# SECURITY+ V4 LAB SERIES

# Lab 7: Performing Active Reconnaissance

**Document Version: 2023-02-27**

| Material in this Lab Aligns to the Following | |
|---|---|
| CompTIA Security+ (SY0-601)<br>Exam Objectives | 1.7: Summarize the techniques used in security assessments<br>1.8: Explain the techniques used in penetration testing<br>4.1: Given a scenario, use the appropriate tool to assess organizational security |
| All-In-One CompTIA Security+ Sixth Edition<br>ISBN-13: 978-1260464009<br>Chapters | 7: Security Assessments<br>8: Penetration Testing<br>26: Tools/Assess Organizational Security |

# Contents

## Introduction

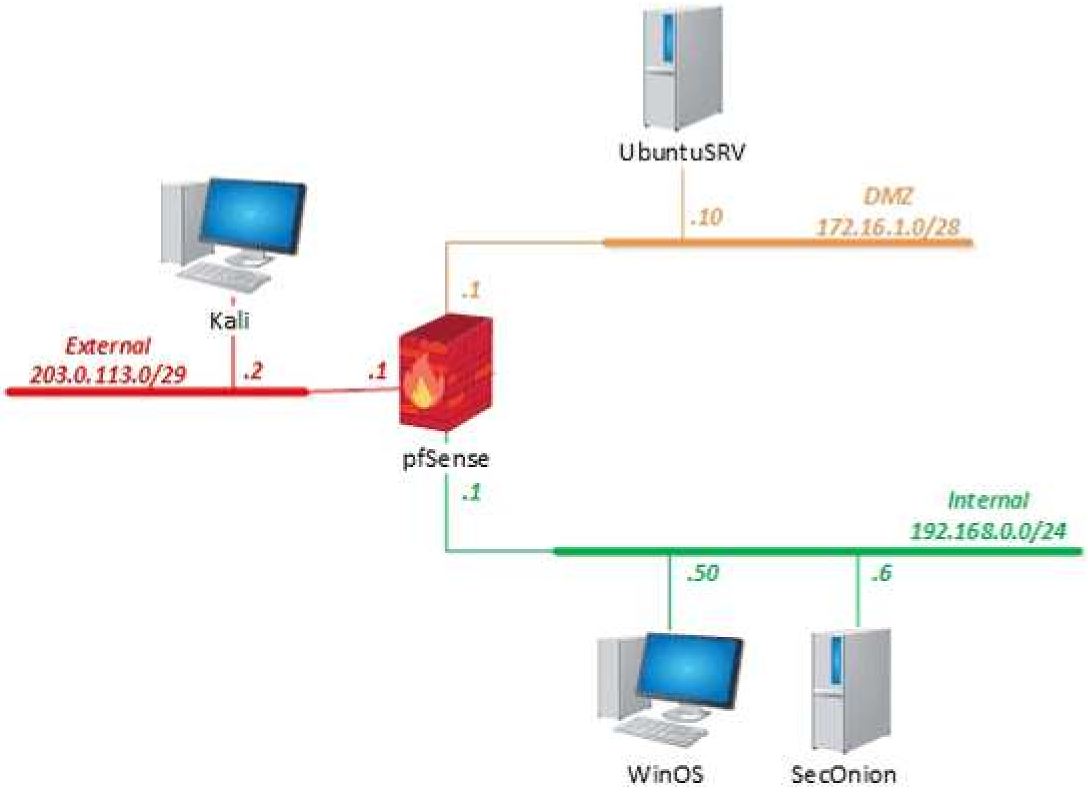
In this lab, you will use PowerShell for active reconnaissance on a Windows server. Also, you will use a variety of tools to finish the same types of tasks on Linux.

## Objective

In this lab, you will perform the following tasks:

- Experience active reconnaissance in Windows and in Linux
- Scan the network for vulnerable systems

## Lab Topology



## Lab Settings

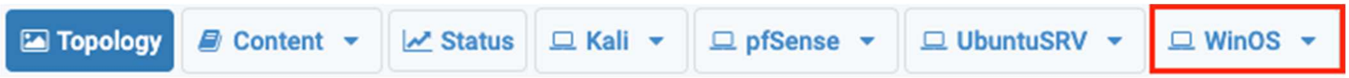
The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

| Virtual Machine | IP Address | Account<br>(if needed) | Password<br>(if needed) |
|---|---|---|---|
| Kali | `203.0.113.2` | `kali` | `kali` |
| pfSense | `192.168.0.1` | `sysadmin` | `NDGlabpass123!` |
| SecOnion | `192.168.0.6` | `sysadmin` | `NDGlabpass123!` |
| UbuntuSRV | `172.16.1.10` | `sysadmin` | `NDGlabpass123!` |
| WinOS | `192.168.0.50` | `Administrator` | `NDGlabpass123!` |

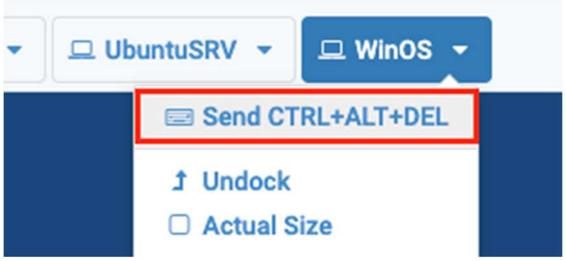
# 1 Use PowerShell to Perform an Active Reconnaissance of a Windows Server

In this section, you will utilize *PowerShell* on the *Windows* server to gather extensive information.

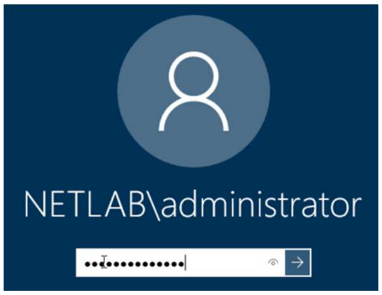
1. Launch the **WinOS** virtual machine to access the graphical login screen.



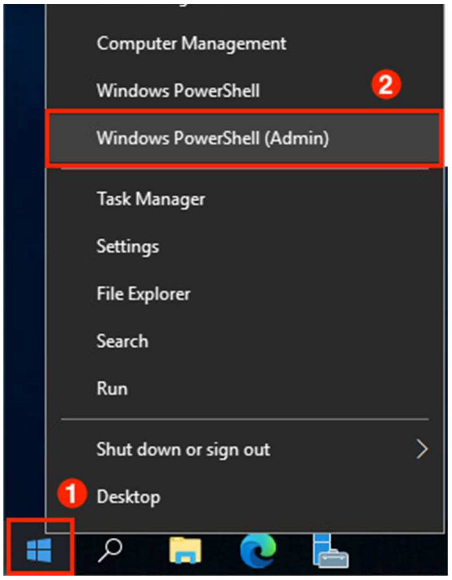
2. While on the splash screen, focus on the *NETLAB+* tabs. Click the dropdown menu for the **WinOS** tab and click on **Send CTRL+ALT+DEL**.



3. Log in as `administrator` using the password `NDGlabpass123!`.

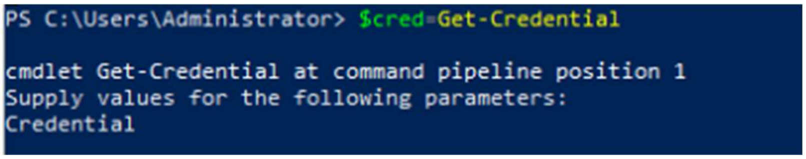


4. Leave the *Server Manager* window untouched (or you can minimize it). Then, right-click on the **Windows** logo in the taskbar and click **Windows PowerShell (admin)**.



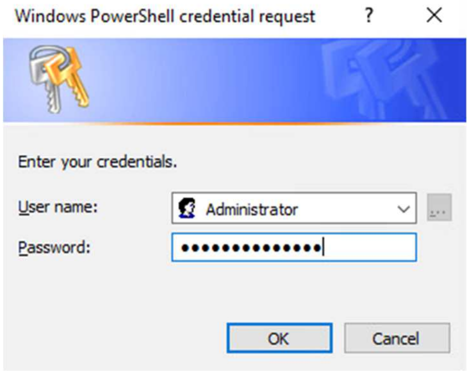
5. In the *PowerShell* window, type the command below, followed by pressing the **Enter** key.

```
PS C:\Users\Administrator> $cred=Get-Credential
```



```
PS C:\Users\Administrator> $cred=Get-Credential

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
```

6. Notice a pop-up window appears. Type `Administrator` in the *User name* field, followed by typing `NDGlabpass123!` in the *Password* field. Click **OK**.



Windows PowerShell credential request

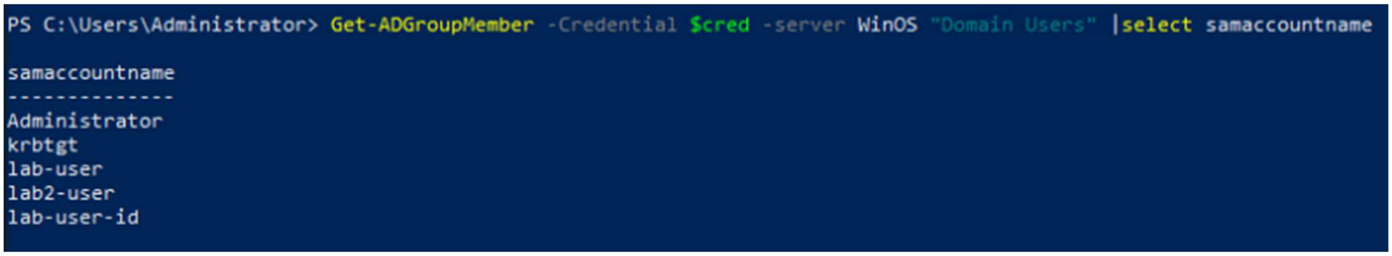
Enter your credentials.

User name: Administrator

Password: ••••••••••••••

7. Back on the *PowerShell* prompt, enter the command below to retrieve a list of domain users on the system.

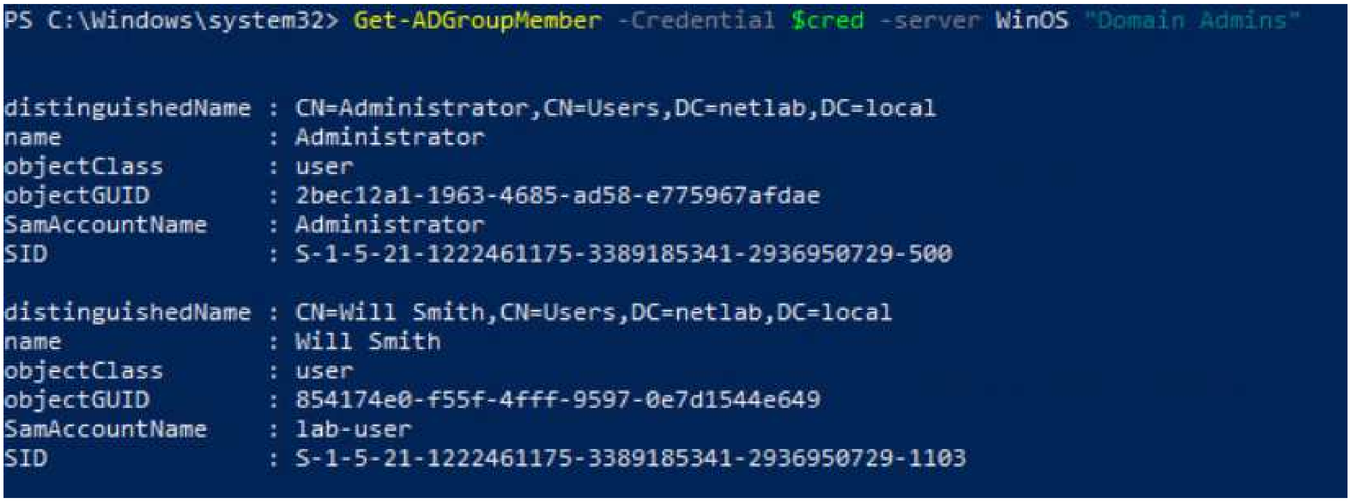
```
PS C:\Users\Administrator> Get-ADGroupMember -Credential $cred -server WinOS
“Domain Users” | select samaccountname
```



```
PS C:\Users\Administrator> Get-ADGroupMember -Credential $cred -server WinOS "Domain Users" |select samaccountname

samaccountname
--------------
Administrator
krbtgt
lab-user
lab2-user
lab-user-id
```

8. Enter the command below to identify which users are *Domain Admin Members.*

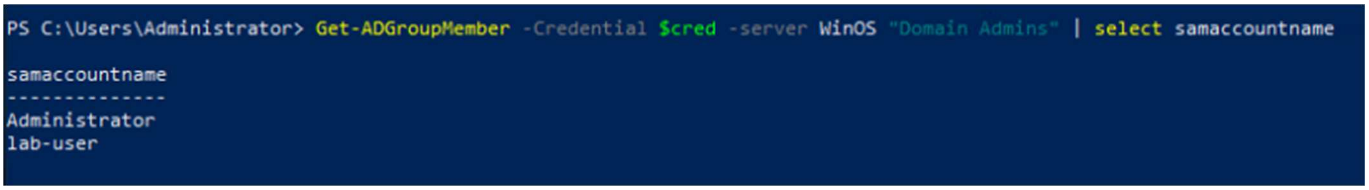
```
PS C:\Users\Administrator> Get-ADGroupMember –Credential $cred –server WinOS
“Domain Admins”
```



```
PS C:\Windows\system32> Get-ADGroupMember -Credential $cred -server WinOS "Domain Admins"


distinguishedName : CN=Administrator,CN=Users,DC=netlab,DC=local
name              : Administrator
objectClass       : user
objectGUID        : 2bec12a1-1963-4685-ad58-e775967afdae
SamAccountName    : Administrator
SID               : S-1-5-21-1222461175-3389185341-2936950729-500

distinguishedName : CN=Will Smith,CN=Users,DC=netlab,DC=local
name              : Will Smith
objectClass       : user
objectGUID        : 854174e0-f55f-4fff-9597-0e7d1544e649
SamAccountName    : lab-user
SID               : S-1-5-21-1222461175-3389185341-2936950729-1103
```

9. Filter the *SAM* account names.

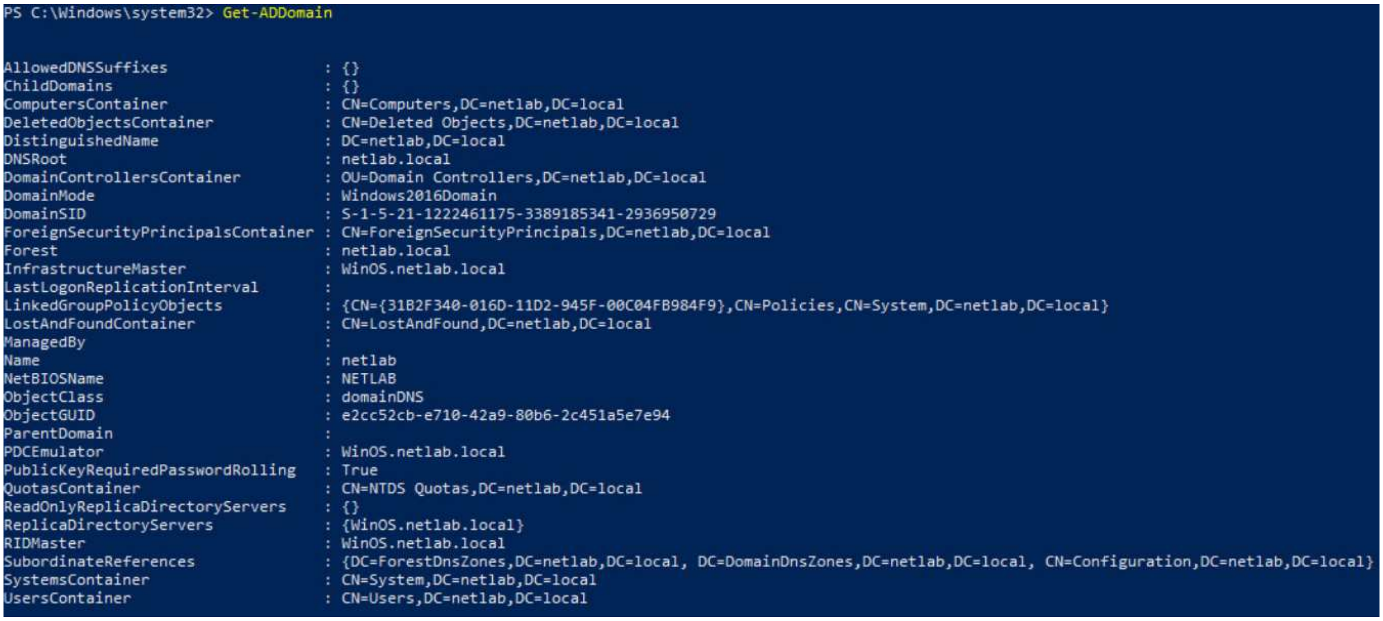
```
PS C:\Users\Administrator> Get-ADGroupMember –Credential $cred –server WinOS
“Domain Admins” | select samaccountname
```



```
PS C:\Users\Administrator> Get-ADGroupMember -Credential $cred -server WinOS "Domain Admins" | select samaccountname

samaccountname
--------------
Administrator
lab-user
```

10. View the domain itself.

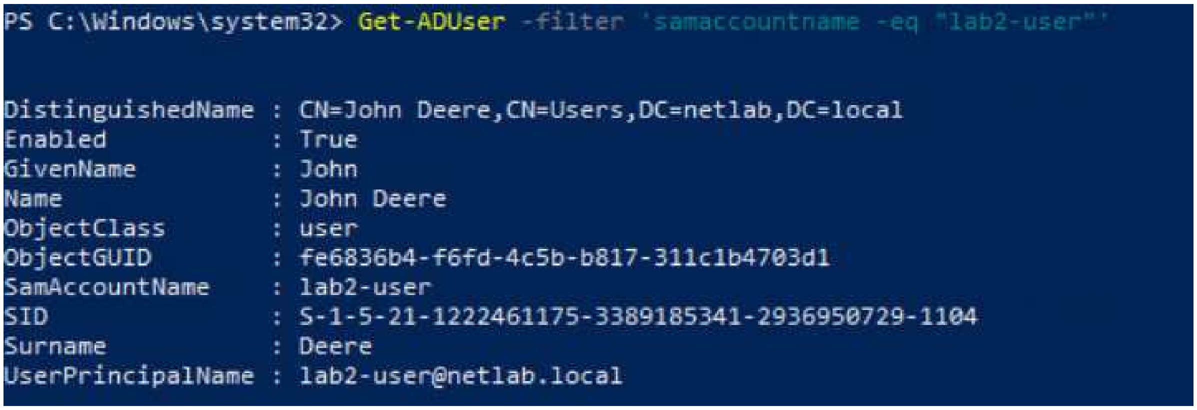
```
PS C:\Users\Administrator> Get-ADDomain
```



```
PS C:\Windows\system32> Get-ADDomain


AllowedDNSSuffixes                 : {}
ChildDomains                       : {}
ComputersContainer                 : CN=Computers,DC=netlab,DC=local
DeletedObjectsContainer            : CN=Deleted Objects,DC=netlab,DC=local
DistinguishedName                  : DC=netlab,DC=local
DNSRoot                            : netlab.local
DomainControllersContainer         : OU=Domain Controllers,DC=netlab,DC=local
DomainMode                         : Windows2016Domain
DomainSID                          : S-1-5-21-1222461175-3389185341-2936950729
ForeignSecurityPrincipalsContainer : CN=ForeignSecurityPrincipals,DC=netlab,DC=local
Forest                             : netlab.local
InfrastructureMaster               : WinOS.netlab.local
LastLogonReplicationInterval       :
LinkedGroupPolicyObjects           : {CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=netlab,DC=local}
LostAndFoundContainer              : CN=LostAndFound,DC=netlab,DC=local
ManagedBy                          :
Name                               : netlab
NetBIOSName                        : NETLAB
ObjectClass                        : domainDNS
ObjectGUID                         : e2cc52cb-e710-42a9-80b6-2c451a5e7e94
ParentDomain                       :
PDCEmulator                        : WinOS.netlab.local
PublicKeyRequiredPasswordRolling   : True
QuotasContainer                    : CN=NTDS Quotas,DC=netlab,DC=local
ReadOnlyReplicaDirectoryServers    : {}
ReplicaDirectoryServers            : {WinOS.netlab.local}
RIDMaster                          : WinOS.netlab.local
SubordinateReferences              : {DC=ForestDnsZones,DC=netlab,DC=local, DC=DomainDnsZones,DC=netlab,DC=local, CN=Configuration,DC=netlab,DC=local}
SystemsContainer                   : CN=System,DC=netlab,DC=local
UsersContainer                     : CN=Users,DC=netlab,DC=local
```

11. See whether the *lab2-user* account is currently enabled.

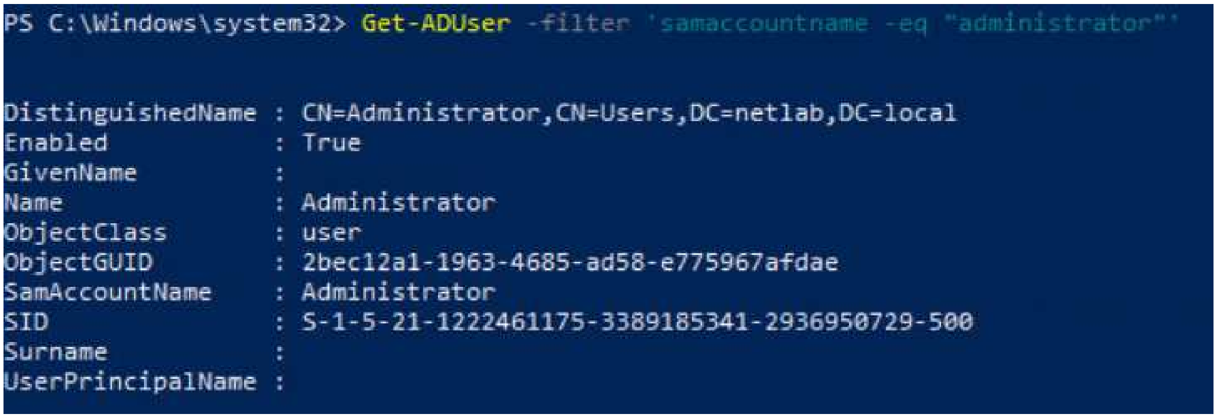
```
PS C:\Users\Administrator> Get-ADUser –filter ‘samaccountname –eq “lab2-user”’
```



```
PS C:\Windows\system32> Get-ADUser -filter 'samaccountname -eq "lab2-user"'


DistinguishedName : CN=John Deere,CN=Users,DC=netlab,DC=local
Enabled           : True
GivenName         : John
Name              : John Deere
ObjectClass       : user
ObjectGUID        : fe6836b4-f6fd-4c5b-b817-311c1b4703d1
SamAccountName    : lab2-user
SID               : S-1-5-21-1222461175-3389185341-2936950729-1104
Surname           : Deere
UserPrincipalName : lab2-user@netlab.local
```

12. Not only do we see that the account *lab2-user* is enabled, but we also have the account's *SID* as well. Try to retrieve more information about the *Administrator* account by entering the command below.

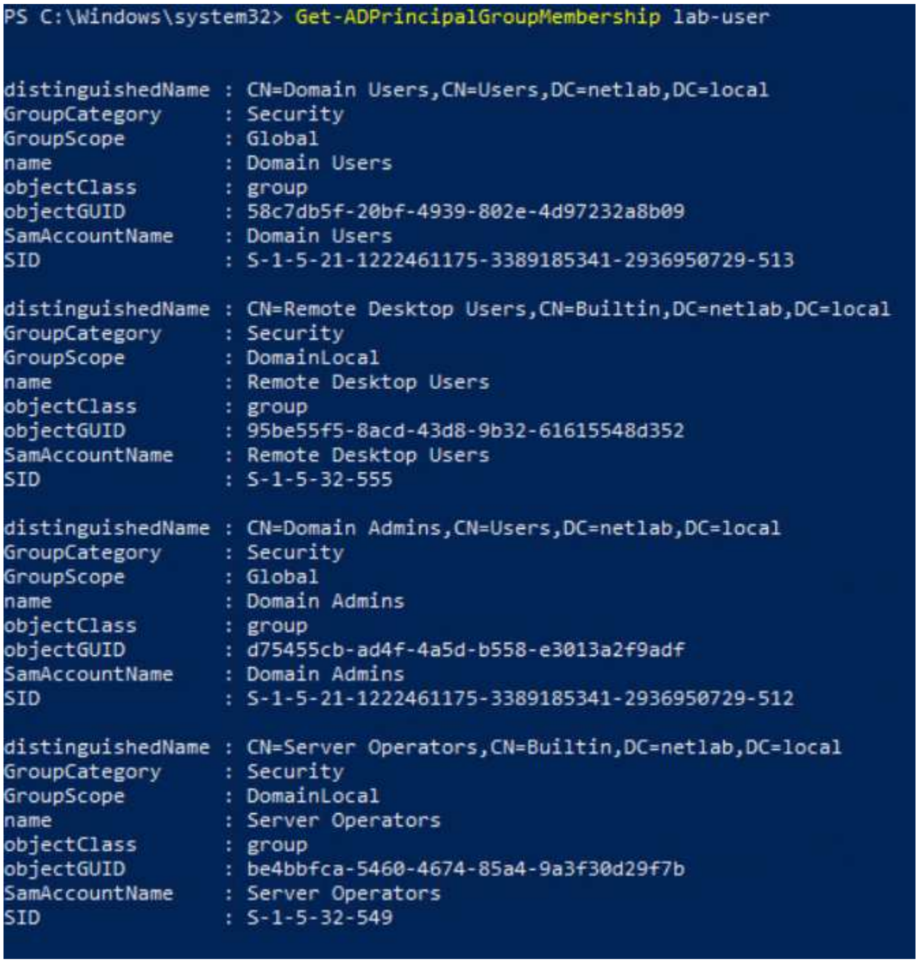
```
PS C:\Users\Administrator> Get-ADUser –filter ‘samaccountname –eq “administrator”’
```



```
PS C:\Windows\system32> Get-ADUser -filter 'samaccountname -eq "administrator"'


DistinguishedName : CN=Administrator,CN=Users,DC=netlab,DC=local
Enabled           : True
GivenName         :
Name              : Administrator
ObjectClass       : user
ObjectGUID        : 2bec12a1-1963-4685-ad58-e775967afdae
SamAccountName    : Administrator
SID               : S-1-5-21-1222461175-3389185341-2936950729-500
Surname           :
UserPrincipalName :
```

13. View the **lab-user** account user's group memberships and confirm whether the account belongs to the *Domain Admins* group.

```
PS C:\Users\Administrator> Get-ADPrincipalGroupMembership lab-user
```



```
PS C:\Windows\system32> Get-ADPrincipalGroupMembership lab-user


distinguishedName : CN=Domain Users,CN=Users,DC=netlab,DC=local
GroupCategory     : Security
GroupScope        : Global
name              : Domain Users
objectClass       : group
objectGUID        : 58c7db5f-20bf-4939-802e-4d97232a8b09
SamAccountName    : Domain Users
SID               : S-1-5-21-1222461175-3389185341-2936950729-513

distinguishedName : CN=Remote Desktop Users,CN=Builtin,DC=netlab,DC=local
GroupCategory     : Security
GroupScope        : DomainLocal
name              : Remote Desktop Users
objectClass       : group
objectGUID        : 95be55f5-8acd-43d8-9b32-61615548d352
SamAccountName    : Remote Desktop Users
SID               : S-1-5-32-555

distinguishedName : CN=Domain Admins,CN=Users,DC=netlab,DC=local
GroupCategory     : Security
GroupScope        : Global
name              : Domain Admins
objectClass       : group
objectGUID        : d75455cb-ad4f-4a5d-b558-e3013a2f9adf
SamAccountName    : Domain Admins
SID               : S-1-5-21-1222461175-3389185341-2936950729-512

distinguishedName : CN=Server Operators,CN=Builtin,DC=netlab,DC=local
GroupCategory     : Security
GroupScope        : DomainLocal
name              : Server Operators
objectClass       : group
objectGUID        : be4bbfca-5460-4674-85a4-9a3f30d29f7b
SamAccountName    : Server Operators
SID               : S-1-5-32-549
```

It can be verified that the *lab-user* is part of the *Domain Admins* group as well as other groups.

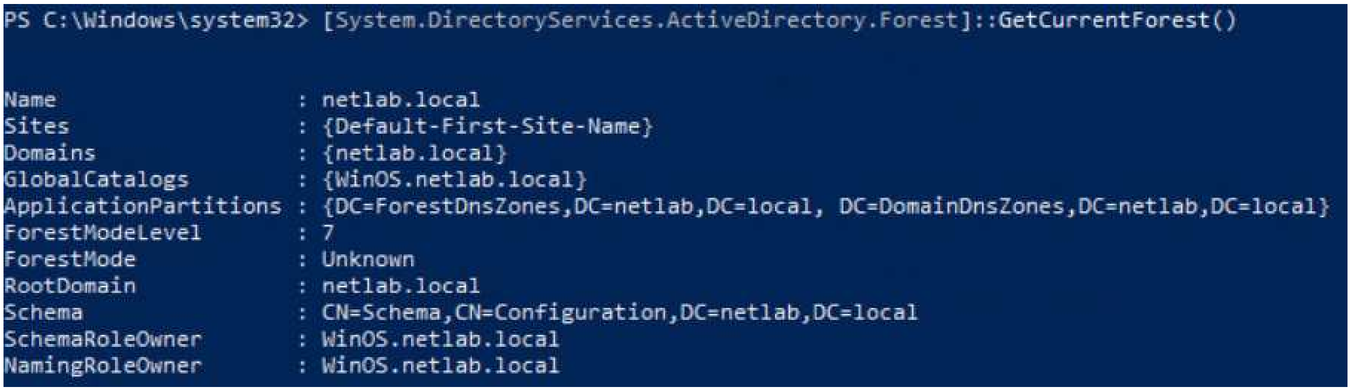
14. Type `cls`, and press **Enter**; leave the *PowerShell* window open to continue with the next task.

## 2 Use PowerShell to Perform an Active Reconnaissance of a Windows Client

In this task, you will utilize *PowerShell* on a *Windows* system to gather extensive information**.**

1. Identify the *Active Directory* that *lab-user* belongs to by entering the *.NET* command with *PowerShell* below.

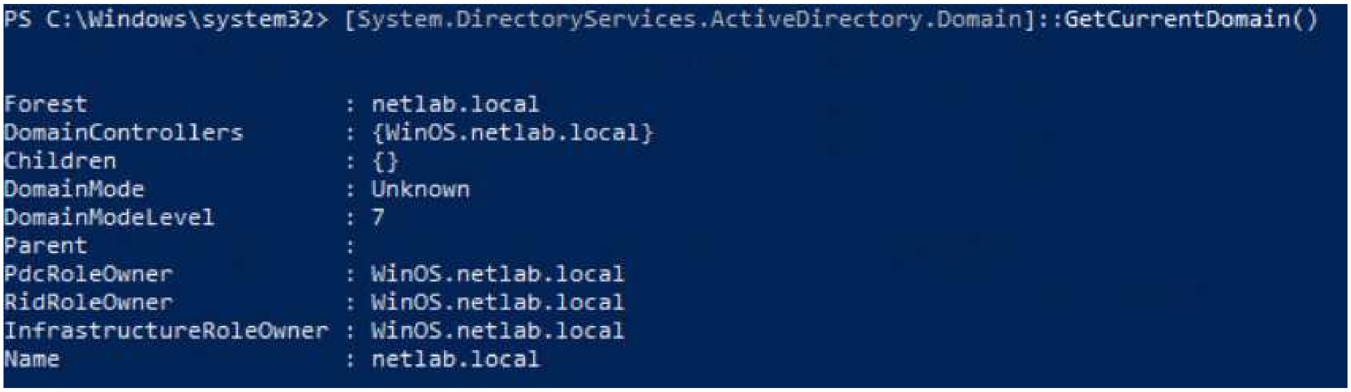
```
PS C:\Users\Administrator>
[System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentForest()
```



```
PS C:\Windows\system32> [System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentForest()


Name                   : netlab.local
Sites                  : {Default-First-Site-Name}
Domains                : {netlab.local}
GlobalCatalogs         : {WinOS.netlab.local}
ApplicationPartitions : {DC=ForestDnsZones,DC=netlab,DC=local, DC=DomainDnsZones,DC=netlab,DC=local}
ForestModeLevel        : 7
ForestMode             : Unknown
RootDomain             : netlab.local
Schema                 : CN=Schema,CN=Configuration,DC=netlab,DC=local
SchemaRoleOwner        : WinOS.netlab.local
NamingRoleOwner        : WinOS.netlab.local
```

2. Since the forest is different from a domain, identify which domain the user is associated with.

```
PS C:\Users\Administrator>
[System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()
```



```
PS C:\Windows\system32> [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()


Forest                  : netlab.local
DomainControllers       : {WinOS.netlab.local}
Children                : {}
DomainMode              : Unknown
DomainModeLevel         : 7
Parent                  :
PdcRoleOwner            : WinOS.netlab.local
RidRoleOwner            : WinOS.netlab.local
InfrastructureRoleOwner : WinOS.netlab.local
Name                    : netlab.local
```

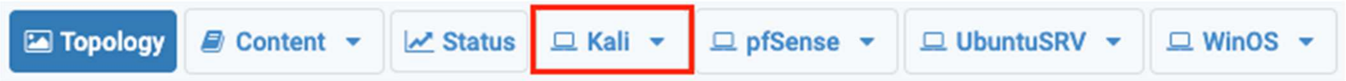
Using *PowerShell*, you successfully obtained the domain name, forest name, and group membership.

3. The Windows reconnaissance portion is now complete. You may proceed with the Linux portion next.

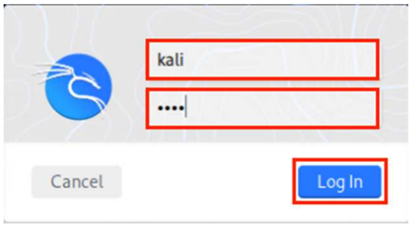
## 3 Scanning the Network for Vulnerable Systems

In this section, you will utilize *nmap* on the *Linux* machine to gather network information.

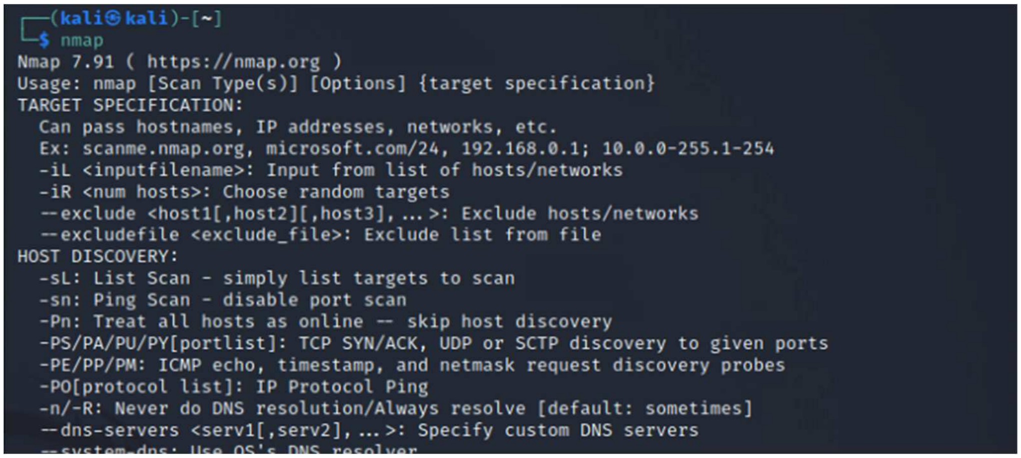
1. Launch the **Kali** virtual machine to access the graphical login screen.



2. Log in as `kali` with `kali` as the password.



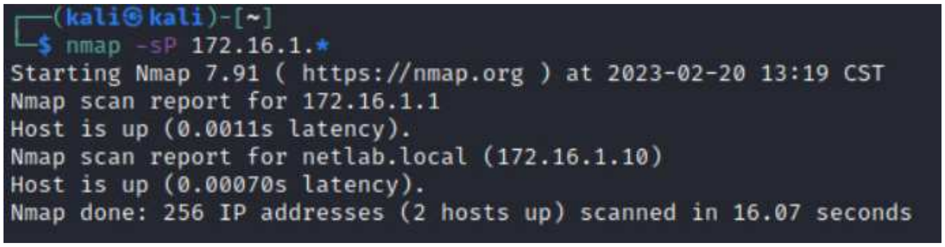
3. Open a new **terminal** and view the available options that can be used with *Nmap* by typing **nmap** into the *terminal* followed by pressing the **Enter key**.



```
┌──(kali㉿kali)-[~]
└─$ nmap
Nmap 7.91 ( https://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[protocol list]: IP Protocol Ping
  -n/-R: Never do DNS resolution/Always resolve [default: sometimes]
  --dns-servers <serv1[,serv2],...>: Specify custom DNS servers
  --system-dns: Use OS's DNS resolver
```

4. Initiate a **quick ping scan** to identify live hosts with a network ID of *172.16.1.*.*

```
kali@kali$ nmap -sP 172.16.1.*
```

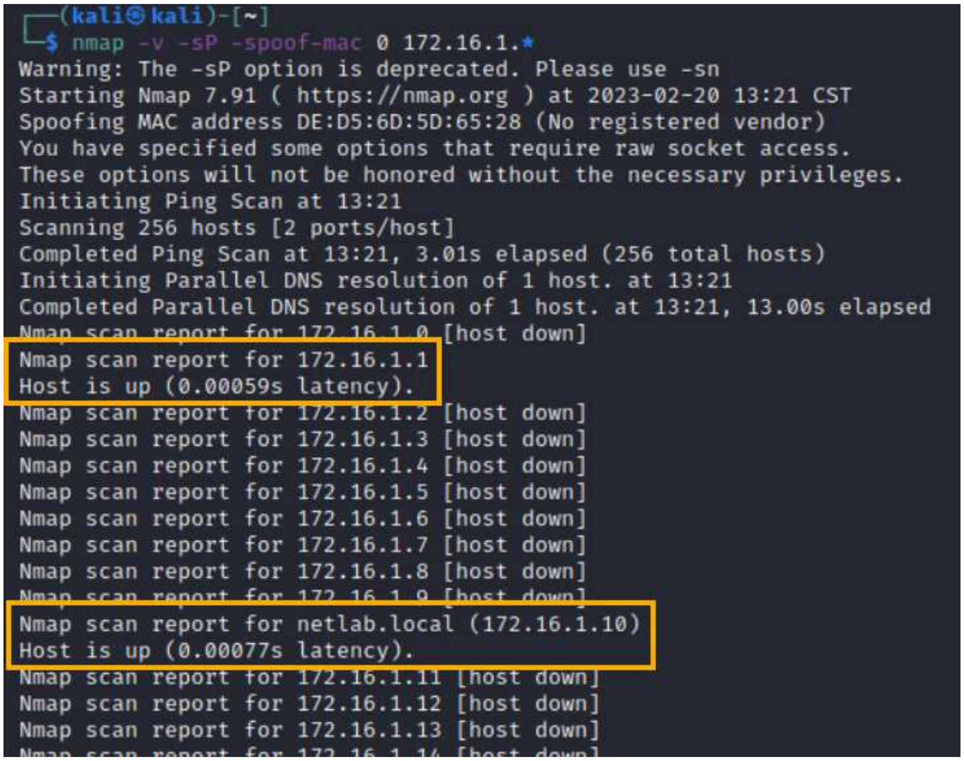


```
┌──(kali㉿kali)-[~]
└─$ nmap -sP 172.16.1.*
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 13:19 CST
Nmap scan report for 172.16.1.1
Host is up (0.0011s latency).
Nmap scan report for netlab.local (172.16.1.10)
Host is up (0.00070s latency).
Nmap done: 256 IP addresses (2 hosts up) scanned in 16.07 seconds
```

You should see two host results; *172.16.1.1* as the *DMZ gateway*, and *172.16.1.10* as Ubuntu Server.

5. Initiate a **ping scan** while spoofing the source *MAC* address at the same time.

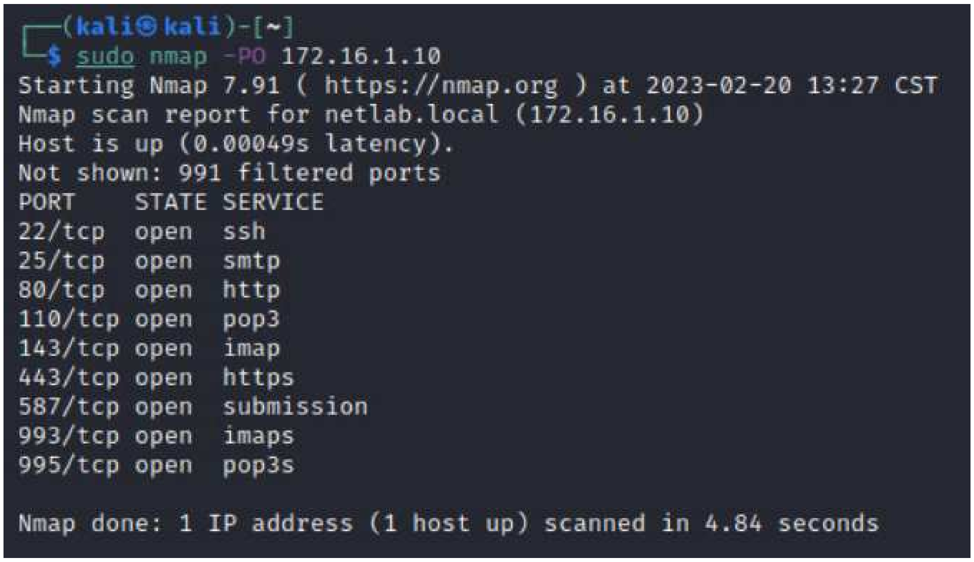
```
kali@kali$ nmap -v -sP -spoof-mac 0 172.16.1.*
```



```
┌──(kali㉿kali)-[~]
└─$ nmap -v -sP -spoof-mac 0 172.16.1.*
Warning: The -sP option is deprecated. Please use -sn
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 13:21 CST
Spoofing MAC address DE:D5:6D:5D:65:28 (No registered vendor)
You have specified some options that require raw socket access.
These options will not be honored without the necessary privileges.
Initiating Ping Scan at 13:21
Scanning 256 hosts [2 ports/host]
Completed Ping Scan at 13:21, 3.01s elapsed (256 total hosts)
Initiating Parallel DNS resolution of 1 host. at 13:21
Completed Parallel DNS resolution of 1 host. at 13:21, 13.00s elapsed
Nmap scan report for 172.16.1.0 [host down]
Nmap scan report for 172.16.1.1
Host is up (0.00059s latency).
Nmap scan report for 172.16.1.2 [host down]
Nmap scan report for 172.16.1.3 [host down]
Nmap scan report for 172.16.1.4 [host down]
Nmap scan report for 172.16.1.5 [host down]
Nmap scan report for 172.16.1.6 [host down]
Nmap scan report for 172.16.1.7 [host down]
Nmap scan report for 172.16.1.8 [host down]
Nmap scan report for 172.16.1.9 [host down]
Nmap scan report for netlab.local (172.16.1.10)
Host is up (0.00077s latency).
Nmap scan report for 172.16.1.11 [host down]
Nmap scan report for 172.16.1.12 [host down]
Nmap scan report for 172.16.1.13 [host down]
Nmap scan report for 172.16.1.14 [host down]
```

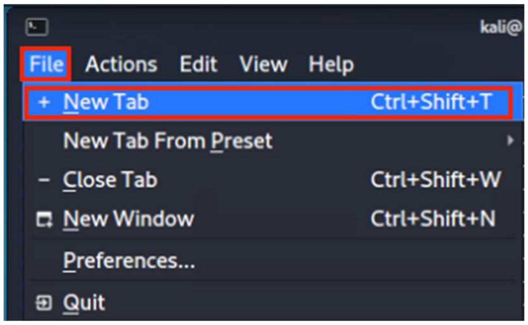
6. When scanning for active systems on a network, *Nmap* also gives the ability to scan for which *IP protocols* are supported by the host involved in the scanning process. Enter the command below, and enter the password **`kali`** when prompted.

```
kali@kali$ sudo nmap -PO 172.16.1.10
```



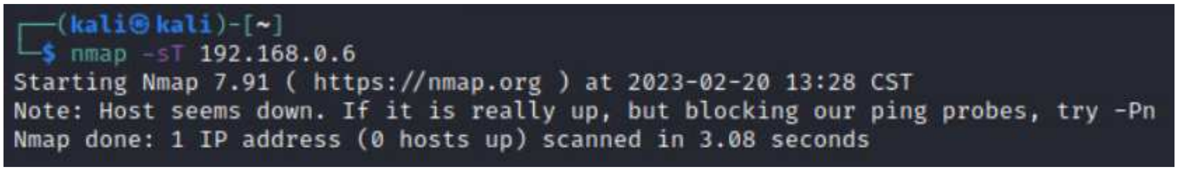
```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -PO 172.16.1.10
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 13:27 CST
Nmap scan report for netlab.local (172.16.1.10)
Host is up (0.00049s latency).
Not shown: 991 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
80/tcp  open  http
110/tcp open  pop3
143/tcp open  imap
443/tcp open  https
587/tcp open  submission
993/tcp open  imaps
995/tcp open  pop3s

Nmap done: 1 IP address (1 host up) scanned in 4.84 seconds
```

7. In the T*erminal* window, select **File** from the top menu pane and click on **New Tab.**



8. While engaged in the new tab, initiate a *Transmission Control Protocol (TCP)* scan against the *SecOnion* system. Type the following command:

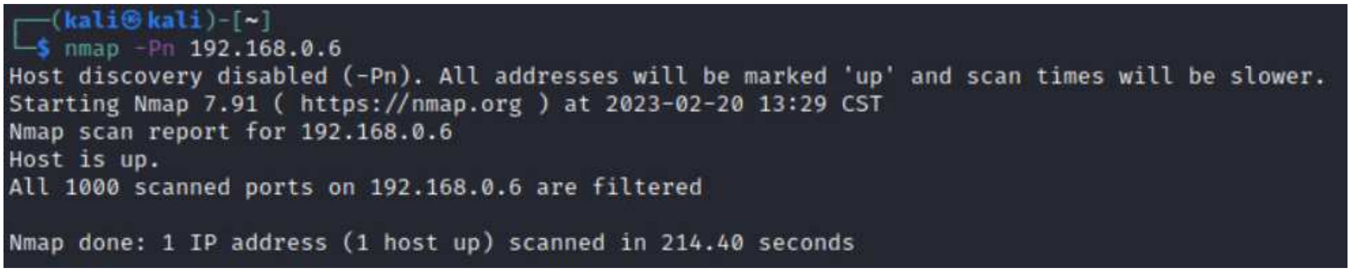
```
kali@kali$ nmap -sT 192.168.0.6
```



```
┌──(kali㉿kali)-[~]
└─$ nmap -sT 192.168.0.6
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 13:28 CST
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.08 seconds
```

9. Notice that the scan came back stating that the host seemed to be down and that it was unable to scan it. Attempt another scan against the same system, but this time using the no ping option.

```
kali@kali$ nmap -PN 192.168.0.6
```

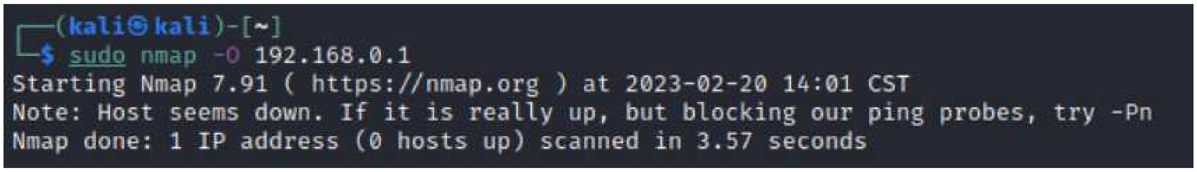


```
┌──(kali㉿kali)-[~]
└─$ nmap -Pn 192.168.0.6
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 13:29 CST
Nmap scan report for 192.168.0.6
Host is up.
All 1000 scanned ports on 192.168.0.6 are filtered

Nmap done: 1 IP address (1 host up) scanned in 214.40 seconds
```

After 2-3 minutes, notice that nmap scan now states that the host is up but that the first 1000 ports are being filtered; most likely due to a firewall.

10. Initiate an **operating system scan** against the *pfSense* system to help identify what version of *OS* it is running on.

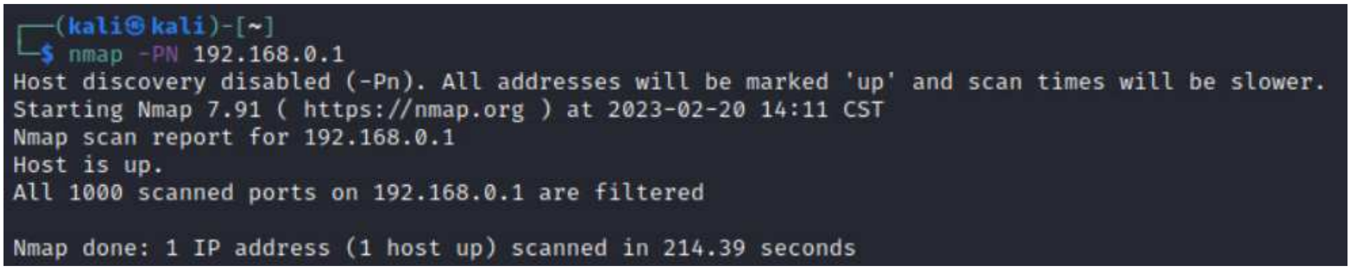
```
kali@kali$ sudo nmap -O 192.168.0.1
```



```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -O 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:01 CST
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.57 seconds
```

11. Notice that the nmap scan returned with a host is down message. Scan the host again, this time, using the -PN option to identify whether the host is actually up or down.

```
kali@kali$ nmap -PN 192.168.0.1
```

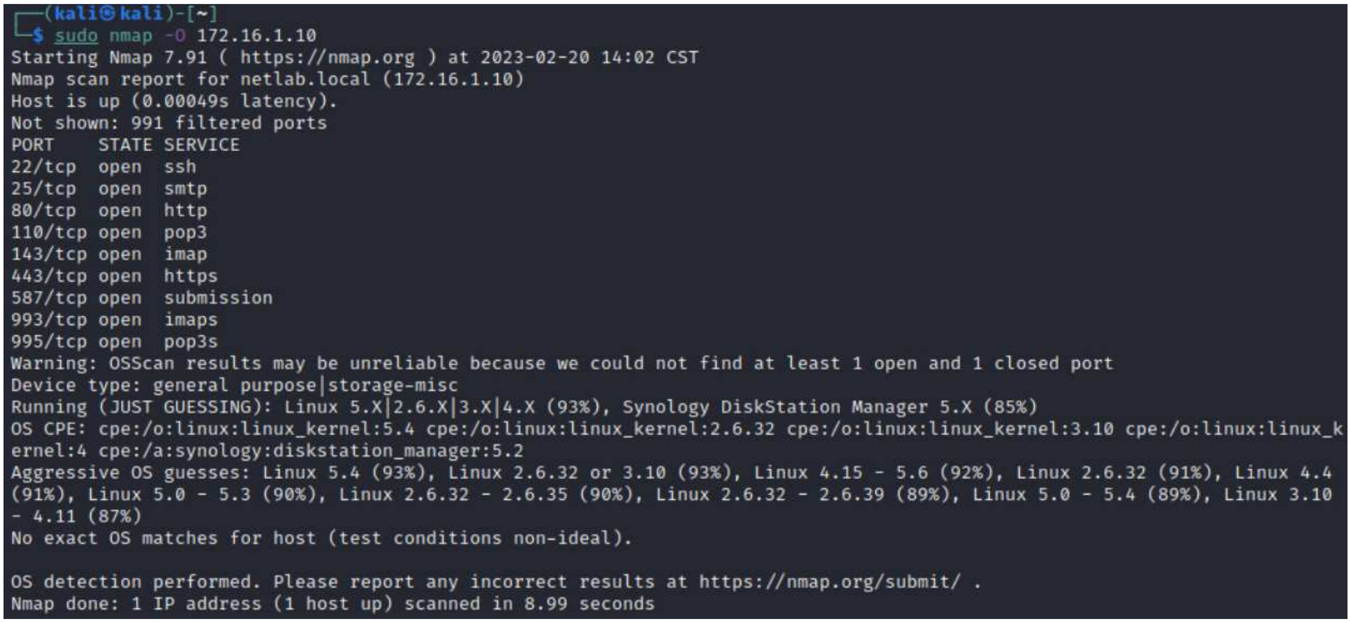


```
┌──(kali㉿kali)-[~]
└─$ nmap -PN 192.168.0.1
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:11 CST
Nmap scan report for 192.168.0.1
Host is up.
All 1000 scanned ports on 192.168.0.1 are filtered

Nmap done: 1 IP address (1 host up) scanned in 214.39 seconds
```

After 2-3 minutes, notice that nmap scan now states that the host is up but that the first 1000 ports are being filtered.

12. Initiate the same scan from the previous step but this time against the *UbuntuSRV* system.

```
kali@kali$ sudo nmap -O 172.16.1.10
```

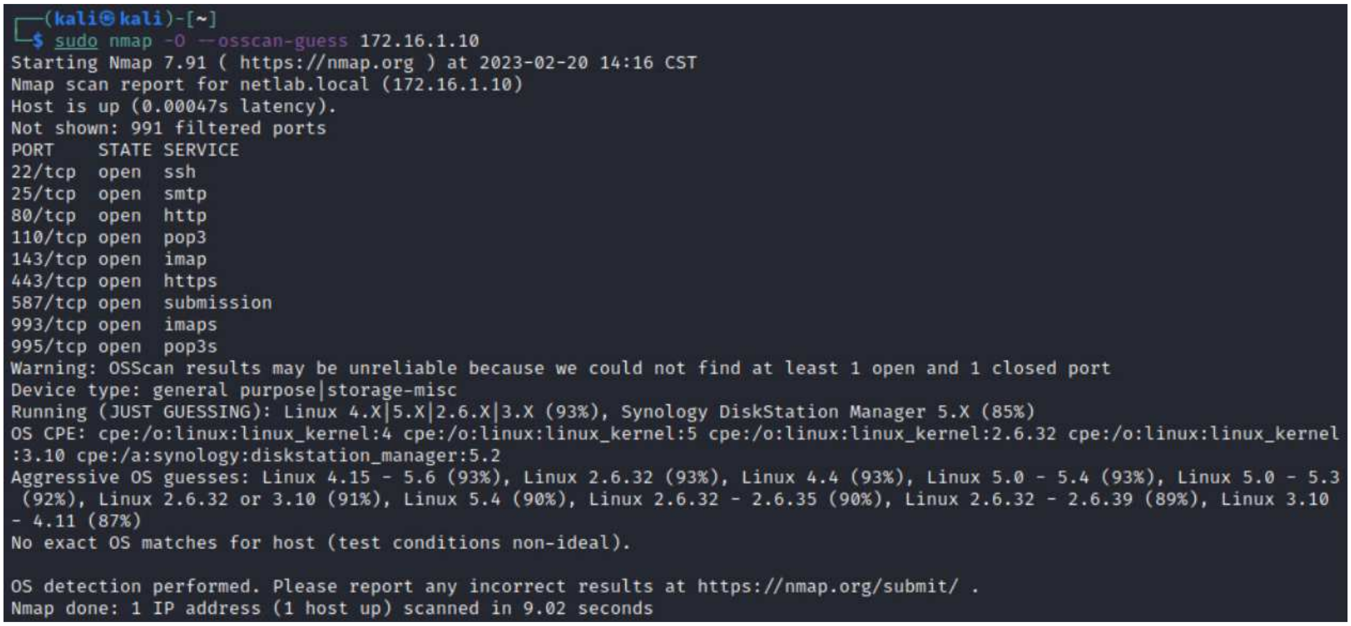


```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -O 172.16.1.10
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:02 CST
Nmap scan report for netlab.local (172.16.1.10)
Host is up (0.00049s latency).
Not shown: 991 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
80/tcp  open  http
110/tcp open  pop3
143/tcp open  imap
443/tcp open  https
587/tcp open  submission
993/tcp open  imaps
995/tcp open  pop3s
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|storage-misc
Running (JUST GUESSING): Linux 5.X|2.6.X|3.X|4.X (93%), Synology DiskStation Manager 5.X (85%)
OS CPE: cpe:/o:linux:linux_kernel:5.4 cpe:/o:linux:linux_kernel:2.6.32 cpe:/o:linux:linux_kernel:3.10 cpe:/o:linux:linux_k
ernel:4 cpe:/a:synology:diskstation_manager:5.2
Aggressive OS guesses: Linux 5.4 (93%), Linux 2.6.32 or 3.10 (93%), Linux 4.15 - 5.6 (92%), Linux 2.6.32 (91%), Linux 4.4
(91%), Linux 5.0 - 5.3 (90%), Linux 2.6.32 - 2.6.35 (90%), Linux 2.6.32 - 2.6.39 (89%), Linux 5.0 - 5.4 (89%), Linux 3.10
- 4.11 (87%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.99 seconds
```

Notice that *Nmap* has a tough time trying to identify the operating system information.

13. To try and gather more information about the same host regarding its *OS*, make *Nmap* take approximate guesses as to what the *OS* is by using the command below with an included script.

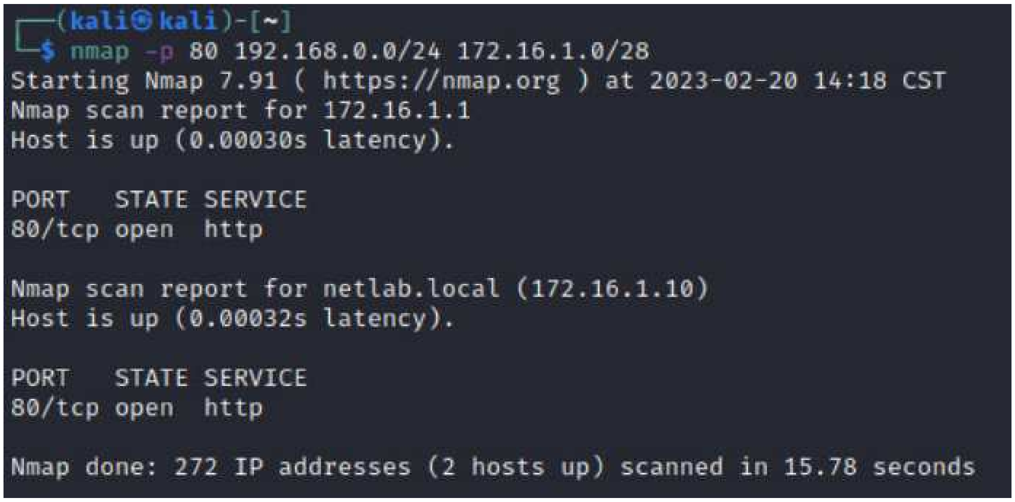
```
kali@kali$ sudo nmap -O --osscan-guess 172.16.1.10
```



```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -O --osscan-guess 172.16.1.10
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:16 CST
Nmap scan report for netlab.local (172.16.1.10)
Host is up (0.00047s latency).
Not shown: 991 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
80/tcp  open  http
110/tcp open  pop3
143/tcp open  imap
443/tcp open  https
587/tcp open  submission
993/tcp open  imaps
995/tcp open  pop3s
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|storage-misc
Running (JUST GUESSING): Linux 4.X|5.X|2.6.X|3.X (93%), Synology DiskStation Manager 5.X (85%)
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:linux:linux_kernel:2.6.32 cpe:/o:linux:linux_kernel
:3.10 cpe:/a:synology:diskstation_manager:5.2
Aggressive OS guesses: Linux 4.15 - 5.6 (93%), Linux 2.6.32 (93%), Linux 4.4 (93%), Linux 5.0 - 5.4 (93%), Linux 5.0 - 5.3
 (92%), Linux 2.6.32 or 3.10 (91%), Linux 5.4 (90%), Linux 2.6.32 - 2.6.35 (90%), Linux 2.6.32 - 2.6.39 (89%), Linux 3.10
- 4.11 (87%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.02 seconds
```

14. Initiate a scan specifically for **port 80,** but this time for all hosts on both networks (*Internal & DMZ*).

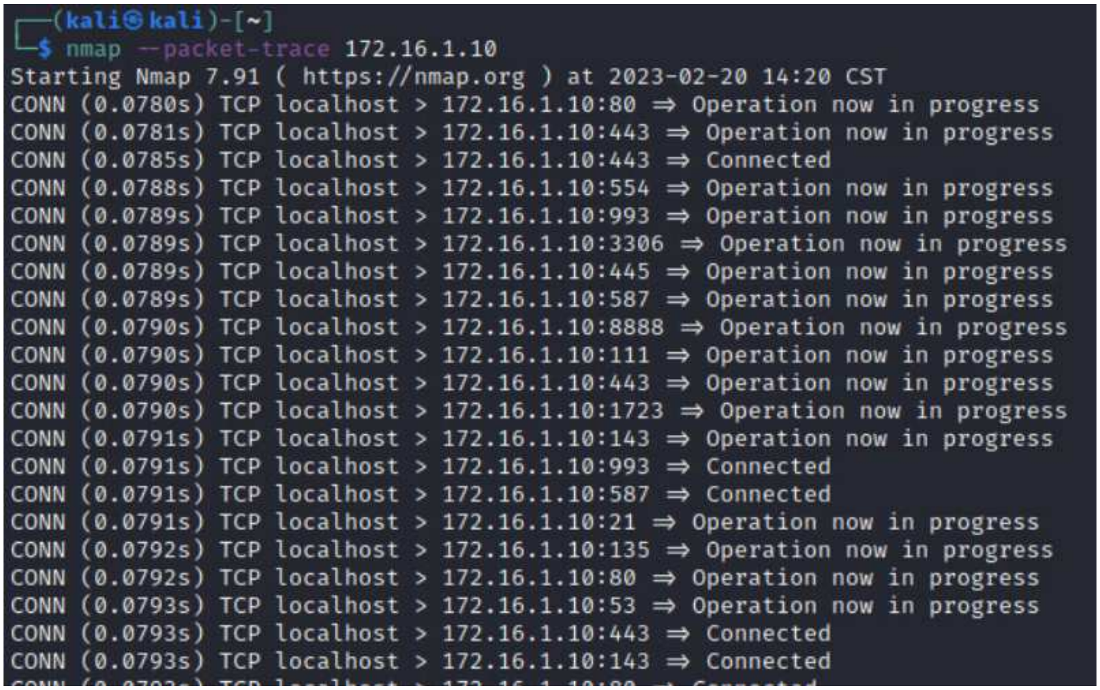
```
kali@kali$ nmap -p 80 192.168.0.0/24 172.16.1.0/28
```



```
┌──(kali㉿kali)-[~]
└─$ nmap -p 80 192.168.0.0/24 172.16.1.0/28
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:18 CST
Nmap scan report for 172.16.1.1
Host is up (0.00030s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap scan report for netlab.local (172.16.1.10)
Host is up (0.00032s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap done: 272 IP addresses (2 hosts up) scanned in 15.78 seconds
```

15. Scan the *UbuntuSRV* system while at the same time displaying all packets being sent and received while initiating the scan.

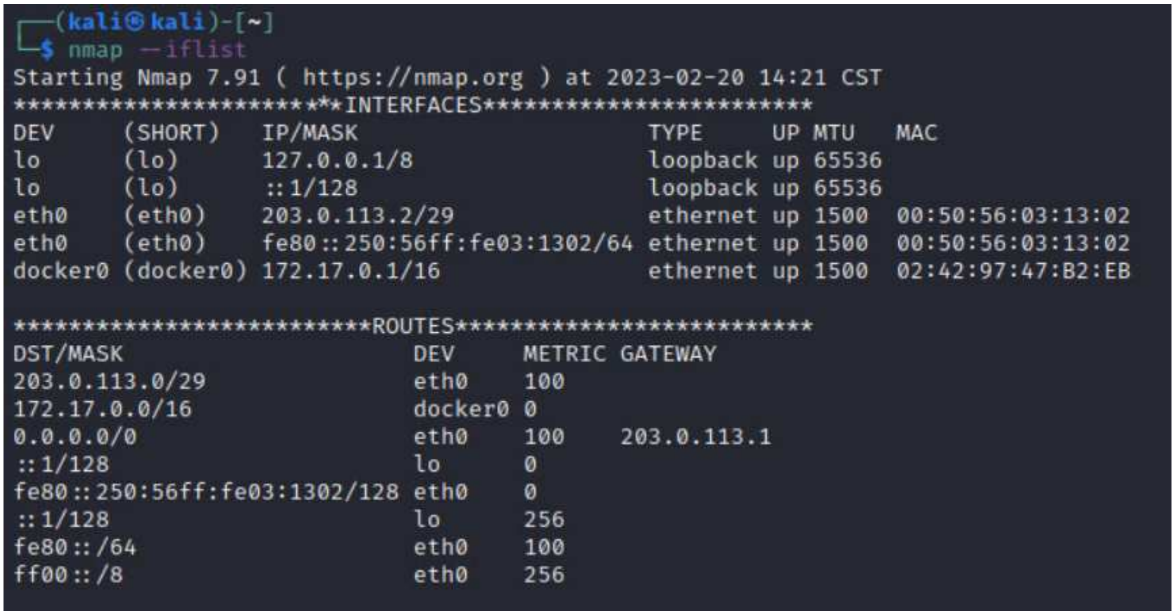
```
kali@kali$ nmap --packet-trace 172.16.1.10
```



```
┌──(kali㉿kali)-[~]
└─$ nmap --packet-trace 172.16.1.10
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:20 CST
CONN (0.0780s) TCP localhost > 172.16.1.10:80 ⇒ Operation now in progress
CONN (0.0781s) TCP localhost > 172.16.1.10:443 ⇒ Operation now in progress
CONN (0.0785s) TCP localhost > 172.16.1.10:443 ⇒ Connected
CONN (0.0788s) TCP localhost > 172.16.1.10:554 ⇒ Operation now in progress
CONN (0.0789s) TCP localhost > 172.16.1.10:993 ⇒ Operation now in progress
CONN (0.0789s) TCP localhost > 172.16.1.10:3306 ⇒ Operation now in progress
CONN (0.0789s) TCP localhost > 172.16.1.10:445 ⇒ Operation now in progress
CONN (0.0789s) TCP localhost > 172.16.1.10:587 ⇒ Operation now in progress
CONN (0.0790s) TCP localhost > 172.16.1.10:8888 ⇒ Operation now in progress
CONN (0.0790s) TCP localhost > 172.16.1.10:111 ⇒ Operation now in progress
CONN (0.0790s) TCP localhost > 172.16.1.10:443 ⇒ Operation now in progress
CONN (0.0790s) TCP localhost > 172.16.1.10:1723 ⇒ Operation now in progress
CONN (0.0791s) TCP localhost > 172.16.1.10:143 ⇒ Operation now in progress
CONN (0.0791s) TCP localhost > 172.16.1.10:993 ⇒ Connected
CONN (0.0791s) TCP localhost > 172.16.1.10:587 ⇒ Connected
CONN (0.0791s) TCP localhost > 172.16.1.10:21 ⇒ Operation now in progress
CONN (0.0792s) TCP localhost > 172.16.1.10:135 ⇒ Operation now in progress
CONN (0.0792s) TCP localhost > 172.16.1.10:80 ⇒ Operation now in progress
CONN (0.0793s) TCP localhost > 172.16.1.10:53 ⇒ Operation now in progress
CONN (0.0793s) TCP localhost > 172.16.1.10:443 ⇒ Connected
CONN (0.0793s) TCP localhost > 172.16.1.10:143 ⇒ Connected
```

16. *Nmap* can also be used to show local host data about which interfaces are up and what the route table looks like. Enter the command below.

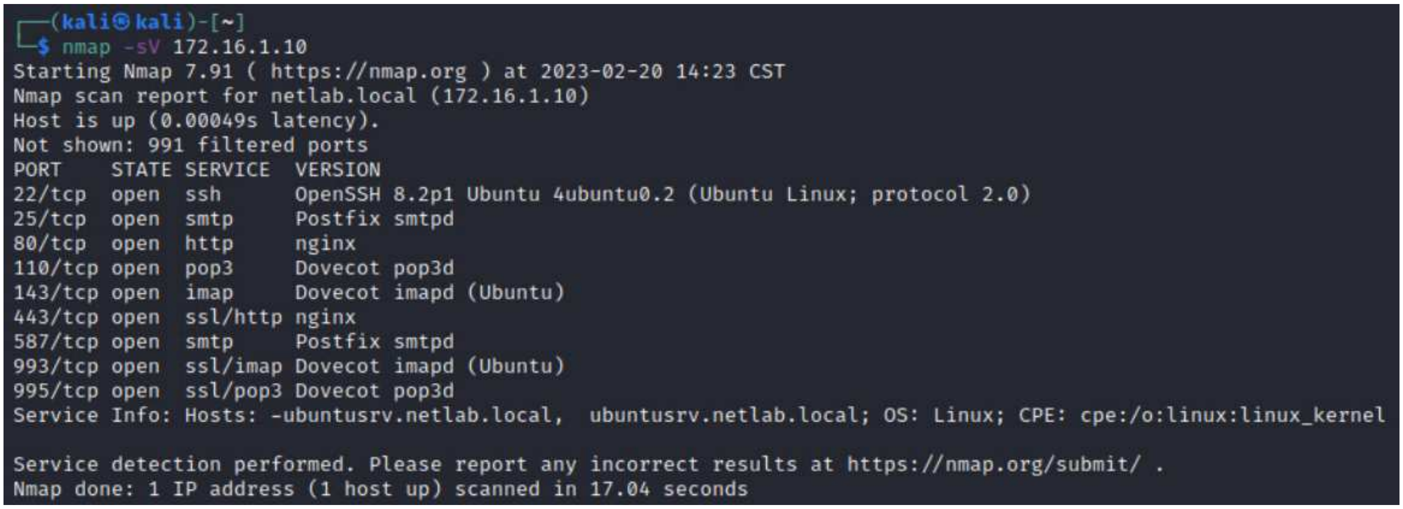
```
kali@kali$ nmap --iflist
```



```
┌──(kali㉿kali)-[~]
└─$ nmap --iflist
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:21 CST
************************INTERFACES**************************
DEV     (SHORT)   IP/MASK                      TYPE     UP MTU   MAC
lo      (lo)      127.0.0.1/8                  loopback up 65536
lo      (lo)      ::1/128                      loopback up 65536
eth0    (eth0)    203.0.113.2/29               ethernet up 1500  00:50:56:03:13:02
eth0    (eth0)    fe80::250:56ff:fe03:1302/64 ethernet up 1500  00:50:56:03:13:02
docker0 (docker0) 172.17.0.1/16                ethernet up 1500  02:42:97:47:B2:EB

**************************ROUTES**************************
DST/MASK                      DEV     METRIC GATEWAY
203.0.113.0/29                eth0    100
172.17.0.0/16                 docker0 0
0.0.0.0/0                     eth0    100    203.0.113.1
::1/128                       lo      0
fe80::250:56ff:fe03:1302/128 eth0    0
::1/128                       lo      256
fe80::/64                     eth0    100
ff00::/8                      eth0    256
```

17. To detect remote services, both *services* and *daemons*, along with their respective version numbers, initiate the *Nmap* command below.

```
kali@kali$ nmap -sV 172.16.1.10
```



```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 172.16.1.10
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-20 14:23 CST
Nmap scan report for netlab.local (172.16.1.10)
Host is up (0.00049s latency).
Not shown: 991 filtered ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
25/tcp  open  smtp     Postfix smtpd
80/tcp  open  http     nginx
110/tcp open  pop3     Dovecot pop3d
143/tcp open  imap     Dovecot imapd (Ubuntu)
443/tcp open  ssl/http nginx
587/tcp open  smtp     Postfix smtpd
993/tcp open  ssl/imap Dovecot imapd (Ubuntu)
995/tcp open  ssl/pop3 Dovecot pop3d
Service Info: Hosts: -ubuntusrv.netlab.local,  ubuntusrv.netlab.local; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.04 seconds
```

18. The lab is now complete; you may end the reservation.