SECURITY+ V4 LAB SERIES

Lab 22: Capturing Network Traffic

Document Version: 2024-01-29

Material in this Lab Aligns to the Following	
CompTIA Security+ (SY0-601) Exam Objectives	3.1: Given a scenario, implement secure protocols 4.1: Given a scenario, use the appropriate tool to assess organizational security
All-In-One CompTIA Security+ Sixth Edition ISBN-13: 978-1260464009 Chapters	17: Secure Protocols 26: Tools/Assess Organizational Security

Contents

Introduction

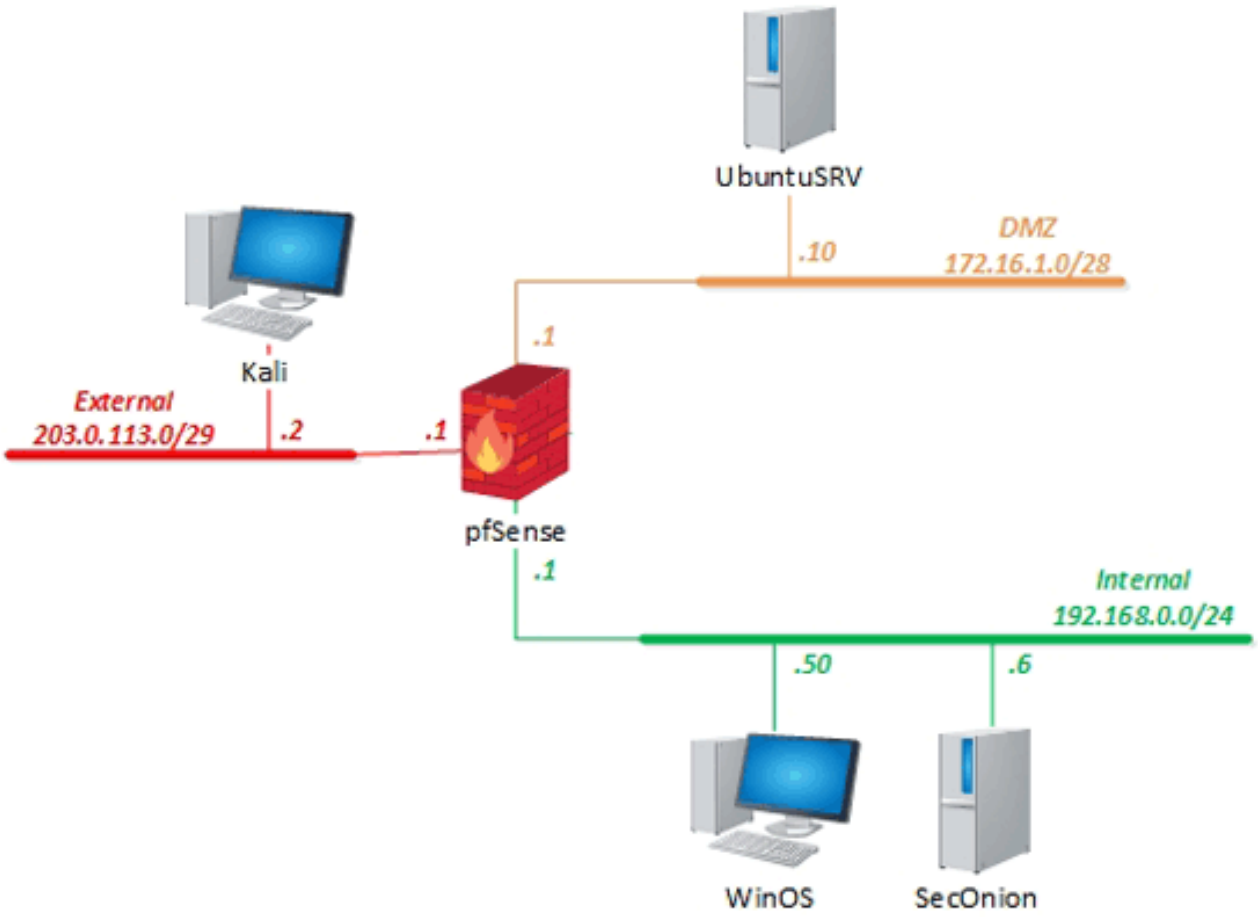
In this lab, you will be conducting network security practices using various tools.

Objective

In this lab, you will perform the following tasks:

- Capture traffic using tcpdump
- Capture traffic using Wireshark
- Capture traffic using dumpcap

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.

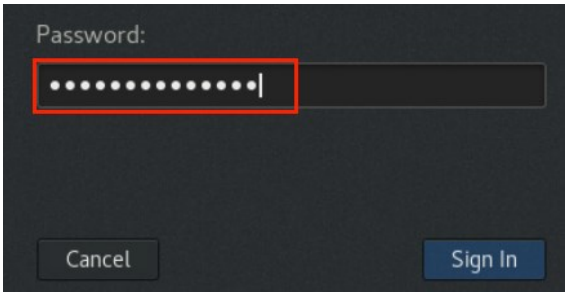
Virtual Machine	IP Address	Account (if needed)	Password (if needed)
Kali	203.0.113.2	kali	kali
pfSense	192.168.0.1	sysadmin	NDGlabpass123!
SecOnion	192.168.0.6	sysadmin	NDGlabpass123!
UbuntuSRV	172.16.1.10	sysadmin	NDGlabpass123!
WinOS	192.168.0.50	Administrator	NDGlabpass123!

1 Using tcpdump to Capture and Analyze Network Traffic

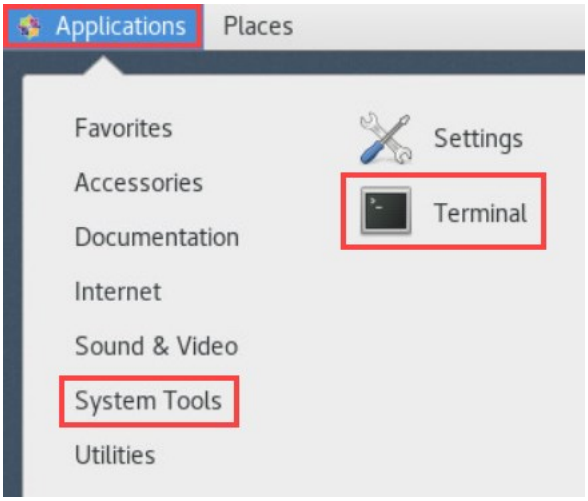
1.1 Using tcpdump to Capture ICMP Traffic

1. Launch the **SecOnion** virtual machine.

2. If you see the *SecOnion* lock screen, click, hold, and drag up to unlock it.
3. On the login screen, type `sysadmin` as the username and `NDGlabpass123!` as the password. Click **Sign In**.



4. Navigate to **Applications > System Tools** and click the **Terminal** icon to launch a new *terminal.*



5. Type the command below, followed by pressing the **Enter** key. If prompted, enter `NDGlabpass123!` for root privileges.

```
sysadmin@seconion ~$ sudo so-status
```

If so-*status* reports back with all modules as *OK*, proceed to the next step. If not, wait a few minutes and try the command again until all show as OK.

6. Type the command below to view all available interfaces on the system. The screenshot below shows two physical interfaces: *ens160* and *ens192*.

```
sysadmin@seconion ~$ ifconfig -a
```

```
[sysadmin@seconion ~]$ ifconfig -a
bond0: flags=5443<UP,BROADCAST,RUNNING,PROMISC,MASTER,MULTICAST>  mtu 1500
        ether 00:50:56:00:00:ff  txqueuelen 1000  (Ethernet)
        RX packets 61  bytes 5324 (5.1 KiB)
        RX errors 0  dropped 12  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.255.0  broadcast 172.17.0.255
        ether 02:42:4d:dd:55:e1  txqueuelen 0  (Ethernet)
        RX packets 50975  bytes 34492067 (32.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 56189  bytes 29732958 (28.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.6  netmask 255.255.255.0  broadcast 192.168.0.255
        ether 00:50:56:92:68:06  txqueuelen 1000  (Ethernet)
        RX packets 20164  bytes 21683942 (20.6 MiB)
        RX errors 0  dropped 14  overruns 0  frame 0
        TX packets 5570  bytes 346477 (338.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens192: flags=2499<UP,BROADCAST,RUNNING,NOARP,PROMISC,SLAVE>  mtu 1500
        ether 00:50:56:00:00:ff  txqueuelen 1000  (Ethernet)
        RX packets 61  bytes 5324 (5.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

7. Issue the command below to identify which flags are configured for each interface.

```
sysadmin@seconion ~$ netstat -i
```

```
[sysadmin@seconion ~]$ netstat -i
Kernel Interface table
Iface             MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
bond0            1500       62      0     12 0           0      0      0      0 BMPmRU
docker0          1500    72572      0      0 0       78410      0      0      0 BMRU
ens160           1500    20312      0     14 0        5707      0      0      0 BMRU
ens192           1500       62      0      0 0           0      0      0      0 BPOsRU
lo              65536    20380      0      0 0       20380      0      0      0 LRU
veth0123d75      1500     1308      0      0 0        1517      0      0      0 BMRU
veth025c70b      1500     3717      0      0 0        6830      0      0      0 BMRU
veth031559e      1500      191      0      0 0         175      0      0      0 BMRU
veth0a892f7      1500        0      0      0 0          38      0      0      0 BMRU
veth1617a4d      1500       66      0      0 0          99      0      0      0 BMRU
veth2606fd2      1500     5115      0      0 0        2570      0      0      0 BMRU
veth37d5a63      1500     1662      0      0 0        1557      0      0      0 BMRU
veth3ae97f6      1500     5141      0      0 0        2584      0      0      0 BMRU
veth4c05d14      1500     7814      0      0 0        8430      0      0      0 BMRU
veth54c81fc      1500       83      0      0 0         106      0      0      0 BMRU
veth595c088      1500     5138      0      0 0       10274      0      0      0 BMRU
veth6d5163f      1500        8      0      0 0          23      0      0      0 BMRU
veth77769a5      1500     7923      0      0 0        4883      0      0      0 BMRU
veth81fdb50      1500      706      0      0 0         620      0      0      0 BMRU
veth8438bae      1500        0      0      0 0           8      0      0      0 BMRU
veth87cade6      1500       39      0      0 0          31      0      0      0 BMRU
veth905fa1c      1500      204      0      0 0        3820      0      0      0 BMRU
veth981dadf      1500        0      0      0 0          24      0      0      0 BMRU
veth9aeadcd      1500     4122      0      0 0        5154      0      0      0 BMRU
vetha486901      1500      272      0      0 0         295      0      0      0 BMRU
vetha0c08fd      1500      629      0      0 0        1109      0      0      0 BMRU
vetha2e178e      1500      134      0      0 0         144      0      0      0 BMRU
vetha4a1e54      1500    22189      0      0 0       21472      0      0      0 BMRU
vethafab5ef      1500      985      0      0 0        1710      0      0      0 BMRU
vethb562b27      1500     4752      0      0 0        4954      0      0      0 BMRU
vethc0467f2      1500        0      0      0 0          38      0      0      0 BMRU
vethce59ed5      1500      119      0      0 0         126      0      0      0 BMRU
vethd61f6ab      1500       72      0      0 0          72      0      0      0 BMRU
vethe934013      1500      181      0      0 0         174      0      0      0 BMRU
[sysadmin@seconion ~]$
```

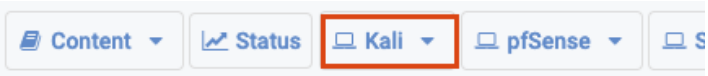
Notice how *BMPRU* is set for the interfaces under the *Flg* column. Notice that *BMPORU* is set for both *eth1* and *eth2*. For a quick overview: *B* flag is for broadcast, *M* flag is for multicast, *P* flag is for promiscuous mode, *O* flag is for no *ARP* (*Address Resolution Protocol*) requests, *R* flag is for running and *U* flag is for up. Also, notice that *LRU* is set for *lo*; the *L* flag means that the specified interface is a loopback device.

8. To familiarize yourself with the *tcpdump* utility, type the following command to view several available options for *tcpdump.*

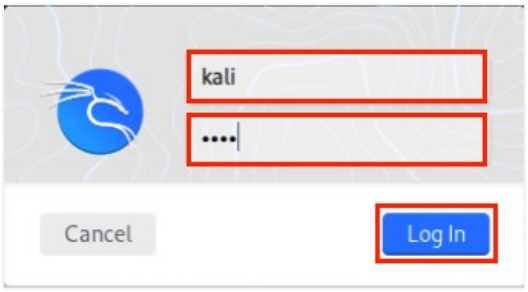
```
sysadmin@seconion ~$ tcpdump --help
```

```
[sysadmin@seconion ~]$ tcpdump --help
tcpdump version 4.9.2
libpcap version 1.5.3
OpenSSL 1.0.2k-fips  26 Jan 2017
Usage: tcpdump [-aAbdDefhHIJKlLnNOpqStuUvxX#] [ -B size ] [ -c count ]
                [ -C file_size ] [ -E algo:secret ] [ -F file ] [ -G seconds ]
                [ -i interface ] [ -j tstamptype ] [ -M secret ] [ --number ]
                [ -Q|-P in|out|inout ]
                [ -r file ] [ -s snaplen ] [ --time-stamp-precision precision ]
                [ --immediate-mode ] [ -T type ] [ --version ] [ -V file ]
                [ -w file ] [ -W filecount ] [ -y datalinktype ] [ -z postrotate-command ]
                [ -Z user ] [ expression ]
[sysadmin@seconion ~]$
```

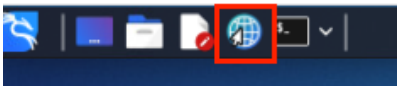
9. Launch the *Kali* virtual machine to access the graphical login screen.



10. Log in as **kali** with **kali** as the password.

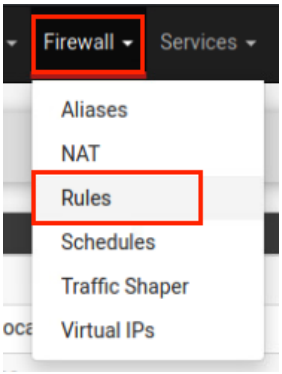


11. Click the *Firefox* icon located in the top menu bar.

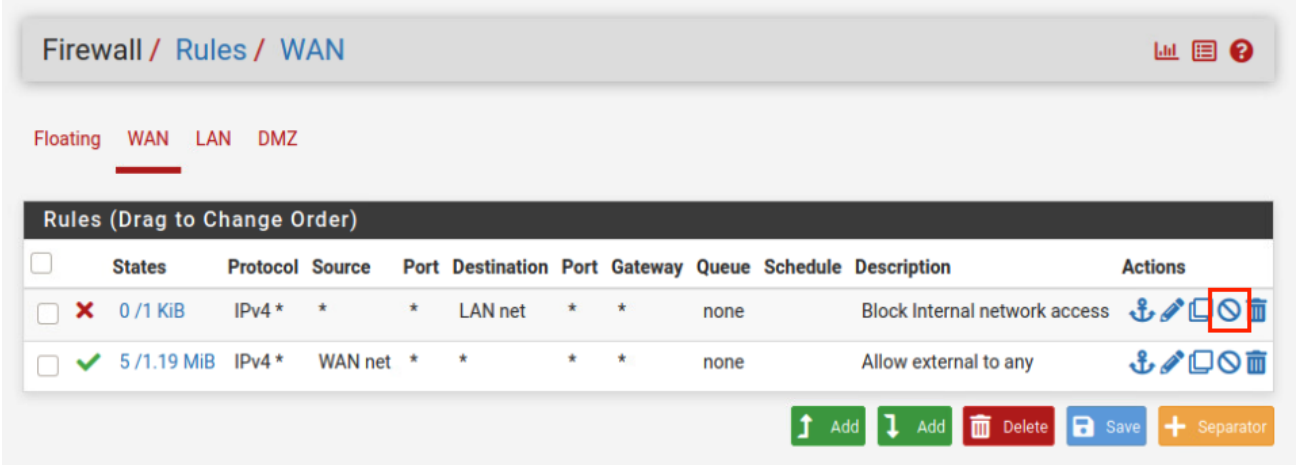


12. Once *Firefox* opens, go to *pfSense* on address **http://203.0.113.1,** then sign in using *username* **sysadmin** and *password* **NDGlabpass123!.**

13. Go to **Firewall > Rules**.

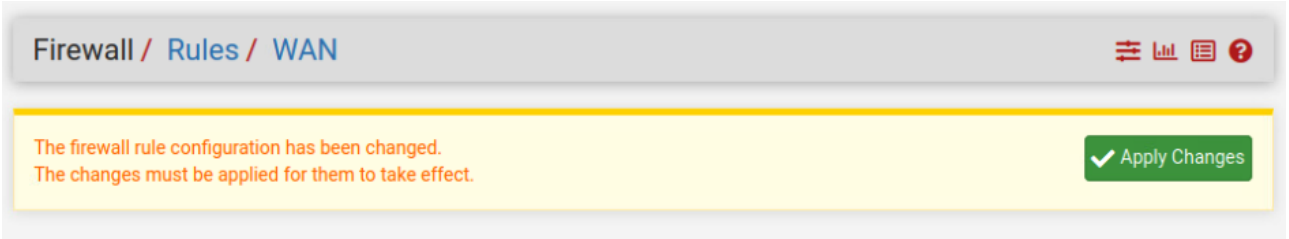


14. On the *Rules* page, make sure you are on the *WAN* category. Then, click the **disable** button to disable the rule where the *Description* says **Block Internal network access**.



In a real-world scenario, you do not want to disable this rule on your firewall. The outside network should not be allowed to have direct access to the intranet.

15. Click **Apply Changes**, to make the change effective.



16. Close *Firefox*, click on the **Terminal** icon located in the top menu bar.

17. Type the following command to initiate a continuous ping to the *WinOS* system. Leave the pings running in the background and proceed to the next step.

```
kali@kali$ ping 192.168.0.50
```

```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.0.50
PING 192.168.0.50 (192.168.0.50) 56(84) bytes of data.
64 bytes from 192.168.0.50: icmp_seq=1 ttl=127 time=0.857 ms
64 bytes from 192.168.0.50: icmp_seq=2 ttl=127 time=0.733 ms
64 bytes from 192.168.0.50: icmp_seq=3 ttl=127 time=0.683 ms
64 bytes from 192.168.0.50: icmp_seq=4 ttl=127 time=0.811 ms
64 bytes from 192.168.0.50: icmp_seq=5 ttl=127 time=0.668 ms
64 bytes from 192.168.0.50: icmp_seq=6 ttl=127 time=0.659 ms
64 bytes from 192.168.0.50: icmp_seq=7 ttl=127 time=0.716 ms
64 bytes from 192.168.0.50: icmp_seq=8 ttl=127 time=0.686 ms
64 bytes from 192.168.0.50: icmp_seq=9 ttl=127 time=0.712 ms
64 bytes from 192.168.0.50: icmp_seq=10 ttl=127 time=0.746 ms
64 bytes from 192.168.0.50: icmp_seq=11 ttl=127 time=0.681 ms
```

18. Switch back to the *SecOnion* system. In the previous *Terminal*, run **tcpdump** on the *internal network* by entering the command below. If prompted with a password, enter **NDGlabpass123!**.

```
sysadmin@seconion ~$ sudo tcpdump -i ens192 icmp
```

19. Notice the output that tcpdump provides: *HH:MM:SS.mmmmmm IP src > dst: ptype, id, seq, len*. Also, take note that for each echo request, there is a reply.

```
[sysadmin@seconion ~]$ sudo tcpdump -i ens192 icmp
[sudo] password for sysadmin:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens192, link-type EN10MB (Ethernet), capture size 262144 bytes
17:48:45.736588 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 41, length 64
17:48:45.736874 IP WinOS.netlab.local > pfsense.netlab.local: ICMP echo reply, id 50097, seq 41, length 64
17:48:46.760343 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 42, length 64
17:48:46.760707 IP WinOS.netlab.local > pfsense.netlab.local: ICMP echo reply, id 50097, seq 42, length 64
17:48:47.387749 IP pfsense.netlab.local > WinOS.netlab.local: ICMP pfsense.netlab.local udp port 60921 unreachable, length 78
17:48:47.387766 IP pfsense.netlab.local > WinOS.netlab.local: ICMP pfsense.netlab.local udp port 60921 unreachable, length 78
17:48:47.784361 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 43, length 64
17:48:47.784711 IP WinOS.netlab.local > pfsense.netlab.local: ICMP echo reply, id 50097, seq 43, length 64
17:48:48.292364 IP seconion.netlab.local > WinOS.netlab.local: ICMP seconion.netlab.local udp port 39256 unreachable, length 77
17:48:48.808349 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 44, length 64
17:48:48.808666 IP WinOS.netlab.local > pfsense.netlab.local: ICMP echo reply, id 50097, seq 44, length 64
17:48:49.206377 IP pfsense.netlab.local > WinOS.netlab.local: ICMP pfsense.netlab.local udp port 57883 unreachable, length 70
17:48:49.206386 IP pfsense.netlab.local > WinOS.netlab.local: ICMP pfsense.netlab.local udp port 57883 unreachable, length 70
17:48:49.832296 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 45, length 64
17:48:49.832571 IP WinOS.netlab.local > pfsense.netlab.local: ICMP echo reply, id 50097, seq 45, length 64
17:48:50.856258 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 46, length 64
17:48:50.856477 IP WinOS.netlab.local > pfsense.netlab.local: ICMP echo reply, id 50097, seq 46, length 64
17:48:51.880332 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 47, length 64
17:48:51.880648 IP WinOS.netlab.local > pfsense.netlab.local: ICMP echo reply, id 50097, seq 47, length 64
17:48:51.931955 IP seconion.netlab.local > WinOS.netlab.local: ICMP seconion.netlab.local udp port 47941 unreachable, length 76
17:48:51.931974 IP seconion.netlab.local > WinOS.netlab.local: ICMP seconion.netlab.local udp port 36479 unreachable, length 76
17:48:51.931979 IP seconion.netlab.local > WinOS.netlab.local: ICMP seconion.netlab.local udp port 47941 unreachable, length 76
17:48:51.931984 IP seconion.netlab.local > WinOS.netlab.local: ICMP seconion.netlab.local udp port 36479 unreachable, length 76
17:48:52.904251 IP pfsense.netlab.local > WinOS.netlab.local: ICMP echo request, id 50097, seq 48, length 64
```

HH:MM:SS.mmmmmm	**Timestamp in hours, minutes, seconds, and microseconds**
IP	Internet Protocol
src > dst	Source and destination IP addresses
ptype	Packet type
id, seq, len	IP headers; identification, protocol (1=ICMP), total length

20. After a minute, press CTRL+C to stop *tcpdump* from running and discontinue the network capture.
21. From an administrator's standpoint, we may want to save the output from a *tcpdump capture* and save it automatically into a compatible file to view later with a program such as *Wireshark*. Initiate the command below to capture traffic on the **192.168.1.0/24** network and send it to a file. If prompted with a password, enter `NDGlabpass123!`.

```
sysadmin@seconion ~$ sudo tcpdump icmp -i ens192 -s 0 -w netcapture1.pcap -c 100
```

The following table lists details of the options used with the *tcpdump* command

icmp	**Captures only ICMP packets (works for tcp, udp, and icmp)**
-i eth0	Use interface zero
-s 0	Disables default packet size, date and time format
-w	Write to a captured file, instead of displaying to the screen
-c	Split the captures into files of this size

22. Wait for about 1-2 minutes until all 100 packets are captured.

```
[sysadmin@seconion ~]$ sudo tcpdump icmp -i ens192 -s 0 -w netcapture1.pcap -c 100
tcpdump: listening on ens192, link-type EN10MB (Ethernet), capture size 262144 bytes
100 packets captured
100 packets received by filter
0 packets dropped by kernel
```

23. To view the captured file in a graphical user interface like *Wireshark*, enter the command below in the *SecOnion terminal*. If prompted with a password, enter `NDGlabpass123!`.

```
sysadmin@seconion ~$ sudo wireshark netcapture1.pcap
```

24. Notice the traffic listed that takes place on the *192.168.1.0/24* network.

No.	Time	Source	Destination	Protoc	Lengt	Info
1	0	192.168.0.1	192.168.0.50	ICMP	98	Echo (ping) request id=
2	0	192.168.0.50	192.168.0.1	ICMP	98	Echo (ping) reply id=
3	1	192.168.0.1	192.168.0.50	ICMP	98	Echo (ping) request id=
4	1	192.168.0.50	192.168.0.1	ICMP	98	Echo (ping) reply id=
5	2	192.168.0.1	192.168.0.50	ICMP	98	Echo (ping) request id=
6	2	192.168.0.50	192.168.0.1	ICMP	98	Echo (ping) reply id=

25. Close **Wireshark.**

26. Switch to the **Kali** machine and press **CTRL+C** to stop the continuous pings.

```
64 bytes from 192.168.0.50: icmp_seq=1178 ttl=127 time=1.12 ms
64 bytes from 192.168.0.50: icmp_seq=1179 ttl=127 time=1.21 ms
^C
```

1.2 Using tcpdump to Capture ARP Traffic

1. Change focus to the **SecOnion** system.
2. In a *Terminal* window, enter the *ARP* command below and examine the results.

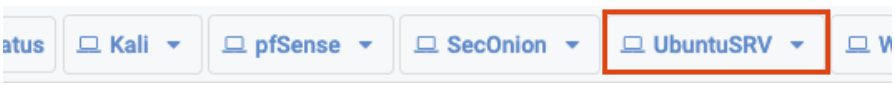
```
sysadmin@seconion ~$ arp -n
```

```
[sysadmin@seconion ~]$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.17.0.16              ether   02:42:ac:11:00:10   C                     docker0
172.17.0.3               ether   02:42:ac:11:00:03   C                     docker0
172.17.0.28              ether   02:42:ac:11:00:1c   C                     docker0
172.17.0.15              ether   02:42:ac:11:00:0f   C                     docker0
172.17.0.25              ether   02:42:ac:11:00:19   C                     docker0
172.17.0.18              ether   02:42:ac:11:00:12   C                     docker0
172.17.0.5               ether   02:42:ac:11:00:05   C                     docker0
172.17.0.30              ether   02:42:ac:11:00:1e   C                     docker0
172.17.0.27              ether   02:42:ac:11:00:1b   C                     docker0
172.17.0.10              ether   02:42:ac:11:00:0a   C                     docker0
172.17.0.20              ether   02:42:ac:11:00:14   C                     docker0
172.17.0.7               ether   02:42:ac:11:00:07   C                     docker0
172.17.0.17              ether   02:42:ac:11:00:11   C                     docker0
172.17.0.29              ether   02:42:ac:11:00:1d   C                     docker0
172.17.0.12              ether   02:42:ac:11:00:0c   C                     docker0
172.17.0.22              ether   02:42:ac:11:00:16   C                     docker0
192.168.0.50             ether   00:50:56:92:68:50   C                     ens160
172.17.0.9               ether   02:42:ac:11:00:09   C                     docker0
192.168.0.1              ether   00:50:56:92:68:01   C                     ens160
172.17.0.19              ether   02:42:ac:11:00:13   C                     docker0
172.17.0.14              ether   02:42:ac:11:00:0e   C                     docker0
172.17.0.24              ether   02:42:ac:11:00:18   C                     docker0
172.17.0.11              ether   02:42:ac:11:00:0b   C                     docker0
172.17.0.21              ether   02:42:ac:11:00:15   C                     docker0
172.17.0.26              ether   02:42:ac:11:00:1a   C                     docker0
172.17.0.13              ether   02:42:ac:11:00:0d   C                     docker0
172.17.0.6               ether   02:42:ac:11:00:06   C                     docker0
```

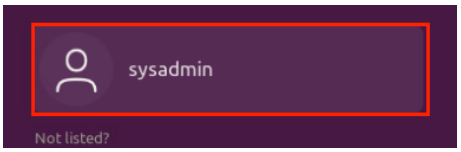
3. Enter the command below to capture *ARP* packets. If prompted for a password, enter `NDGlabpass123!`.

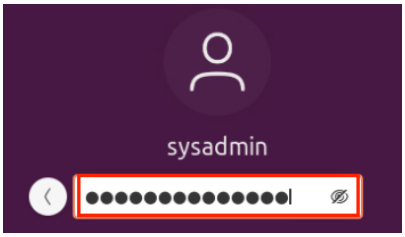
```
sysadmin@seconion ~$ sudo tcpdump -i ens160 -nn -e arp
```

4. Launch the **UbuntuSRV** virtual machine to access the graphical login screen.

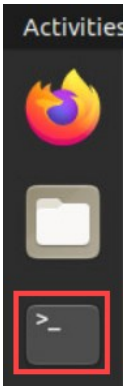


5. Log in as **sysadmin** with **NDGlabpass123!** as the password.





6. Open a *Terminal* window by clicking on the **terminal** icon located in the left menu pane.



7. Type the *ping* command below.

```
sysadmin@ubuntusrv:~$ ping -c4 192.168.0.6
```

```
sysadmin@ubuntusrv:~$ ping -c4 192.168.0.6
PING 192.168.0.6 (192.168.0.6) 56(84) bytes of data.
64 bytes from 192.168.0.6: icmp_seq=1 ttl=63 time=1.27 ms
64 bytes from 192.168.0.6: icmp_seq=2 ttl=63 time=0.876 ms
64 bytes from 192.168.0.6: icmp_seq=3 ttl=63 time=0.942 ms
64 bytes from 192.168.0.6: icmp_seq=4 ttl=63 time=0.870 ms

--- 192.168.0.6 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.870/0.989/1.270/0.164 ms
```

8. Switch back to **SecOnion** and press **CTRL+C** to stop the *tcpdump* capture. Notice the *ARP* output: *HH:MM:SS:mmmmmm srcMAC > dstMAC: ptype, len, request/response, length.*

```
[sysadmin@seconion ~]$ sudo tcpdump -i ens192 -nn -e arp
[sudo] password for sysadmin:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens192, link-type EN10MB (Ethernet), capture size 262144 bytes
17:45:21.122014 00:0c:29:08:dc:72 > 00:50:56:92:68:01, ethertype ARP (0x0806), l
ength 60: Request who-has 192.168.0.1 tell 192.168.0.254, length 46
17:45:21.122279 00:50:56:92:68:01 > 00:0c:29:08:dc:72, ethertype ARP (0x0806), l
ength 60: Reply 192.168.0.1 is-at 00:50:56:92:68:01, length 46
17:45:29.532991 00:50:56:92:68:06 > 00:50:56:92:68:50, ethertype ARP (0x0806), l
ength 60: Request who-has 192.168.0.50 tell 192.168.0.6, length 46
17:45:29.533346 00:50:56:92:68:50 > 00:50:56:92:68:06, ethertype ARP (0x0806), l
ength 60: Reply 192.168.0.50 is-at 00:50:56:92:68:50, length 46
17:45:30.845001 00:50:56:92:68:06 > 00:50:56:92:68:01, ethertype ARP (0x0806), l
ength 60: Request who-has 192.168.0.1 tell 192.168.0.6, length 46
17:45:30.845526 00:50:56:92:68:01 > 00:50:56:92:68:06, ethertype ARP (0x0806), l
ength 60: Reply 192.168.0.1 is-at 00:50:56:92:68:01, length 46
17:45:44.562006 00:0c:29:08:dc:72 > 00:50:56:92:68:01, ethertype ARP (0x0806), l
ength 60: Request who-has 192.168.0.1 tell 192.168.0.254, length 46
17:45:44.562355 00:50:56:92:68:01 > 00:0c:29:08:dc:72, ethertype ARP (0x0806), l
ength 60: Reply 192.168.0.1 is-at 00:50:56:92:68:01, length 46
```

9. Enter the command below to display the ARP table. Notice the ARP entry for the IP address ***192.168.0.1*** is showing in the orange square.

```
sysadmin@seconion ~$ arp -n
```

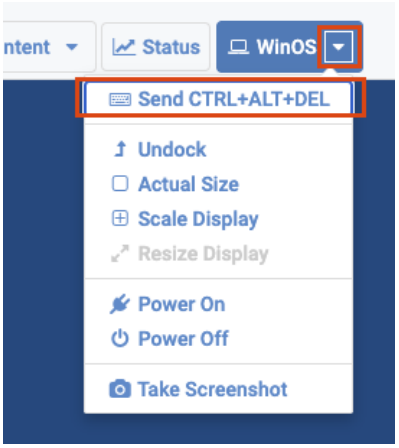
```
[sysadmin@seconion ~]$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.17.0.16              ether   02:42:ac:11:00:10   C                     docker0
172.17.0.3               ether   02:42:ac:11:00:03   C                     docker0
172.17.0.28              ether   02:42:ac:11:00:1c   C                     docker0
172.17.0.15              ether   02:42:ac:11:00:0f   C                     docker0
172.17.0.25              ether   02:42:ac:11:00:19   C                     docker0
172.17.0.18              ether   02:42:ac:11:00:12   C                     docker0
172.17.0.5               ether   02:42:ac:11:00:05   C                     docker0
172.17.0.30              ether   02:42:ac:11:00:1e   C                     docker0
172.17.0.27              ether   02:42:ac:11:00:1b   C                     docker0
172.17.0.10              ether   02:42:ac:11:00:0a   C                     docker0
172.17.0.20              ether   02:42:ac:11:00:14   C                     docker0
172.17.0.7               ether   02:42:ac:11:00:07   C                     docker0
172.17.0.17              ether   02:42:ac:11:00:11   C                     docker0
172.17.0.29              ether   02:42:ac:11:00:1d   C                     docker0
172.17.0.12              ether   02:42:ac:11:00:0c   C                     docker0
172.17.0.22              ether   02:42:ac:11:00:16   C                     docker0
192.168.0.50             ether   00:50:56:92:68:50   C                     ens160
172.17.0.9               ether   02:42:ac:11:00:09   C                     docker0
192.168.0.1              ether   00:50:56:92:68:01   C                     ens160
172.17.0.19              ether   02:42:ac:11:00:13   C                     docker0
172.17.0.14              ether   02:42:ac:11:00:0e   C                     docker0
172.17.0.24              ether   02:42:ac:11:00:18   C                     docker0
172.17.0.11              ether   02:42:ac:11:00:0b   C                     docker0
172.17.0.21              ether   02:42:ac:11:00:15   C                     docker0
172.17.0.26              ether   02:42:ac:11:00:1a   C                     docker0
172.17.0.13              ether   02:42:ac:11:00:0d   C                     docker0
172.17.0.6               ether   02:42:ac:11:00:06   C                     docker0
```

2 Using Wireshark to Capture & Analyze Network Traffic

2.1 Using Wireshark to Capture FTP Traffic

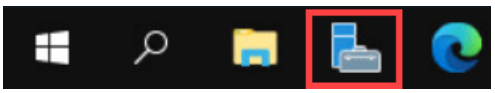
2.1.1 Setup FTP Server

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

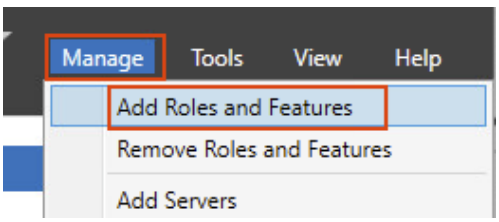


2. Log in as `Administrator` using the password `NDGlabpass123!`.

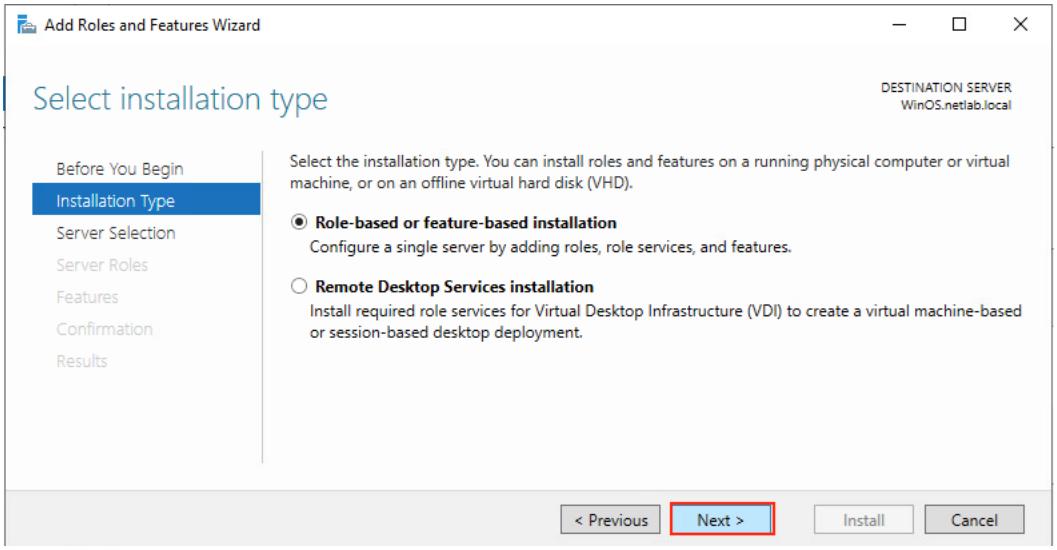
3. Once logged in, click the **Server Manager** icon to launch it.



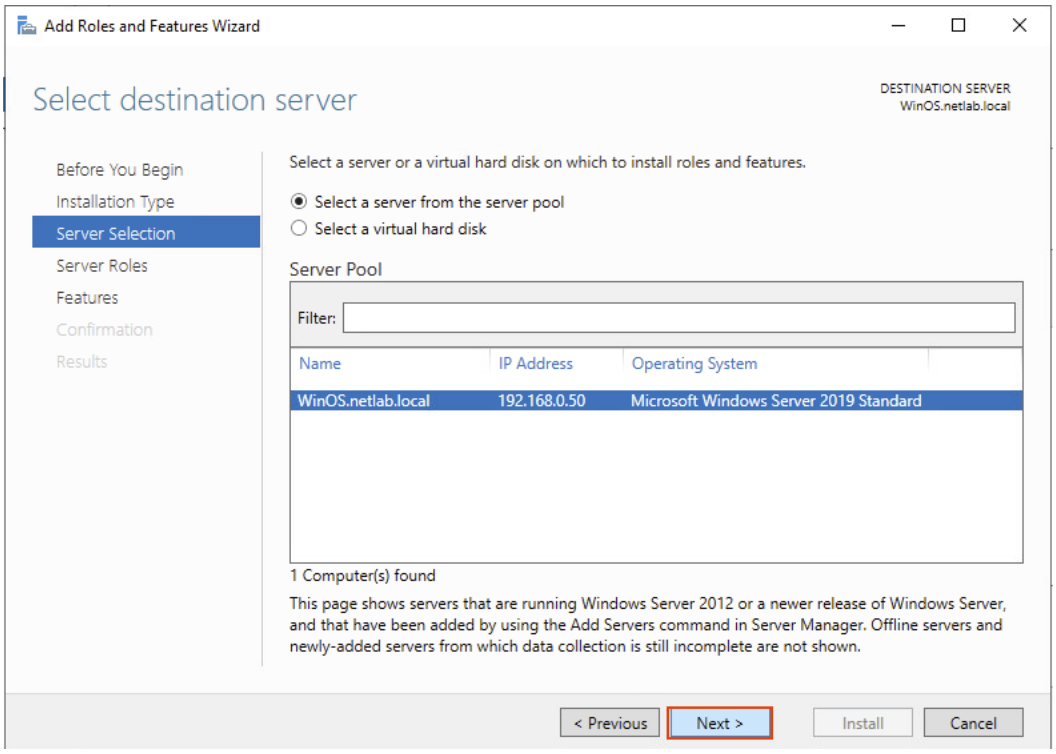
4. In the *Server Manager* window, click **Manage** in the upper-right corner. We will install FTP service to the server.



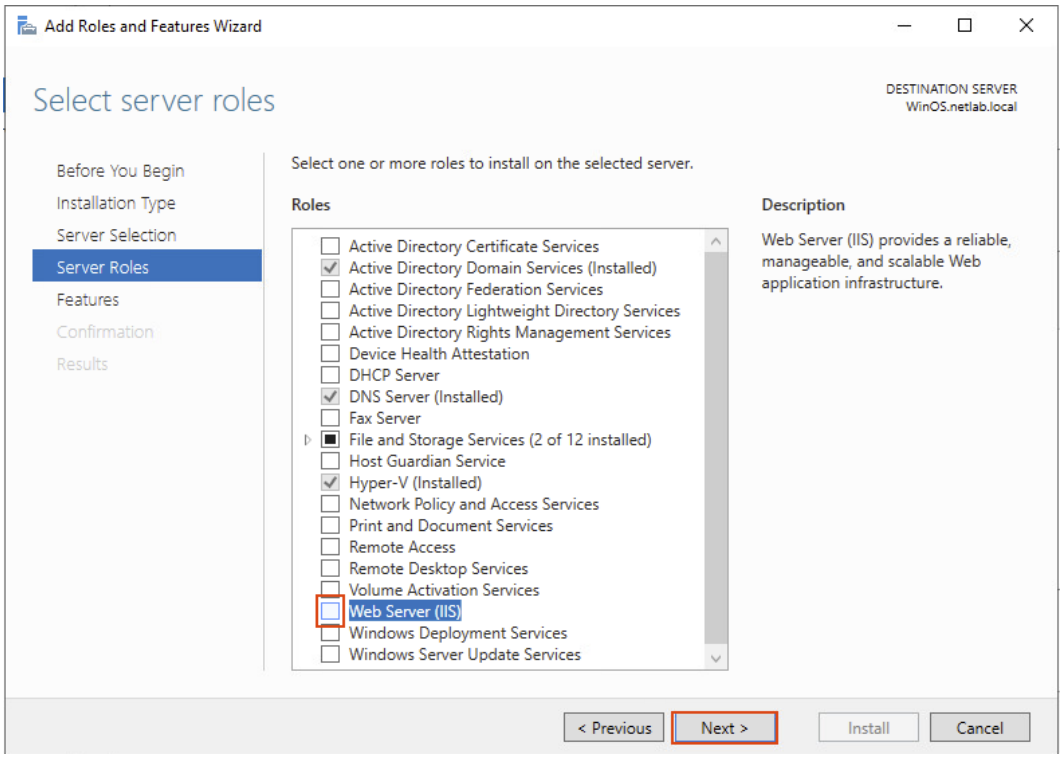
5. In the pop-up window, click **Next**.



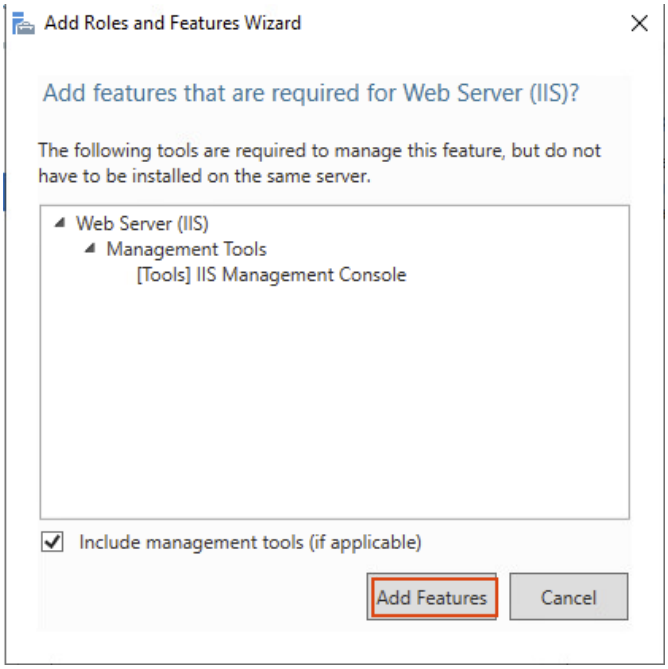
6. On the *Select destination server* step, click **Next.**



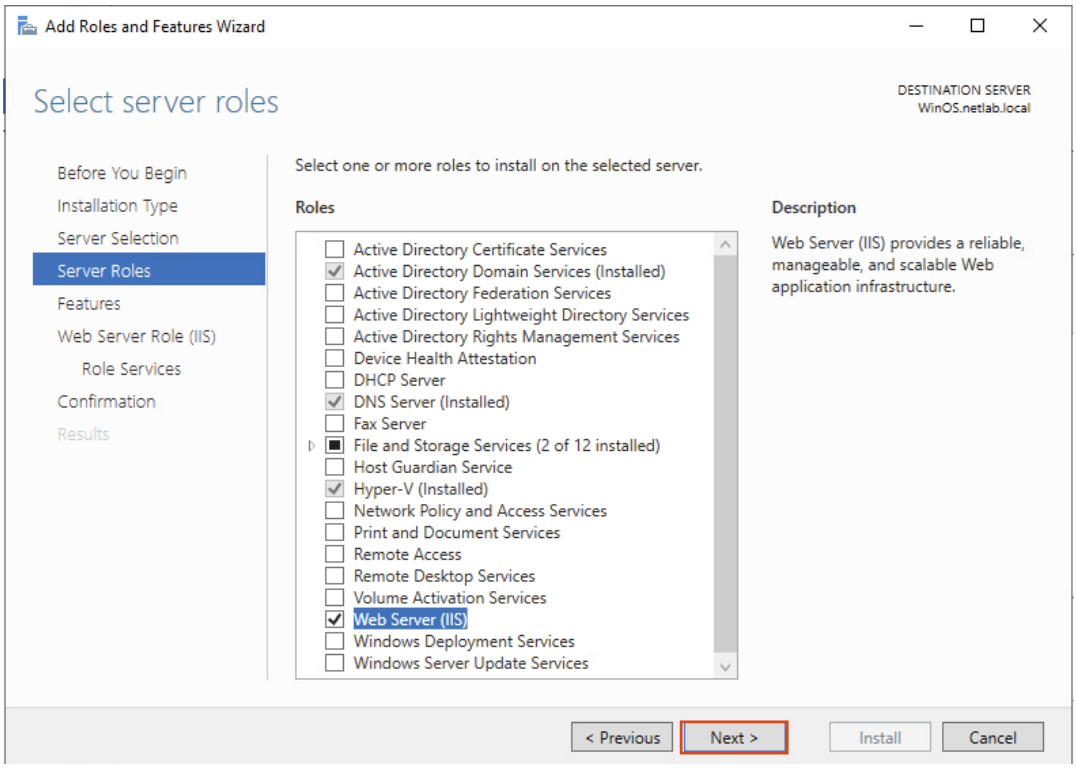
7. On the *Select server roles* screen, scroll down to check the box in front of **Web Server(IIS).**



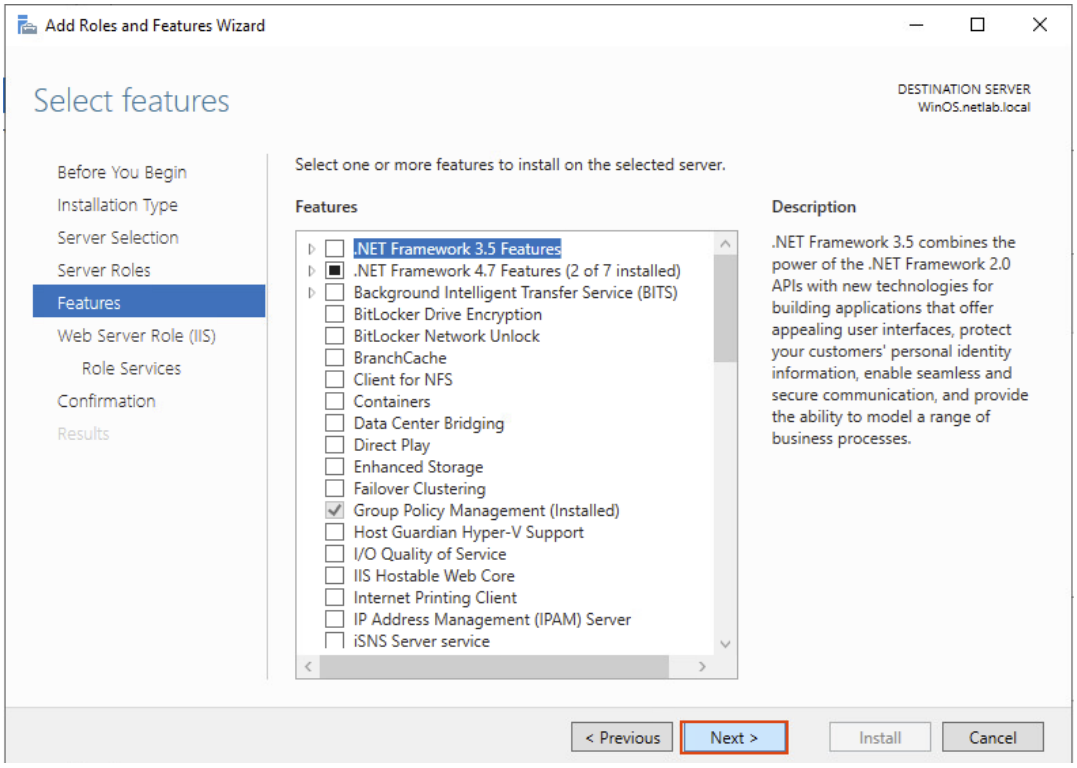
8. In the new pop-up window, click **Add Features.**



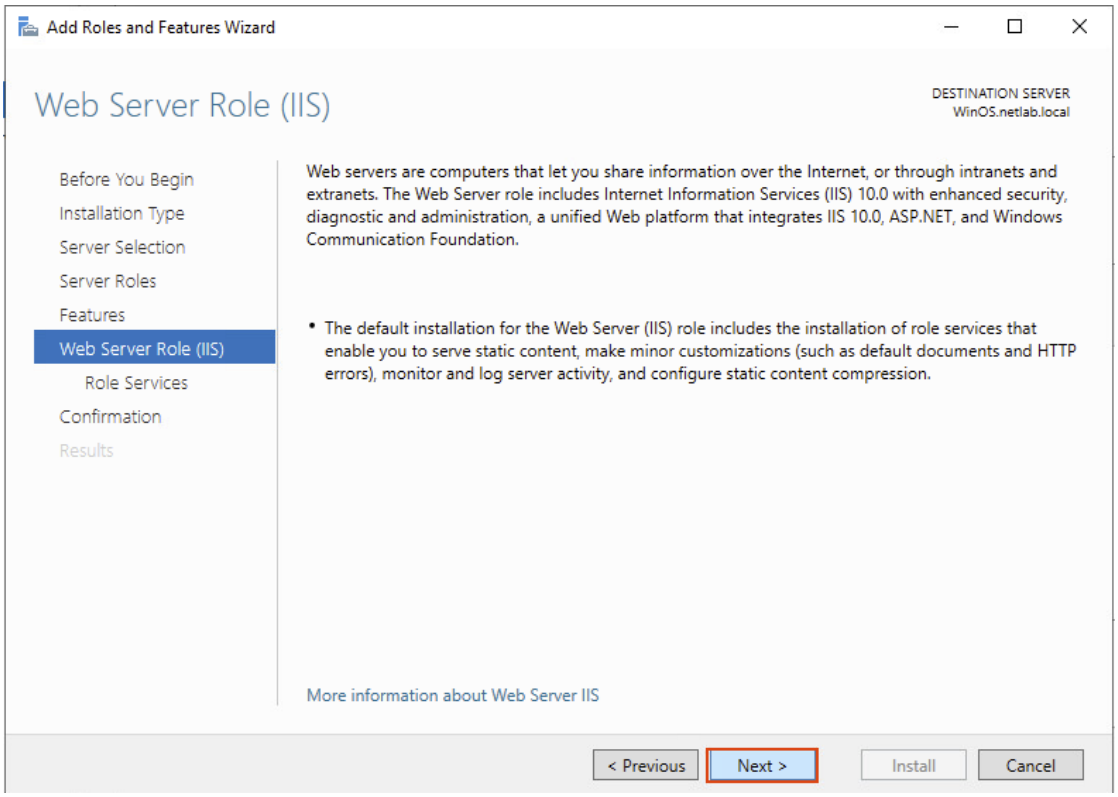
9. When brought back to the *Select server roles* window, click **Next.**



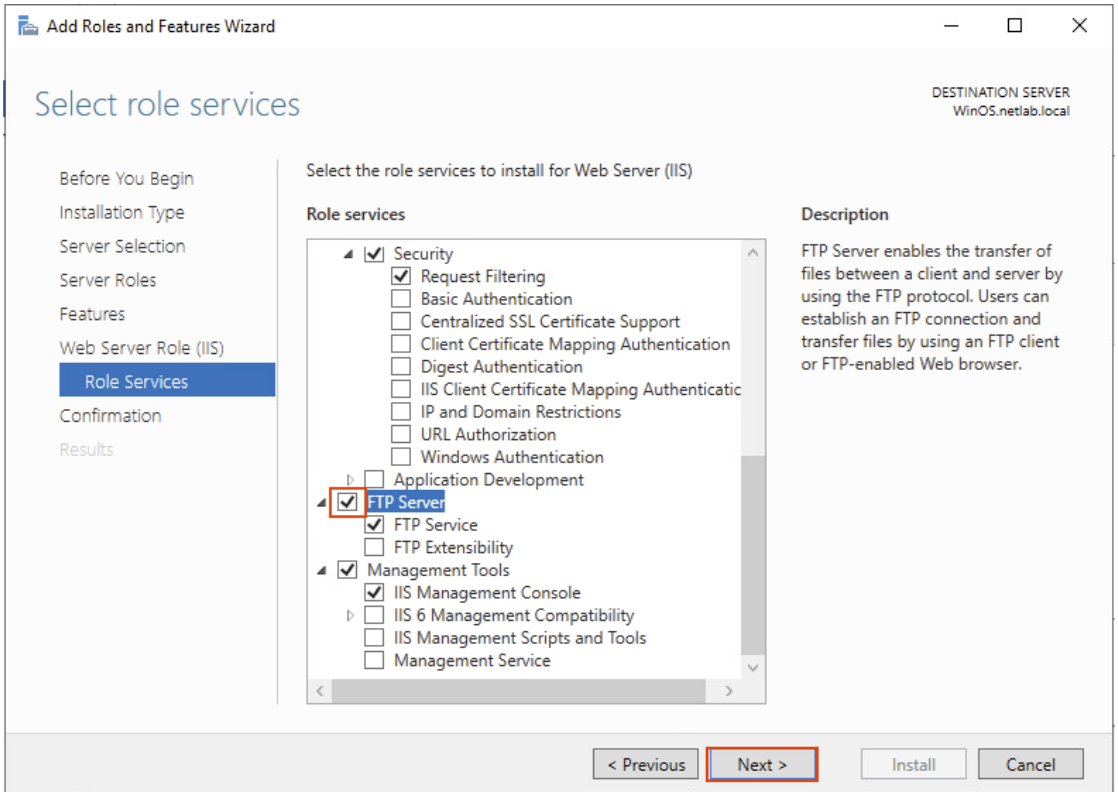
10. On the *Select features* window, click **Next.**



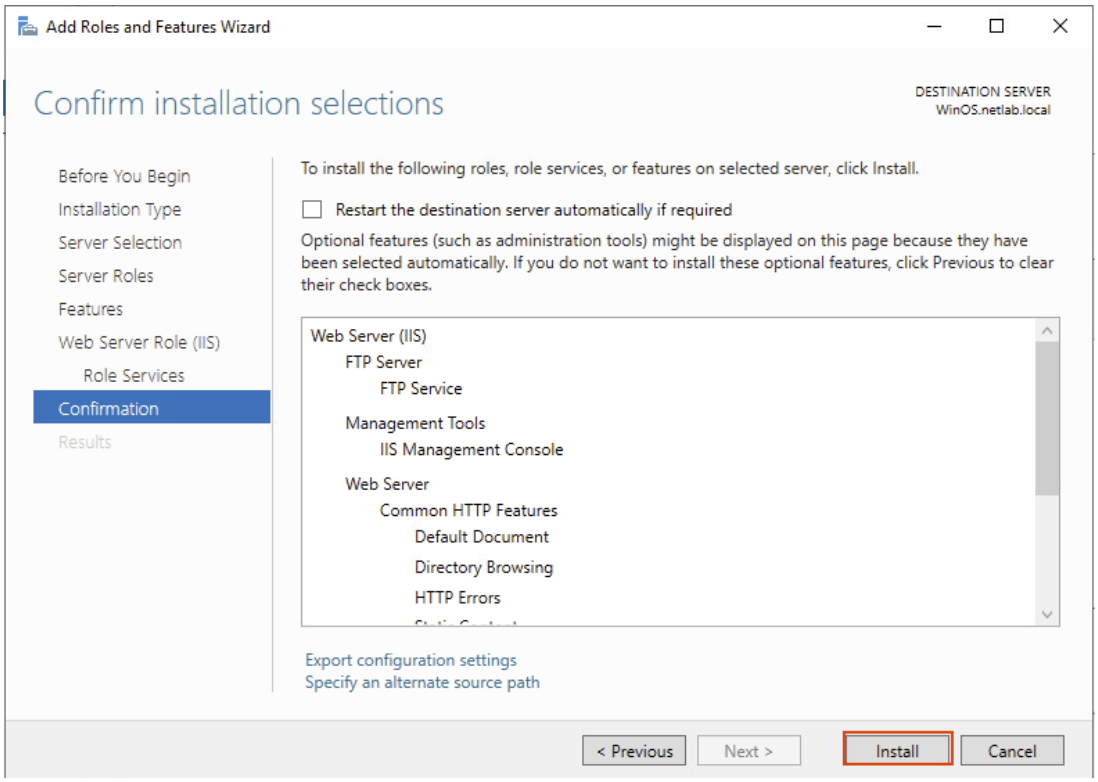
11. At the *Web Server Role (IIS)* window, click **Next.**



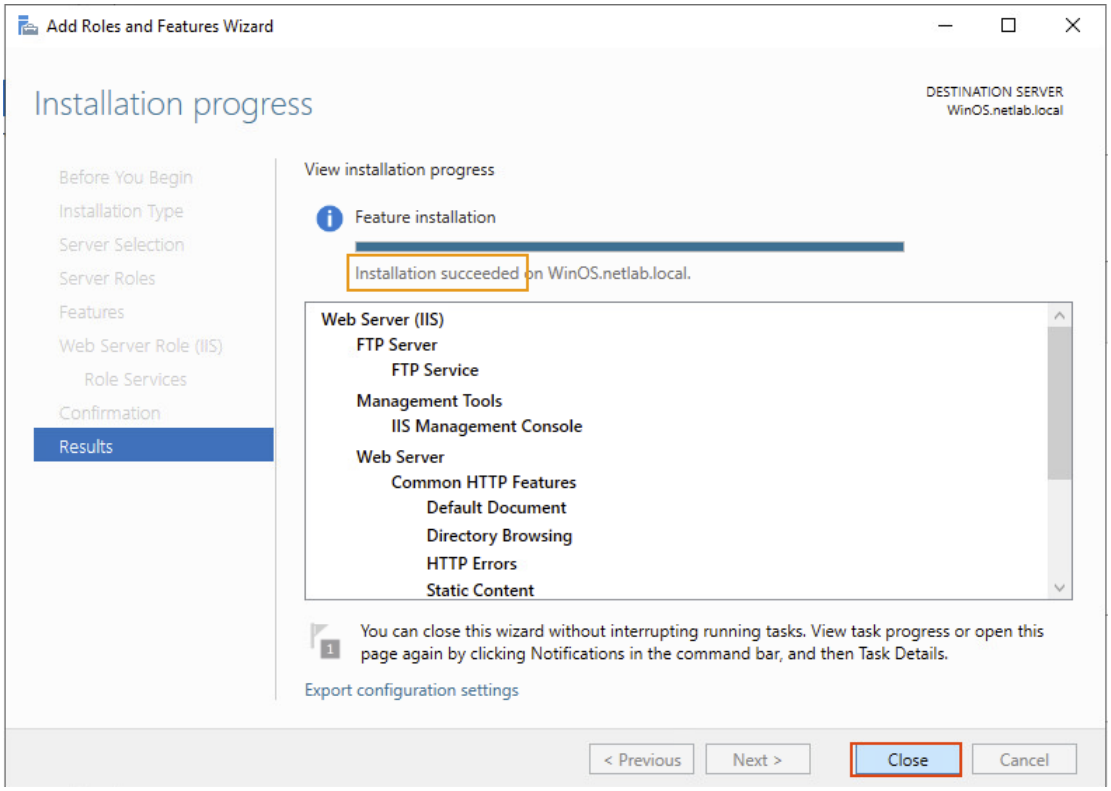
12. At the *Select role services* window, scroll down, and check the **FTP Server** box, click **Next.**



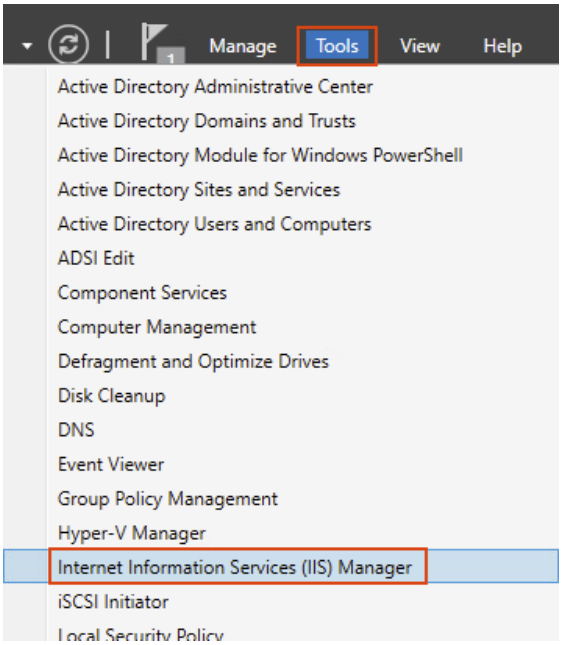
13. At the *Confirm installation selections* window, click **Install.**



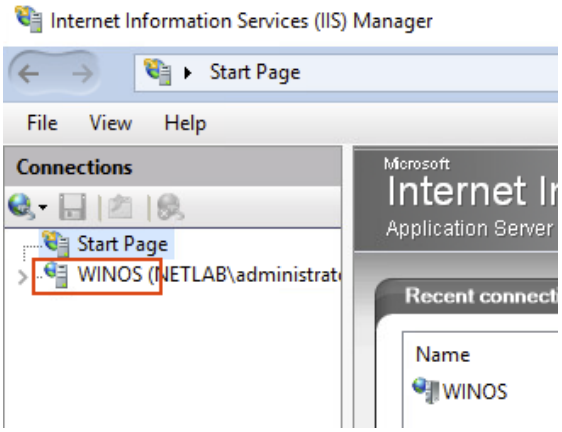
14. At the *Installation progress* window, wait until the process is finished. Click **Close.**



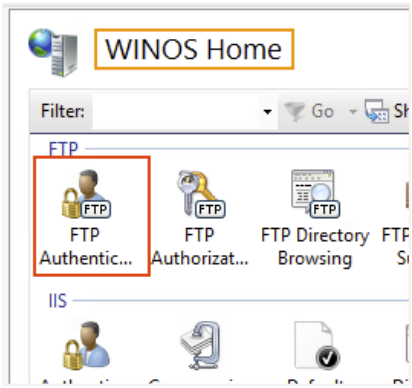
15. Once the window is closed, click the **Tools** menu in the *Server Manager* window, then select **Internet Information Services (IIS) Manager**.



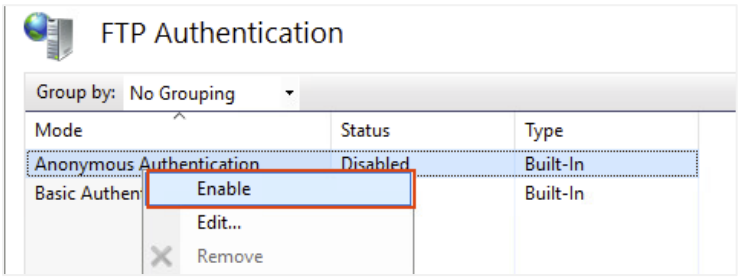
16. In the *IIS Manager* window, click the **WINOS** server.



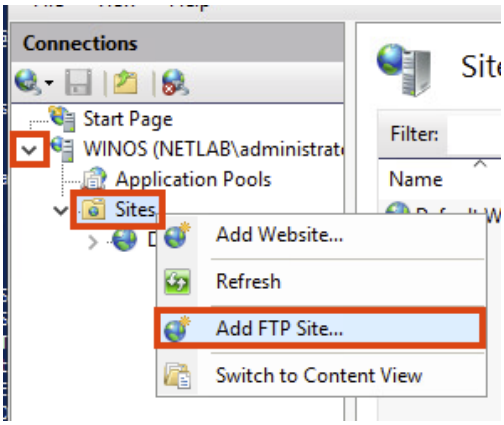
17. To the right side, you will see the *WINOS Home* page. Double-click the **FTP Authentication** icon.



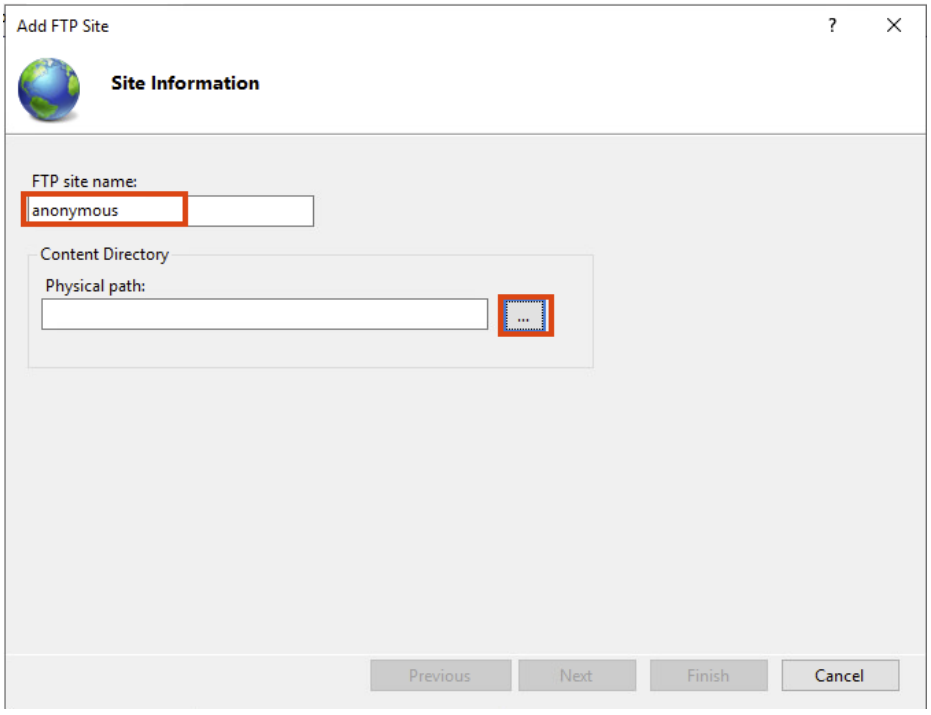
18. You will see the list of authentication modes and their status. For simplicity purposes, we will just enable the anonymous user. Right-click on **Anonymous Authentication**, select **Enable**.



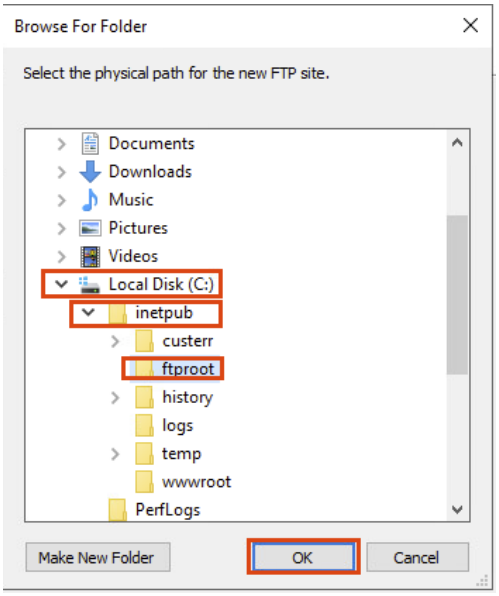
19. The next step is to create a site so *anonymous* can log in. In the left pane, click the **arrow** to expand *WINOS*, then right-click on the **Sites**. In the pop-up menu, select the **Add FTP Site** option.



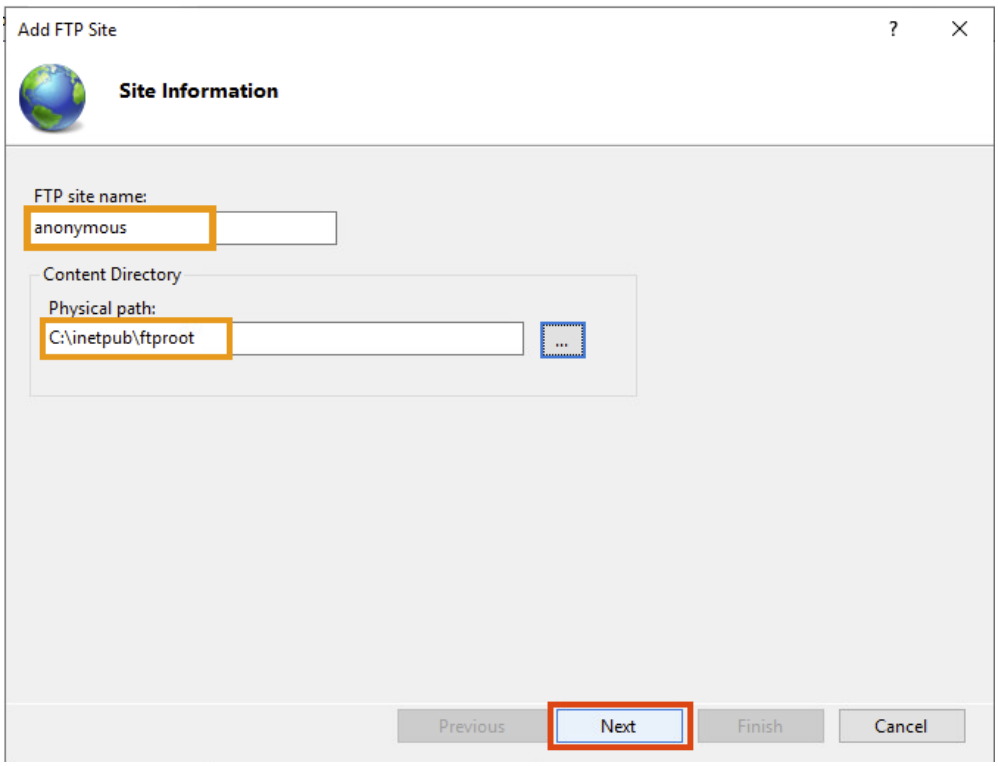
20. In the new window, type `anonymous` for the site name, and click the **button** with three horizon dots to set the path.



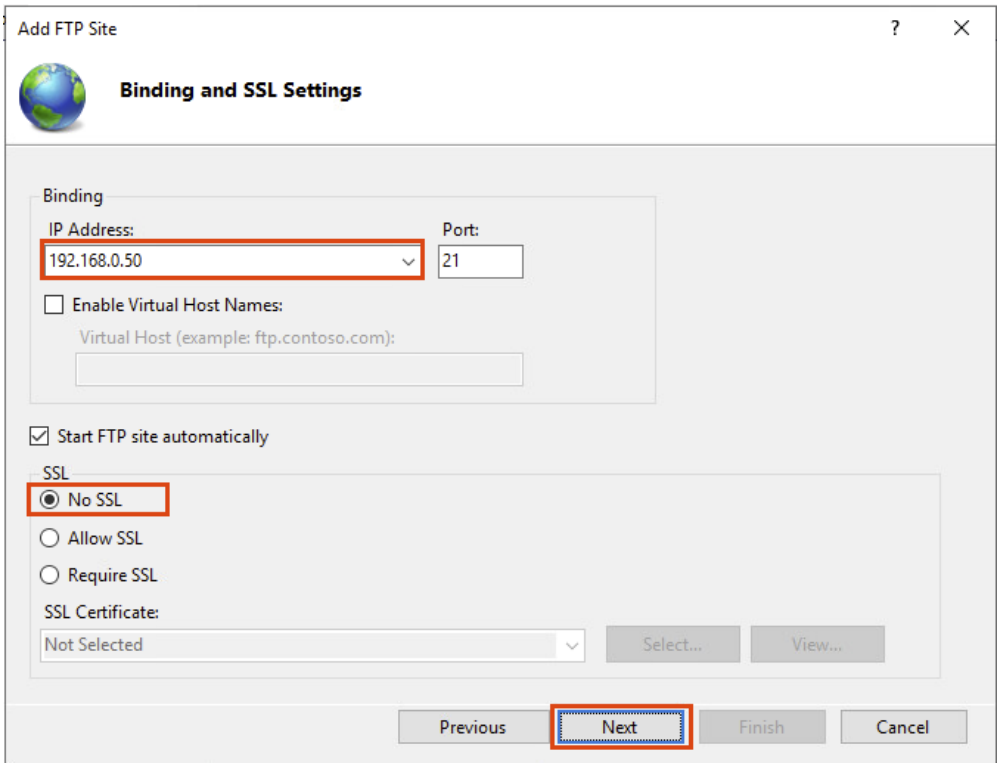
21. In the **Browse For Folder** window, expand **Local Disk (C:)**, then expand **inetpub**, then click on **ftproot**. Click **OK** to confirm.



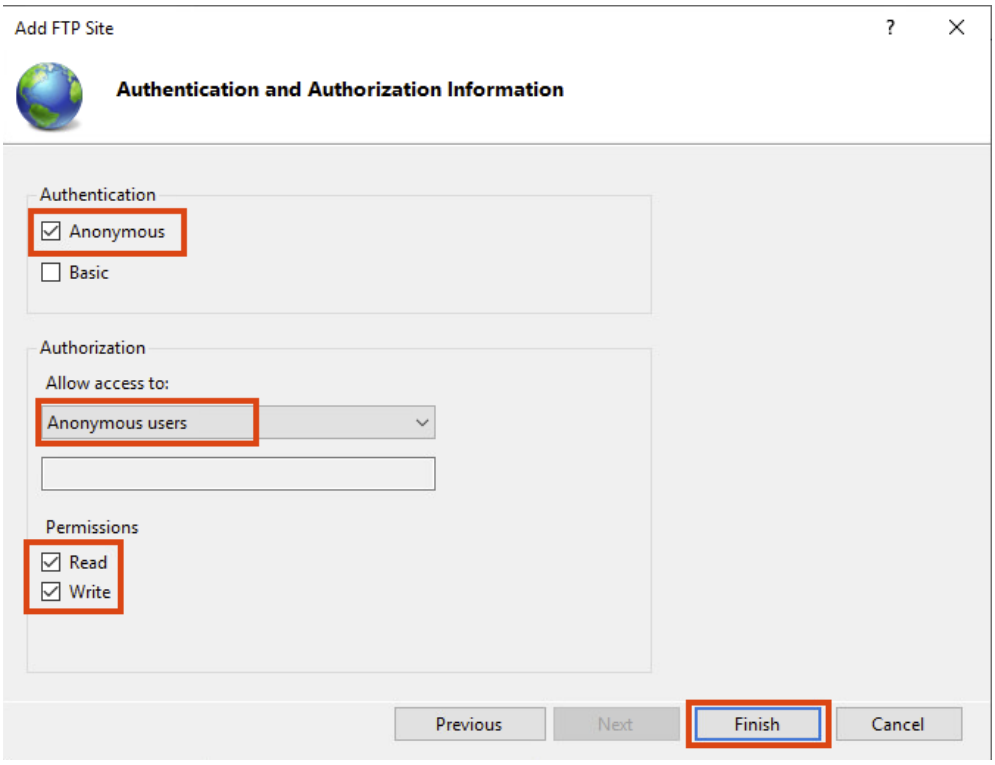
22. You will be brought back to the *Add FTP Site* window. Double-check everything, then click the **Next** button.



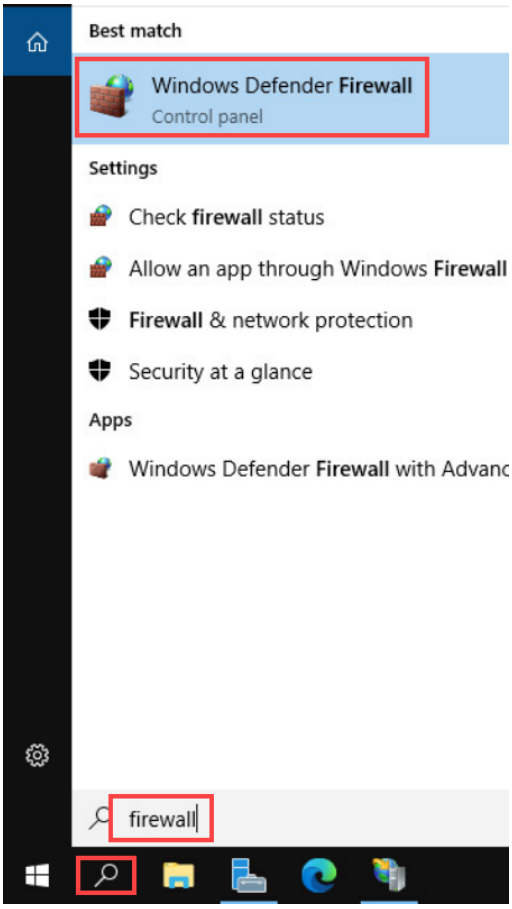
23. On the next page, click and select the binding address **192.168.0.50**, check to use **No SSL** and click **Next.**



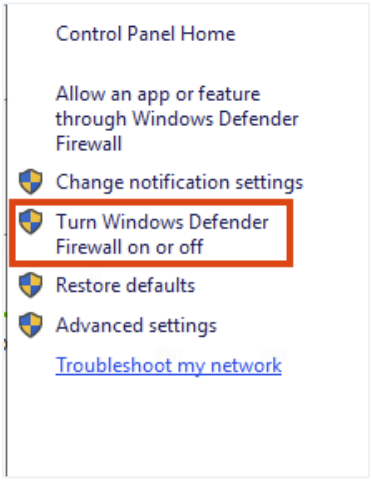
24. In the last step, make sure the **Anonymous** authentication is checked, and *Allow access to* **Anonymous users**, then add both **Read** and **Write** permissions. Click **Finish** to confirm.



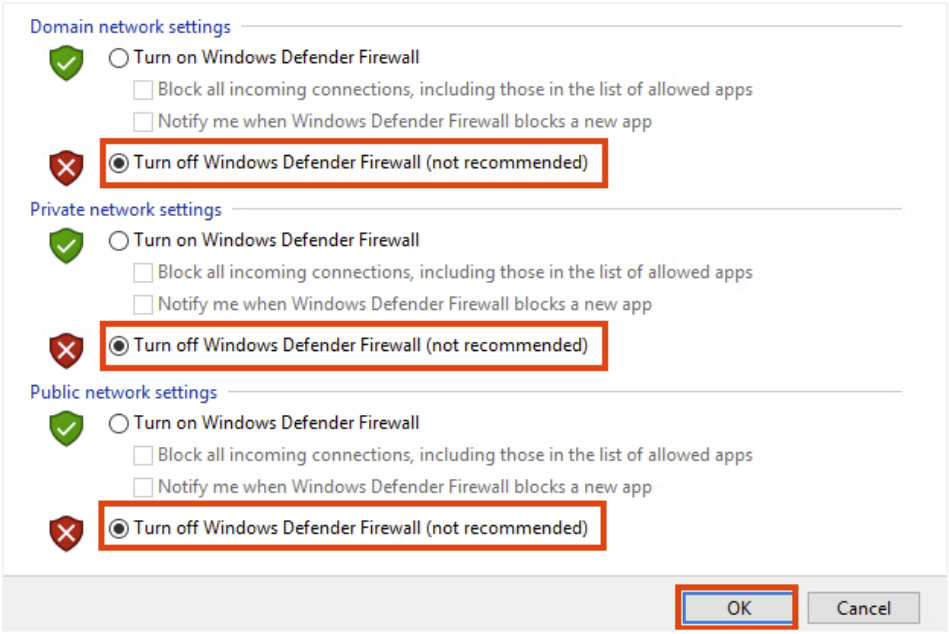
25. We will finish the last step for the FTP setup. Open the port to the world. Click the Search button, and type `firewall`. In the search result, click the first **Windows Defender Firewall.**



26. In the left pane of the new window, click **Turn Windows Defender Firewall on or off.**



27. Make sure to check **Turn Off Windows Defender Firewall** for all three types of networks. Then, click **OK** to confirm.



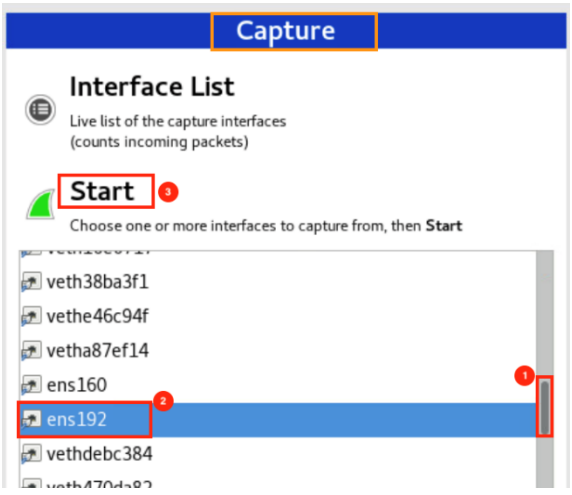
2.1.2 Monitor the FTP Traffic

1. In the *SecOnion* system, open a *Terminal* window and type the command below to run *Wireshark* as root. If prompted for a password, enter `NDGlabpass123!`.

```
sysadmin@seconion ~$ sudo wireshark
```

```
[sysadmin@seconion ~]$ sudo wireshark
[sudo] password for sysadmin:
```

2. In the *Wireshark* window, to the left side of the *Capture* section, scroll down and find **ens192**, click to select, then click the **Start** button.



3. Now, switch the focus to the *Kali* machine. Change focus to the terminal and type the command below to connect to the FTP server located on the *WinOS Server*. When prompted for a username and password, enter **anonymous** as the *user* and **anonymous** again as the *password*.

```
kali@kali ~$ ftp 192.168.0.50
```

```
┌──(kali㉿kali)-[~]
└─$ ftp 192.168.0.50
Connected to 192.168.0.50.
220 Microsoft FTP Service
Name (192.168.0.50:kali): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp>
```

4. Once the FTP connection is complete, type **exit** to exit out of the connection.

```
ftp> exit
```

```
ftp> exit
421 Service not available, remote server has closed connection

┌──(kali㉿kali)-[~]
└─$
```

5. Switch back to the *Wireshark* window on *SecOnion* and press the **Stop Capture** button.

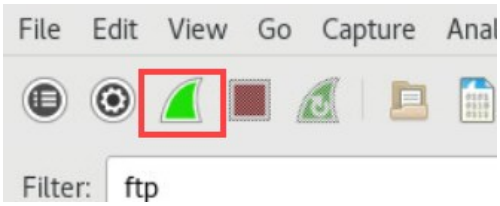
6. In the *Filter* field, type **ftp** and press **Enter**. Now that the packet focus is on *FTP* only, locate the initial request. Here, you can see the username *anonymous* and the password of *anonymous* in clear text

Filter: ftp Expression... Clear Apply Save

No.	Time	Source	Destination	Protoc	Lengt	Info
198	31.3417663	192.168.0.50	192.168.0.1	FTP	81	Response: 220 Microsoft FTP Service
222	36.2912945	192.168.0.1	192.168.0.50	FTP	70	Request: USER anonymous
223	36.2916119	192.168.0.50	192.168.0.1	FTP	126	Response: 331 Anonymous access allowed, send identity (e-mail name) as password.
256	39.1768601	192.168.0.1	192.168.0.50	FTP	70	Request: PASS anonmyous
257	39.1792199	192.168.0.50	192.168.0.1	FTP	75	Response: 230 User logged in.
259	39.1794560	192.168.0.1	192.168.0.50	FTP	60	Request: SYST
260	39.1795707	192.168.0.50	192.168.0.1	FTP	70	Response: 215 Windows_NT
276	43.7209644	192.168.0.1	192.168.0.50	FTP	60	Request: QUIT
277	43.7212495	192.168.0.50	192.168.0.1	FTP	68	Response: 221 Goodbye.

2.2 Using Wireshark to Capture SFTP Traffic

1. Start a new capture by clicking on the **Start a new live capture** button.

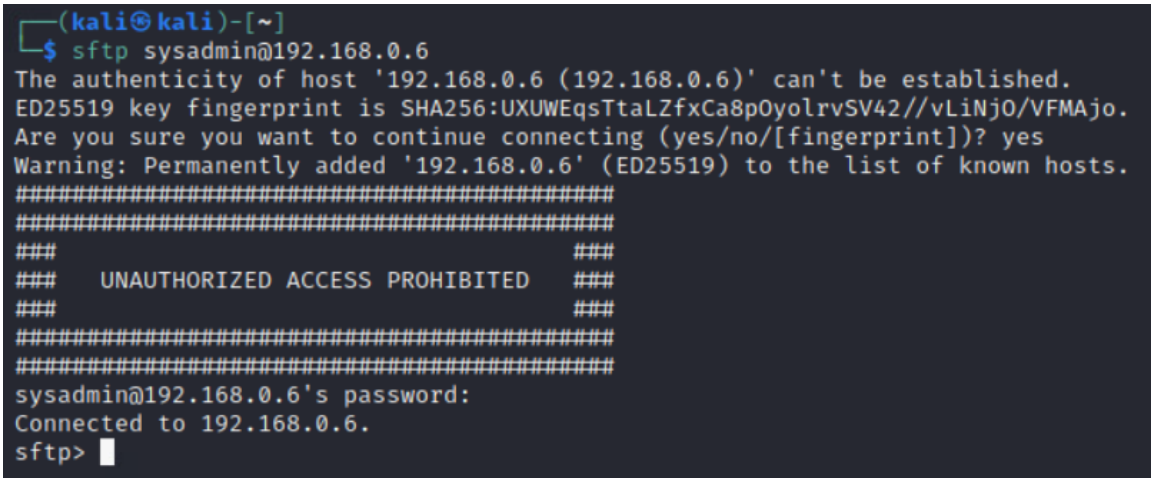


2. If prompted to save the capture file, select **Continue without Saving.**

3. In the filter pane, type `ssh` and press **Enter**.

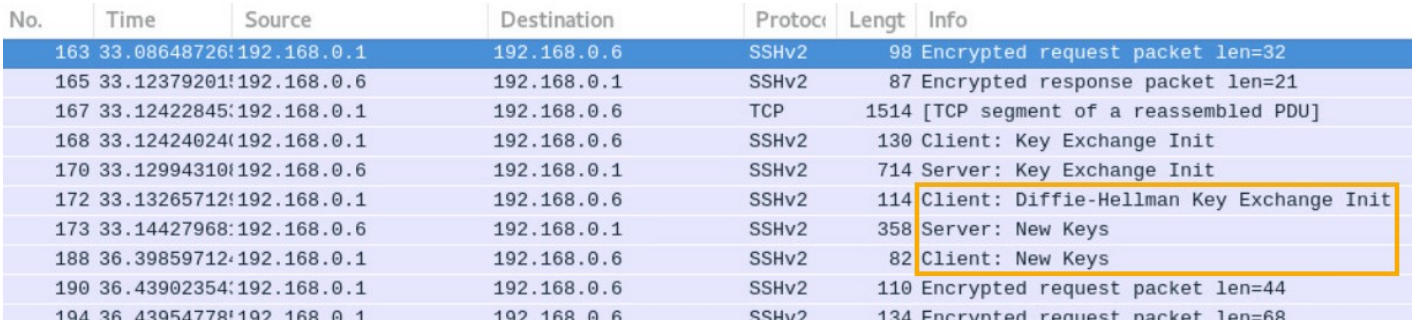
4. Change focus to the **Kali** system. Focus on the terminal and enter the command below, when asked, *are you sure you want to continue*, type `yes`. Then, you will be prompted for a password; enter `NDGlabpass123!`.

```
kali@kali:~$ sftp sysadmin@192.168.0.6
```



5. Switch to the **SecOnion** system and stop the capture by clicking the **Stop Capture** button.

6. Locate the *Diffie-Hellman key exchange* between the client and the *SFTP* service.



No.	Time	Source	Destination	Protoc	Lengt	Info
163	33.08648726	192.168.0.1	192.168.0.6	SSHv2	98	Encrypted request packet len=32
165	33.12379201	192.168.0.6	192.168.0.1	SSHv2	87	Encrypted response packet len=21
167	33.12422845	192.168.0.1	192.168.0.6	TCP	1514	[TCP segment of a reassembled PDU]
168	33.12424024	192.168.0.1	192.168.0.6	SSHv2	130	Client: Key Exchange Init
170	33.12994310	192.168.0.6	192.168.0.1	SSHv2	714	Server: Key Exchange Init
172	33.13265712	192.168.0.1	192.168.0.6	SSHv2	114	Client: Diffie-Hellman Key Exchange Init
173	33.14427968	192.168.0.6	192.168.0.1	SSHv2	358	Server: New Keys
188	36.39859712	192.168.0.1	192.168.0.6	SSHv2	82	Client: New Keys
190	36.43902354	192.168.0.1	192.168.0.6	SSHv2	110	Encrypted request packet len=44
194	36.43954778	192.168.0.1	192.168.0.6	SSHv2	134	Encrypted request packet len=68

After the key exchange, the packets that follow are encrypted over the medium. Feel free to clear the filter and examine the traffic. You will no longer see the username and password in clear text compared to FTP.

7. Close the **Wireshark** application. When prompted, click **Quit without Saving**.
8. Leave the *SecOnion* viewer open to continue with the next task.

3 Capturing and Analyzing HTTP Traffic

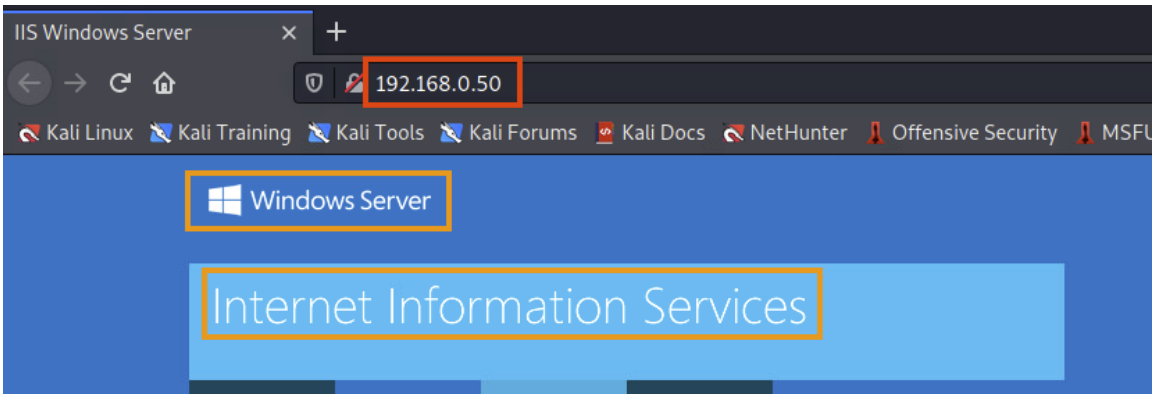
3.1 Using dumpcap to Capture HTTP Traffic

1. While on the *SecOnion* system, focus on the terminal window and enter the command below. If prompted for a password, enter `NDGlabpass123!`.

```
sysadmin@seconion ~$ sudo dumpcap -P -i ens192 -w /tmp/netcapture2.pcap
```

```
[sysadmin@seconion ~]$ sudo dumpcap -P -i ens192 -w /tmp/netcapture2.pcap
Capturing on 'ens192'
File: /tmp/netcapture2.pcap
```

2. Switch focus to the *Kali* system, start a browser, then go to address `192.168.0.50` followed by pressing the **Enter** key. Wait until the page finishes loading. The address should return the *Internet Information Services* default page on *Windows Server*.



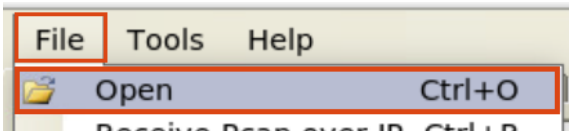
3. Switch back to the *Terminal* running *dumpcap* and press **CTRL+C** to stop the running process

```
[sysadmin@seconion ~]$ sudo dumpcap -P -i ens192 -w /tmp/netcapture2.pcap
Capturing on 'ens192'
File: /tmp/netcapture2.pcap
Packets captured: 306
Packets received/dropped on interface 'ens192': 306/0 (pcap:0/dumpcap:0/flushed:
0) (100.0%)
```

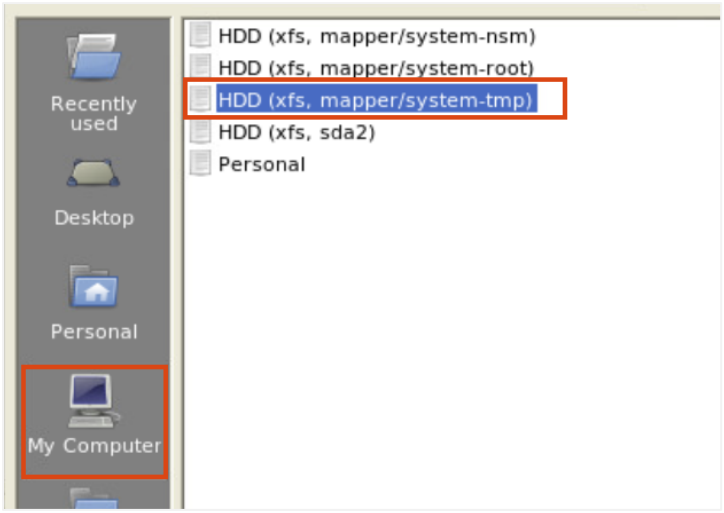
4. Leave the *Terminal* open to continue with the next task.

3.2 Using Network Miner to Capture HTTP Traffic

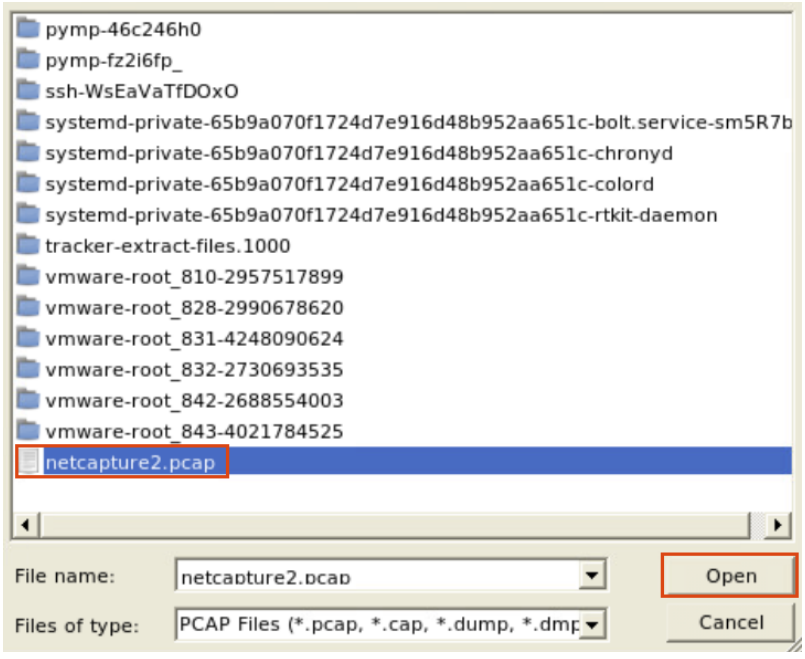
1. While in the *Terminal*, enter the command **`sudo networkminer`** to open the program (enter **`NDGlabpass123!`** if prompted for a password). On the *Network Miner* application window, navigate to **File > Open.**



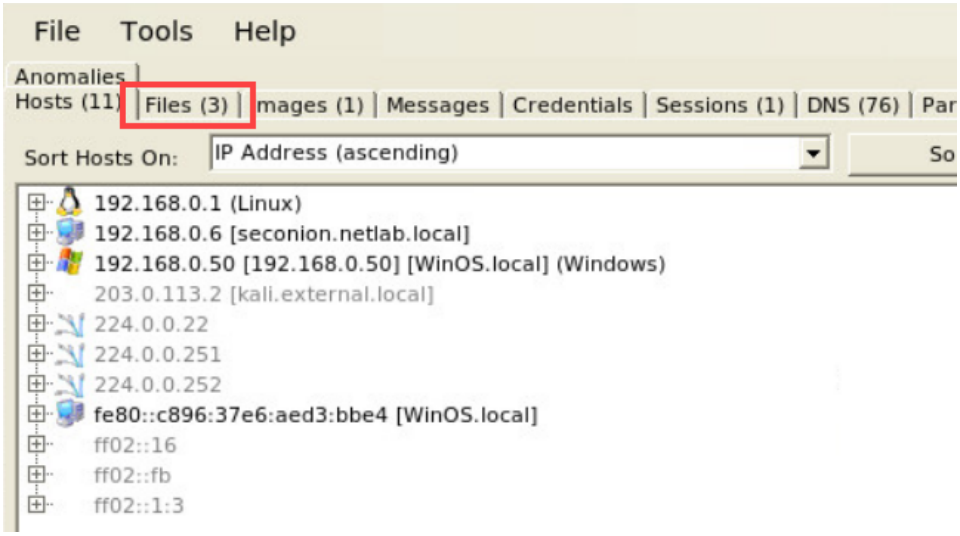
2. Select the **My Computer** icon from the left pane. On the right side, double-click **HDD (xfs, mapper/system/tmp).**



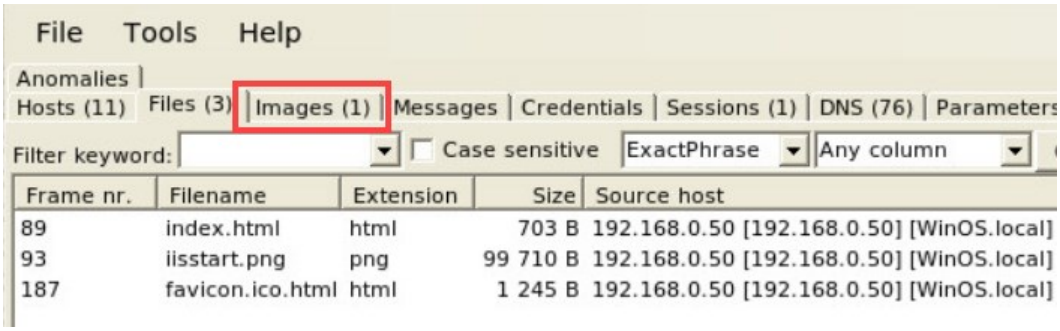
3. Next, Select the **netcapture2.pcap** file and select **Open.** If you don't see the *netcapture2.pcap* file, scroll the horizontal bar to the right side, until you see *netcapture2.pcap* on the top.



4. Once the *PCAP* is finished loading, click on the **Files** tab within the *Network Miner* program.



5. Notice the list of files acquired. Click on the **Images** tab.



6. Notice the images that are captured.
7. The lab is now complete; you may end the reservation.