives:**

  The firewall rules were designed to ensure that only NFS-related traffic from authorized internal clients (uranusN and uranusR) is allowed. By filtering traffic based on both service port and source

IP, the policy avoids mistakenly blocking legitimate access to the NFS service. This prevents service disruption for trusted clients and ensures that only unauthorized hosts are denied access, so effectively minimizing false positives.

- **For minimizing False Negatives:**

The firewall configuration explicitly blocks all traffic trying to access the NFS (port 2049) and Portmapper (port 111) services from untrusted sources. By using DROP rules for these ports and allowing only specific trusted IP addresses, the system avoids missing or ignoring unwanted connections. This configuration makes sure that no unauthorized host can access the NFS share, reducing the chance of false negatives and stopping the attack.

- **Service-Focused Filtering:**

To avoid blocking unrelated services, the firewall rules focus only on the specific ports used in the attack (2049 for NFS and 111 for Portmapper). This service-based filtering preserves system functionality while securing vulnerable services.

Additionally, while Snort runs in passive mode on uranusR to detect suspicious traffic, IPTables on uranusZ actively blocks unauthorized access. This clear separation between detection and prevention ensures better control and improves the overall security of the system.

Note: CVE-1999-0170 is caused by a misconfiguration in the NFS server, which allows unauthenticated access to NFS exports. Both legitimate users and attackers use the same protocol and send identical mount requests, making it difficult to differentiate between them based on packet content. Therefore, content-based filtering is ineffective because it cannot distinguish between legitimate and malicious traffic. The most effective solution in this scenario is to use IP whitelisting to restrict access to trusted internal machines. This approach ensures that only authorized users can access the NFS service, while blocking all unauthorized access.
In this context, whitelisting is the most reliable and efficient method to secure the NFS server, as it directly addresses the root cause of the vulnerability: unrestricted access.

2.2 Test the firewall rules and provide screenshots documenting the results (2%).

Solution:
The rules were configured on UranusZ as follows:

UranusZ 1 [Running] - Oracle VM VirtualBox

```
root@UranusZ:~# iptables -A INPUT -s 192.168.56.104 -p tcp --dport 2049 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -s 192.168.56.104 -p udp --dport 2049 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -s 192.168.56.104 -p tcp --dport 111 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -s 192.168.56.104 -p udp --dport 111 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -s 192.168.56.151 -p tcp --dport 2049 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -s 192.168.56.151 -p udp --dport 2049 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -s 192.168.56.151 -p tcp --dport 111 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -s 192.168.56.151 -p udp --dport 111 -j ACCEPT
root@UranusZ:~# iptables -A INPUT -p tcp --dport 2049 -j DROP
root@UranusZ:~# iptables -A INPUT -p udp --dport 2049 -j DROP
root@UranusZ:~# iptables -A INPUT -p tcp --dport 111 -j DROP
root@UranusZ:~# iptables -A INPUT -p udp --dport 111 -j DROP
root@UranusZ:~# _
```

To validate the IPTables rules applied on the target machine (uranusZ – 192.168.56.105), we performed controlled testing from two clients:

1. An unauthorized attacker (Kali – 192.168.56.106), which is not included and should be blocked.
2. A legitimate internal host (uranusN – 192.168.56.104), which is included in the whitelist.

Test 1: Mount attempt from attacker (Kali)

Mount commands

```
┌──(kali㉿kali)-[~]
└─$ showmount -e 192.168.56.105
clnt_create: RPC: Timed out

┌──(kali㉿kali)-[/mnt/nfs_test]
└─$ mount -t nfs 192.168.56.105:/home /mnt/nfs_test
mount.nfs: failed to apply fstab options
```

Port Scanning with NMAP

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.56.105
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-26 22:24 PDT
Nmap scan report for 192.168.56.105
Host is up (0.000071s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE    SERVICE VERSION
22/tcp   open     ssh     OpenSSH 7.4p1 Debian 10+deb9u6 (proto
80/tcp   open     http    nginx 1.15.10
111/tcp  filtered rpcbind
2049/tcp filtered nfs
MAC Address: 08:00:27:03:CC:13 (PCS Systemtechnik/Oracle Virtua
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect result
Nmap done: 1 IP address (1 host up) scanned in 7.55 seconds
```

Result:
The attacker is unable to mount the NFS share. The mount command fails (Time out), confirming that the IPTables rules successfully block access to both Portmapper (111) and NFS (2049) from untrusted IP addresses.

Test 2: Mount attempt from trusted hosts: UranusN and UranusR

UranusN – 192.168.56.104

```
root@UranusN:/# showmount -e 192.168.56.105
Export list for 192.168.56.105:
/     *
/home *
root@UranusN:/# mount -t nfs 192.168.56.105:/home /mnt/nfs_test
root@UranusN:/# ls -ali /mnt/nfs_test/
total 44
 8197 drwxr-xr-x 11 root      root      4096 Oct 24 10:40 .
  103 drwxr-xr-x  3 root      root      4096 Mar 26 20:39 ..
  900 drwxr-xr-x  3 jrousseau jlagasse  4096 Apr  6  2019 aachamy
10524 drwxr-xr-x  3 wwang     lzhang    4096 Nov  3  2020 aalezani
 9744 drwxr-xr-x  2 alarousse jrousseau 4096 Apr  6  2019 akeita
10555 drwxr-xr-x  2 aalezani  wwang     4096 Oct 28  2020 jcampbell
  129 drwxr-xr-x  3 lzhang    snort     4096 Oct 31  2020 jlagasse
10540 drwxr-xr-x  2 akeita    aamusala  4096 Oct 28  2020 rgandhi
11748 drwxr-xr-x  2 rgandhi   akeita    4096 Oct 29  2020 shared-vault
11799 drwxr-xr-x  4 rgandhi   akeita    4096 Oct 24 11:15 svault
10477 drwxr-xr-x  2 aamusala  alarousse 4096 Oct 31  2020 wwang
root@UranusN:/# _
```

UranusR – 192.168.56.151

```
root@marsouin:/# showmount -e 192.168.56.105
Export list for 192.168.56.105:
/     *
/home *
root@marsouin:/# mount -t nfs 192.168.56.105:/home /mnt/nfs_test
root@marsouin:/# cd /mnt/nfs_test
root@marsouin:/mnt/nfs_test# ls -ali
total 44
  8197 drwxr-xr-x 11 root root 4096 Oct 24 17:40 .
260611 drwxr-xr-x  4 root root 4096 Mar 27 04:19 ..
   900 drwxr-xr-x  3 1002 1002 4096 Apr  6  2019 aachamy
 10524 drwxr-xr-x  3 1005 1005 4096 Nov  4  2020 aalezani
  9744 drwxr-xr-x  2 1003 1003 4096 Apr  6  2019 akeita
 10555 drwxr-xr-x  2 1007 1007 4096 Oct 28  2020 jcampbell
   129 drwxr-xr-x  3 1001 1001 4096 Oct 31  2020 jlagasse
 10540 drwxr-xr-x  2 1006 1006 4096 Oct 28  2020 rgandhi
 11748 drwxr-xr-x  2 1008 1008 4096 Oct 29  2020 shared-vault
 11799 drwxr-xr-x  4 1008 1008 4096 Oct 24 18:15 svault
 10477 drwxr-xr-x  2 1004 1004 4096 Oct 31  2020 wwang
```

Result:
The trusted hosts can access and mount the NFS share without issues, showing that access for authorized IPs is preserved.

Test 3: Verifying the Rules on UranusZ

Confirming the IP tables rules were correctly installed and operating:

```
root@Uranus2:~# iptables -L -n -v
Chain INPUT (policy ACCEPT 1158 packets, 79783 bytes)
 pkts bytes target     prot opt in     out     source               destination
  452 43480 ACCEPT     tcp  --  *      *       192.168.56.104       0.0.0.0/0            tcp dpt:204
9
    0     0 ACCEPT     udp  --  *      *       192.168.56.104       0.0.0.0/0            udp dpt:204
9
    0     0 ACCEPT     tcp  --  *      *       192.168.56.104       0.0.0.0/0            tcp dpt:111
    3   252 ACCEPT     udp  --  *      *       192.168.56.104       0.0.0.0/0            udp dpt:111
  529 48388 ACCEPT     tcp  --  *      *       192.168.56.151       0.0.0.0/0            tcp dpt:204
9
    0     0 ACCEPT     udp  --  *      *       192.168.56.151       0.0.0.0/0            udp dpt:204
9
    0     0 ACCEPT     tcp  --  *      *       192.168.56.151       0.0.0.0/0            tcp dpt:111
    1    84 ACCEPT     udp  --  *      *       192.168.56.151       0.0.0.0/0            udp dpt:111
  491 30308 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:204
9
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:204
9
   68  4048 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:111
   12  1440 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:111

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 1812 packets, 132K bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Result:
The rules show accepted traffic from whitelisted IPs and dropped packets from others, including logs of denied NFS attempts if logging is enabled.

Conclusion

These tests confirm the firewall configuration:

- ✓ Successfully blocks the CVE-1999-0170 attack from unauthorized sources.
- ✓ Allows legitimate access to the NFS service without disruption.
- ✓ Implements effective prevention without false positives or false negatives.

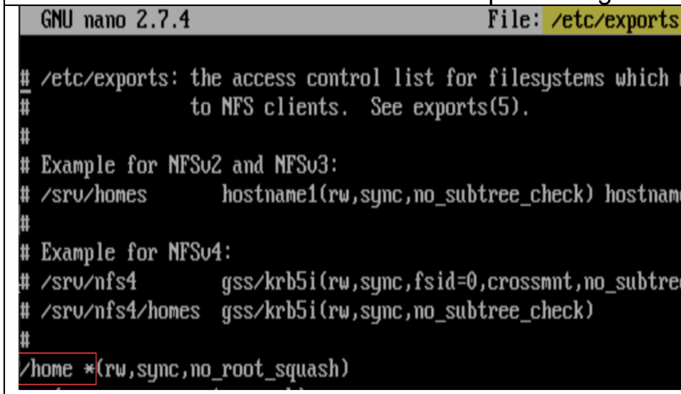
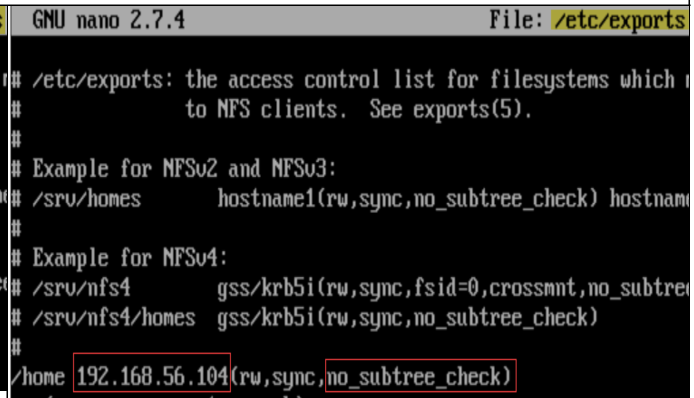
2.3 By reviewing the scan results (obtained in project – Part 1), propose and describe any additional defense strategy to protect the target systems (2%).

Solution:
The scan results from Part 1 revealed that the target system (uranusZ) exposed several open services: SSH (22), HTTP (80), Portmapper (111), and NFS (2049). While the initial defense strategy focused on restricting access to NFS and Portmapper (RPC), additional measures can strengthen system security:

1. Harden NFS Export Configuration

On UranusZ (192.168.56.105), the NFS export was originally configured as /home *, allowing access from any host. This exposes the system to unauthorized mounts. To restrict access, the /etc/exports file should be updated to allow only the trusted internal hosts, for example: UranusN (192.168.56.104):

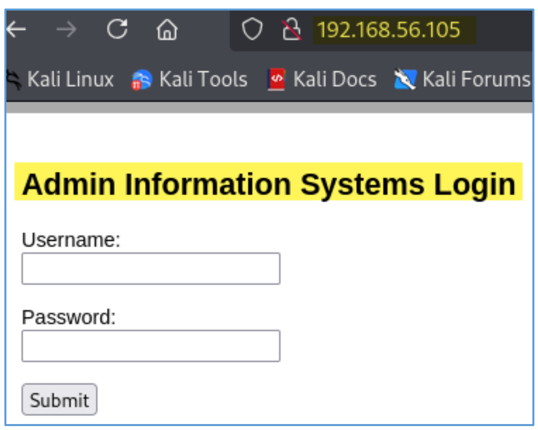
NFS export configuration Correction on UranusZ

Before	After
	

This export rule ensures that only the specified host can mount the shared directory, adding an extra layer of access control beyond IPTables.

2. Disable Unnecessary Services

The Nmap scan from Part 1 shows that UranusZ (192.168.56.105) has port 80 open, running Nginx version 1.15.10. This adds an additional attack surface to the system, and this HTTP service revealed a web login over unencrypted HTTP, making it possible for attackers on the same network to capture usernames and passwords.



a

Figure. Admin login page exposed on port 80 (HTTP) of UranusZ

To reduce risk and protect the target system, the Nginx service could be disabled using these commands, if this web login visualization is not mandatory to be public:

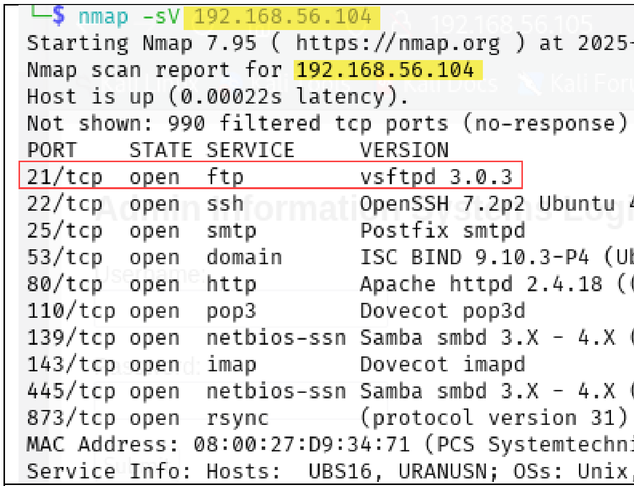
```
sudo systemctl stop nginx
sudo systemctl disable nginx
```

3. Apply system updates to patch known vulnerabilities

Some services such as OpenSSH and Nginx were running outdated versions, which are known to have vulnerabilities, so running security updates is a simple and effective way to fix such issues.

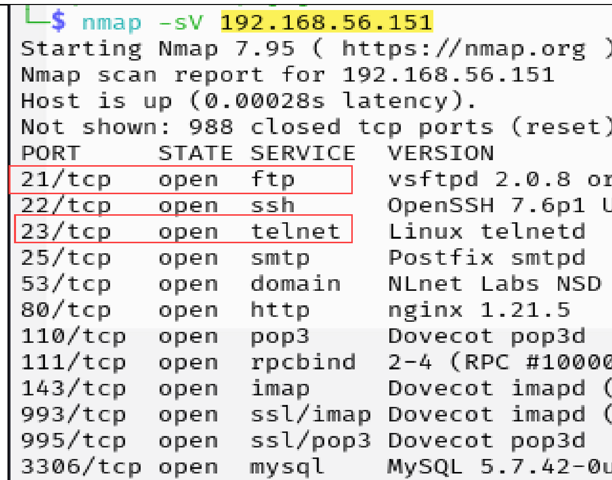
4. Remove Legacy or Insecure Protocols from Other Hosts

The other hosts: UranusN and UranusR were found to have FTP (21), and Telnet (23) services open. These services are insecure and should be disabled or replaced by more secure protocols such as SFTP.



```
└─$ nmap -sV 192.168.56.104
Starting Nmap 7.95 ( https://nmap.org ) at 2025-
Nmap scan report for 192.168.56.104
Host is up (0.00022s latency).
Not shown: 990 filtered tcp ports (no-response)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 3.0.3
22/tcp   open  ssh         OpenSSH 7.2p2 Ubuntu 4
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.10.3-P4 (Ub
80/tcp   open  http        Apache httpd 2.4.18 ((
110/tcp  open  pop3        Dovecot pop3d
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (
143/tcp  open  imap        Dovecot imapd
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (
873/tcp  open  rsync       (protocol version 31)
MAC Address: 08:00:27:D9:34:71 (PCS Systemtechni
Service Info: Hosts:  UBS16, URANUSN; OSs: Unix,
```

UranusN - Port scanning showing FTP enabled



```
└─$ nmap -sV 192.168.56.151
Starting Nmap 7.95 ( https://nmap.org )
Nmap scan report for 192.168.56.151
Host is up (0.00028s latency).
Not shown: 988 closed tcp ports (reset)
PORT     STATE SERVICE  VERSION
21/tcp   open  ftp      vsftpd 2.0.8 or
22/tcp   open  ssh      OpenSSH 7.6p1 U
23/tcp   open  telnet   Linux telnetd
25/tcp   open  smtp     Postfix smtpd
53/tcp   open  domain   NLnet Labs NSD
80/tcp   open  http     nginx 1.21.5
110/tcp  open  pop3     Dovecot pop3d
111/tcp  open  rpcbind  2-4 (RPC #10000
143/tcp  open  imap     Dovecot imapd (
993/tcp  open  ssl/imap Dovecot imapd (
995/tcp  open  ssl/pop3 Dovecot pop3d
3306/tcp open  mysql    MySQL 5.7.42-0u
```

UranusR - Port scanning showing FTP and Telnet enabled