Everlyn Leon

Abstract

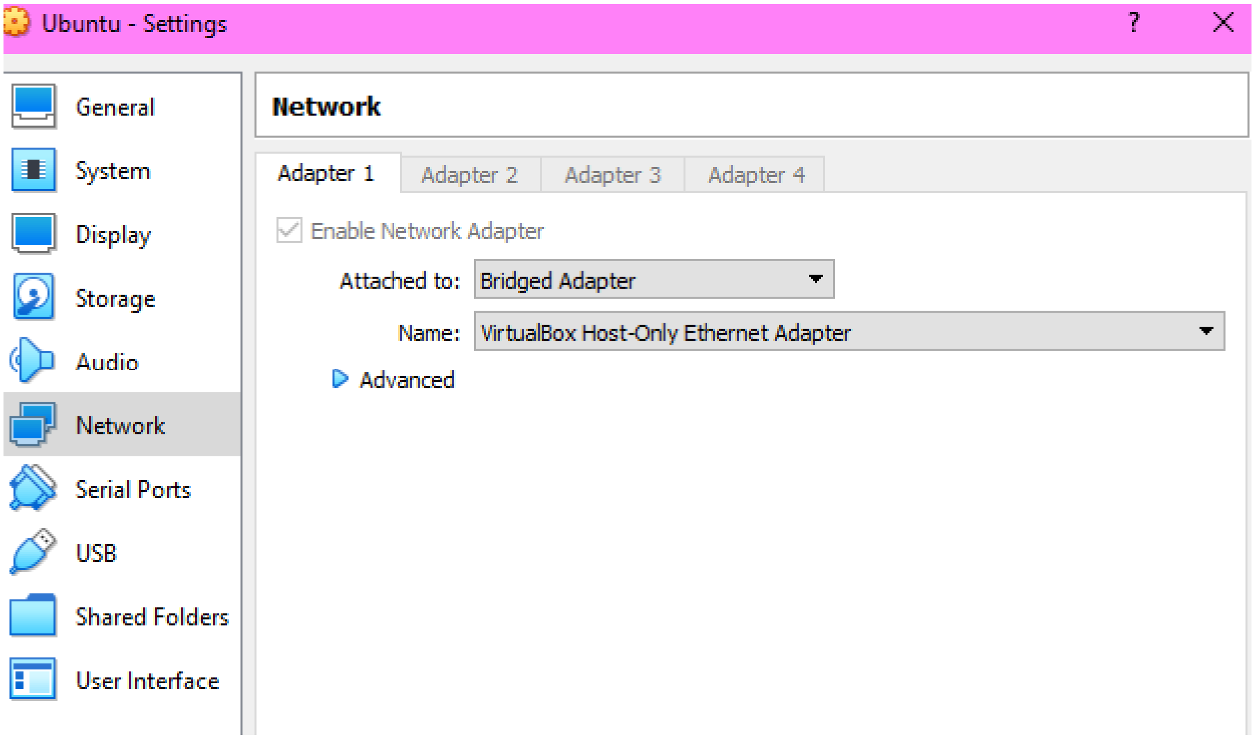
Demonstrate network mapping using Nmap, Zenmap and Wireshark using a Ubuntu Virtual Machine and Kali Linux Virtual Machine.

Introduction

Demonstrating the capabilities of Nmap, Zenmap, and Wireshark, as well as setting up the Ubuntu Virtual Machine to be scanned by disabling its firewall and the Kali Linux Virtual Machine to scan and capture packets using Wireshark, Nmap and Zenmap. The Kali Linux machine will attempt to connect to the Ubuntu machine using SSH, view a webpage and connect a nc listener.

Summary of Results

The workspace has to be set up by making sure that both the Ubuntu Virtual Machine and the Kali Linux Virtual Machine are existing on the same network by choosing the 'Bridged Adapter' option.



Network

☑ Enable Network Adapter

Attached to: Bridged Adapter

Name: VirtualBox Host-Only Ethernet Adapter

Setting up Ubuntu to be Scanned

In order to demonstrate network mapping using Nmap, Zenmap and Wireshark, we first need to disable the firewall on the Ubuntu machine using the command:

sudo ufw disable

To SSH from the Kali Linux machine to the Ubuntu machine, SSH needs to be installed using the command:

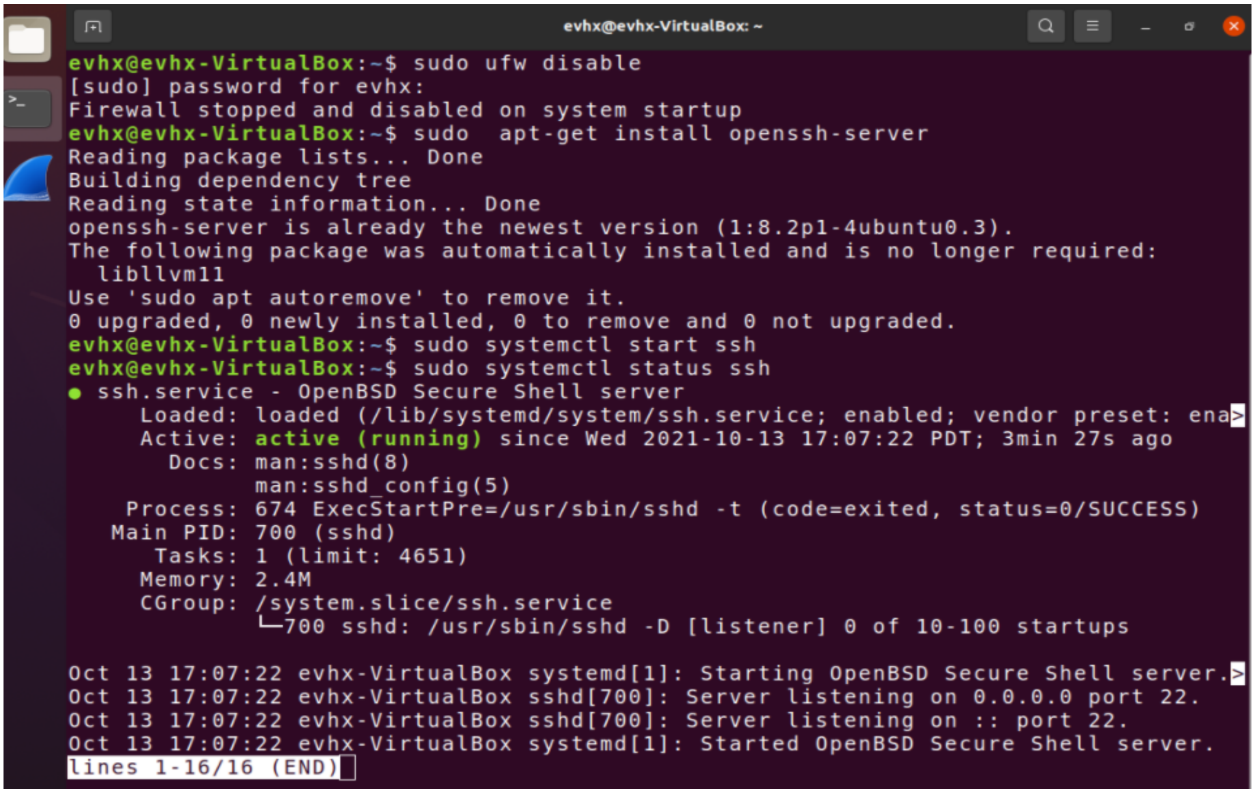
sudo apt-get install openssh-server

To start SSH and check the status of SSH, use the commands:

sudo systemctl start ssh
sudo systemctl status ssh

Commands used:

sudo	(provides root permission	)
apt-get install openssh-server	(install the openssh-server	)
ufw disable	(disables firewall	)
systemctl start ssh	(starts SSH	)
systemctl status ssh	(checks status of SSH	)



```
evhx@evhx-VirtualBox:~$ sudo ufw disable
[sudo] password for evhx:
Firewall stopped and disabled on system startup
evhx@evhx-VirtualBox:~$ sudo  apt-get install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-server is already the newest version (1:8.2p1-4ubuntu0.3).
The following package was automatically installed and is no longer required:
  libllvm11
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
evhx@evhx-VirtualBox:~$ sudo systemctl start ssh
evhx@evhx-VirtualBox:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: ena>
     Active: active (running) since Wed 2021-10-13 17:07:22 PDT; 3min 27s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 674 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 700 (sshd)
      Tasks: 1 (limit: 4651)
     Memory: 2.4M
     CGroup: /system.slice/ssh.service
             └─700 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Oct 13 17:07:22 evhx-VirtualBox systemd[1]: Starting OpenBSD Secure Shell server.>
Oct 13 17:07:22 evhx-VirtualBox sshd[700]: Server listening on 0.0.0.0 port 22.
Oct 13 17:07:22 evhx-VirtualBox sshd[700]: Server listening on :: port 22.
Oct 13 17:07:22 evhx-VirtualBox systemd[1]: Started OpenBSD Secure Shell server.
lines 1-16/16 (END)
```

After SSH is successfully installed and active on the Ubuntu machine, the ip address of the machine will be needed for future operations. To gather information on the Ubuntu machines addresses, use the command:

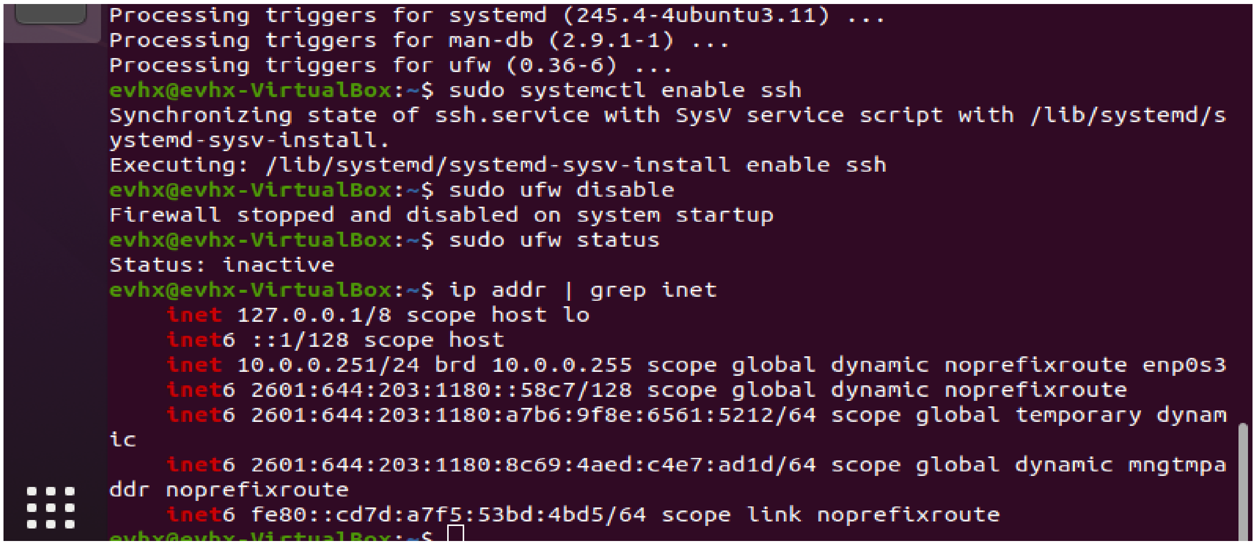
ip addr | grep inet

Then create a netcat listener on the Ubuntu machine using the command:

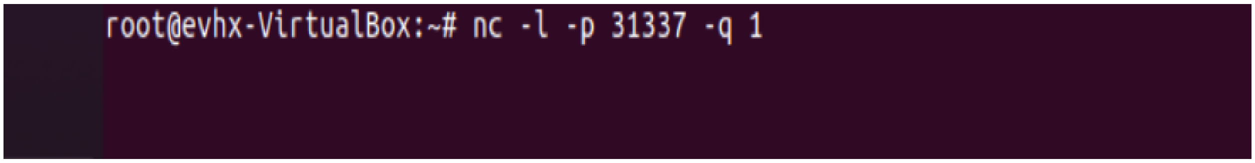
nc -l -p 31337 -q 1

Commands used:

ip addr	(display ip addresses	)
grep inet	(specify the inet	)
nc	(netcat	)
-l	(listener	)
-p	(port number	)



```
Processing triggers for systemd (245.4-4ubuntu3.11) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for ufw (0.36-6) ...
evhx@evhx-VirtualBox:~$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /lib/systemd/s
ystemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
evhx@evhx-VirtualBox:~$ sudo ufw disable
Firewall stopped and disabled on system startup
evhx@evhx-VirtualBox:~$ sudo ufw status
Status: inactive
evhx@evhx-VirtualBox:~$ ip addr | grep inet
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
    inet 10.0.0.251/24 brd 10.0.0.255 scope global dynamic noprefixroute enp0s3
    inet6 2601:644:203:1180::58c7/128 scope global dynamic noprefixroute
    inet6 2601:644:203:1180:a7b6:9f8e:6561:5212/64 scope global temporary dynam
ic
    inet6 2601:644:203:1180:8c69:4aed:c4e7:ad1d/64 scope global dynamic mngtmpa
ddr noprefixroute
    inet6 fe80::cd7d:a7f5:53bd:4bd5/64 scope link noprefixroute
evhx@evhx-VirtualBox:~$ 
```



```
root@evhx-VirtualBox:~# nc -l -p 31337 -q 1
```

Setting up the Kali Machine to Scan

To set up the Kali Linux machine, we must first update the machine, which is a core rule. Always update your machine. Update the Kali Linux machine with the command:

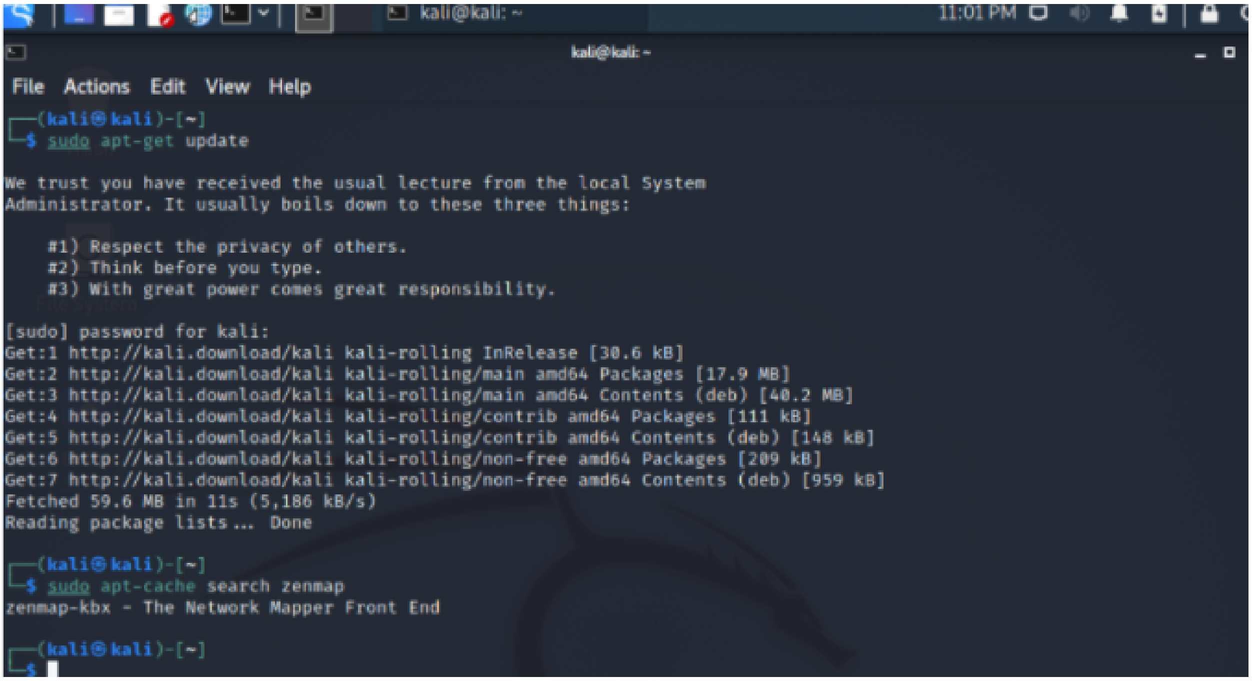
sudo apt-get update

Then to specifically install Zenmap, we need to search for it and then we can install it after given the installation name with the commands:

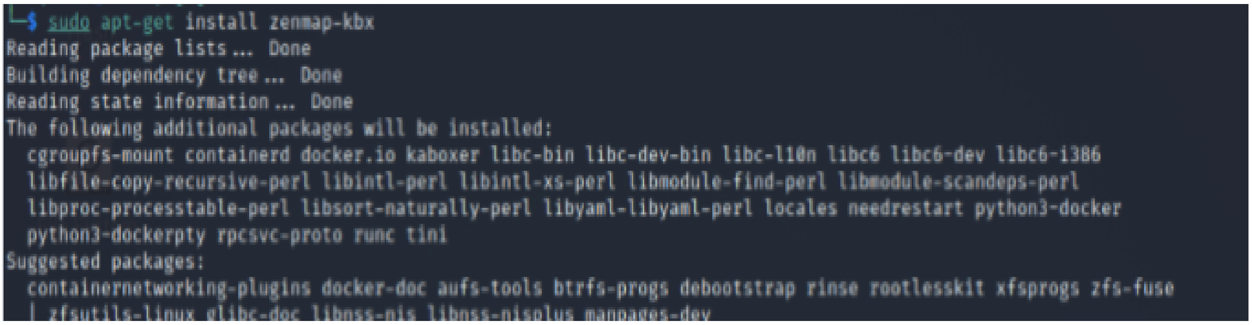
sudo apt-cache search zenmap
sudo apt-get install zenmap-kbx

Commands used:

sudo apt-get update	(get update	)
sudo apt-cache search zenmap	(search for Zenmap package name	)
sudo apt-get install zenmap-kbx	(install Zenmap	)



```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get update

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [30.6 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [17.9 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [40.2 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [111 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [148 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [209 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [959 kB]
Fetched 59.6 MB in 11s (5,186 kB/s)
Reading package lists... Done

┌──(kali㉿kali)-[~]
└─$ sudo apt-cache search zenmap
zenmap-kbx - The Network Mapper Front End

┌──(kali㉿kali)-[~]
└─$ 
```

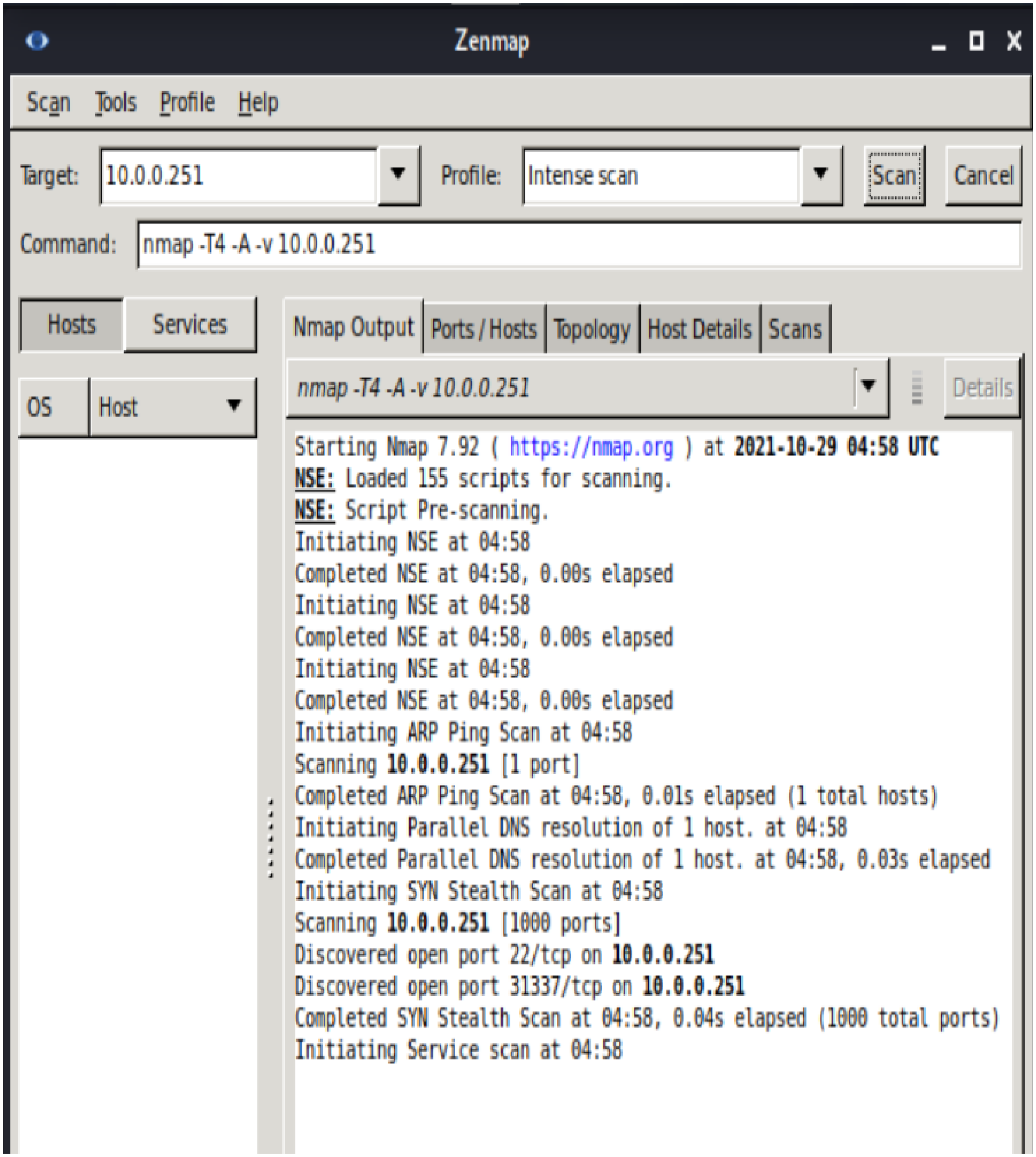


```
└─$ sudo apt-get install zenmap-kbx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  cgroupfs-mount containerd docker.io kaboxer libc-bin libc-dev-bin libc-l10n libc6 libc6-dev libc6-i386
  libfile-copy-recursive-perl libintl-perl libintl-xs-perl libmodule-find-perl libmodule-scandeps-perl
  libproc-processtable-perl libsort-naturally-perl libyaml-libyaml-perl locales needrestart python3-docker
  python3-dockerpty rpcsvc-proto runc tini
Suggested packages:
  containernetworking-plugins docker-doc aufs-tools btrfs-progs debootstrap rinse rootlesskit xfsprogs zfs-fuse
  | zfsutils-linux glibc-doc libnss-nis libnss-nisplus manpages-dev
```

After Zenmap has been successfully installed, you can view its GUI by using the command:

zenmap-kbx

Commands used:
zenmap-kbx (Start Zenmap)



Target: 10.0.0.251

Profile: Intense scan

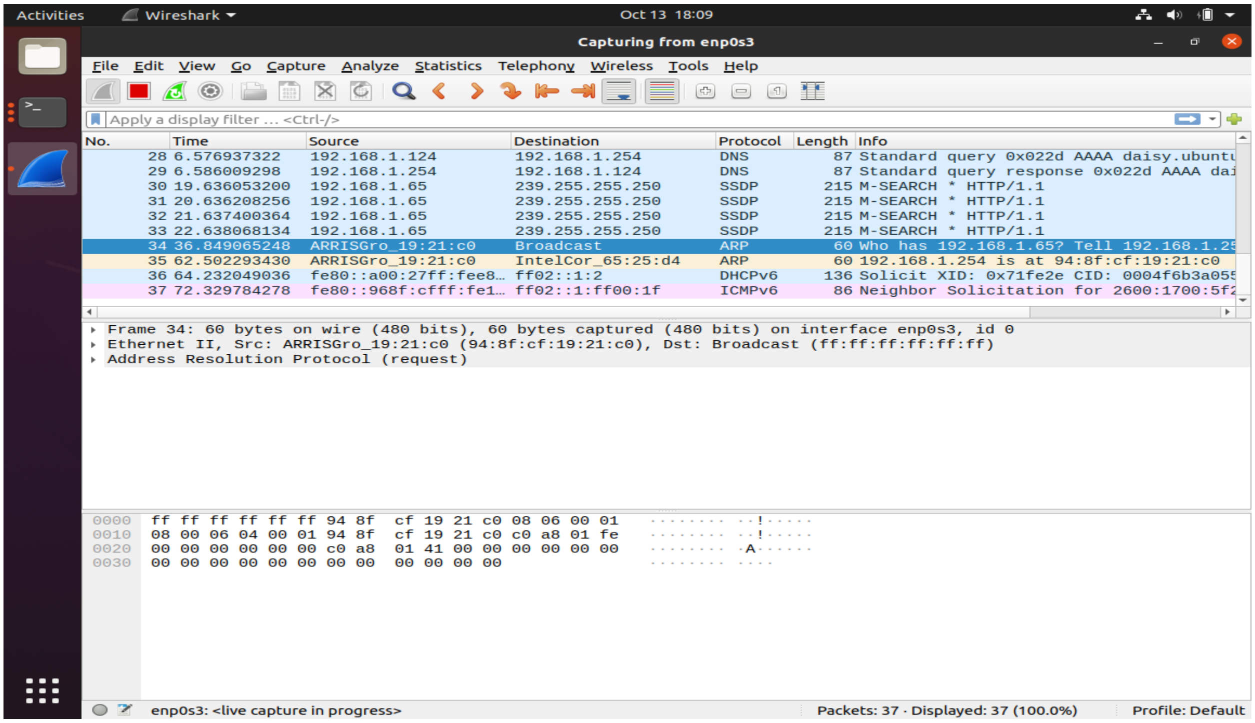
Command: nmap -T4 -A -v 10.0.0.251

```
nmap -T4 -A -v 10.0.0.251
```

```
Starting Nmap 7.92 ( https://nmap.org ) at 2021-10-29 04:58 UTC
NSE: Loaded 155 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 04:58
Completed NSE at 04:58, 0.00s elapsed
Initiating NSE at 04:58
Completed NSE at 04:58, 0.00s elapsed
Initiating NSE at 04:58
Completed NSE at 04:58, 0.00s elapsed
Initiating ARP Ping Scan at 04:58
Scanning 10.0.0.251 [1 port]
Completed ARP Ping Scan at 04:58, 0.01s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 04:58
Completed Parallel DNS resolution of 1 host. at 04:58, 0.03s elapsed
Initiating SYN Stealth Scan at 04:58
Scanning 10.0.0.251 [1000 ports]
Discovered open port 22/tcp on 10.0.0.251
Discovered open port 31337/tcp on 10.0.0.251
Completed SYN Stealth Scan at 04:58, 0.04s elapsed (1000 total ports)
Initiating Service scan at 04:58
```

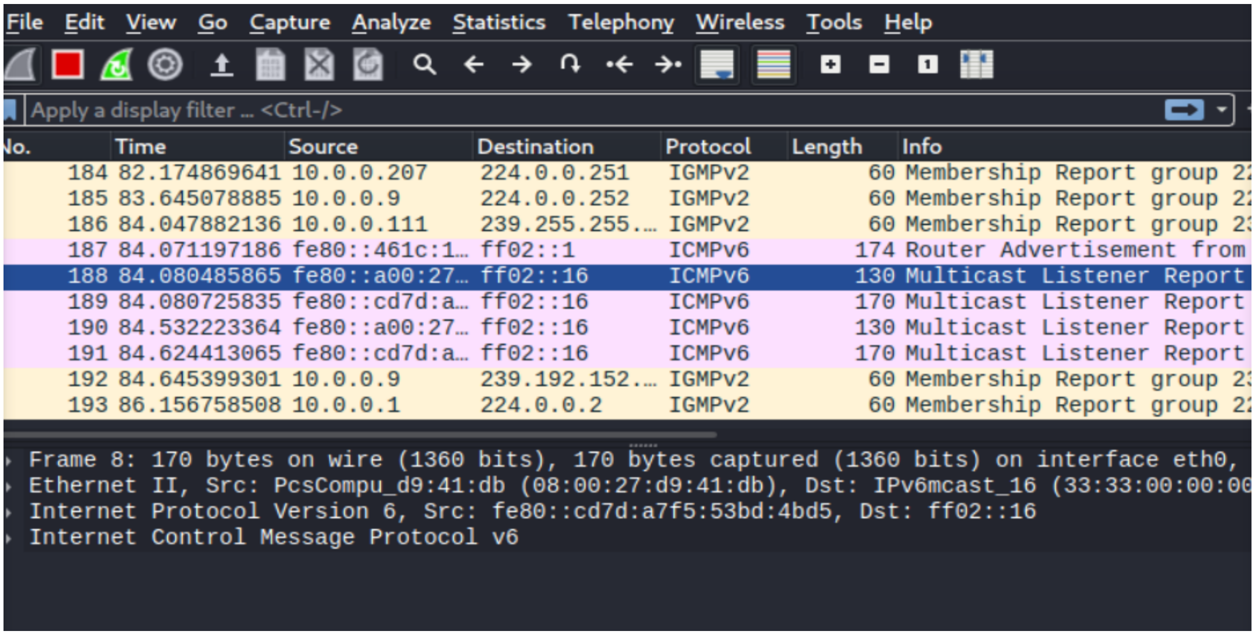
Wireshark will be used because it provides a variety of tools to make network packets easy to analyze, such as a display filter, the list of captured packets, the details to any selected packet, and the ASCII and hexadecimal contained within the packets.

On both the Ubuntu machine and the Kali Linux machine, start packet capturing with Wireshark.



Activities
Wireshark
Oct 13 18:09
Capturing from enp0s3
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

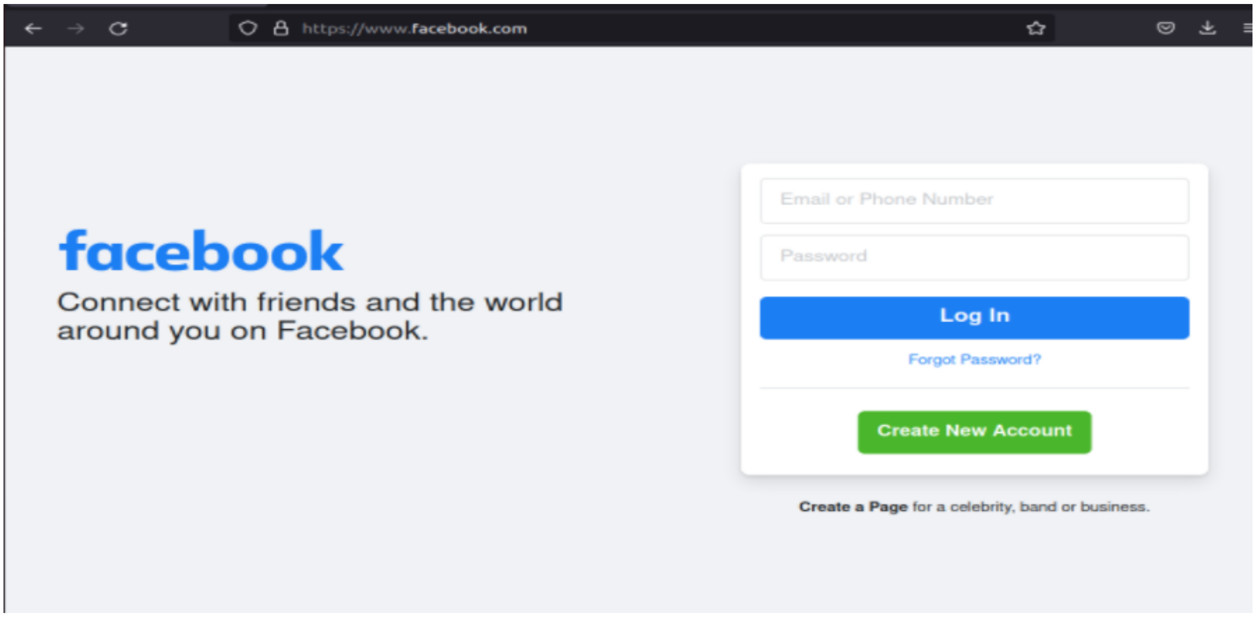
```
No.    Time          Source                  Destination        Protocol  Length Info
    28 6.576937322   192.168.1.124           192.168.1.254      DNS           87 Standard query 0x022d AAAA daisy.ubuntu
    29 6.586009298   192.168.1.254           192.168.1.124      DNS           87 Standard query response 0x022d AAAA dai
    30 19.636053200  192.168.1.65            239.255.255.250    SSDP         215 M-SEARCH * HTTP/1.1
    31 20.636208256  192.168.1.65            239.255.255.250    SSDP         215 M-SEARCH * HTTP/1.1
    32 21.637400364  192.168.1.65            239.255.255.250    SSDP         215 M-SEARCH * HTTP/1.1
    33 22.638068134  192.168.1.65            239.255.255.250    SSDP         215 M-SEARCH * HTTP/1.1
    34 36.849065248  ARRISGro_19:21:c0       Broadcast          ARP           60 Who has 192.168.1.65? Tell 192.168.1.25
    35 62.502293430  ARRISGro_19:21:c0       IntelCor_65:25:d4  ARP           60 192.168.1.254 is at 94:8f:cf:19:21:c0
    36 64.232049036  fe80::a00:27ff:fee8…    ff02::1:2          DHCPv6       136 Solicit XID: 0x71fe2e CID: 0004f6b3a055
    37 72.329784278  fe80::968f:cfff:fe1…    ff02::1:ff00:1f    ICMPv6        86 Neighbor Solicitation for 2600:1700:5f2

▸ Frame 34: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface enp0s3, id 0
▸ Ethernet II, Src: ARRISGro_19:21:c0 (94:8f:cf:19:21:c0), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▸ Address Resolution Protocol (request)

0000  ff ff ff ff ff ff 94 8f  cf 19 21 c0 08 06 00 01   ········ ··!·····
0010  08 00 06 04 00 01 94 8f  cf 19 21 c0 c0 a8 01 fe   ········ ··!·····
0020  00 00 00 00 00 00 c0 a8  01 41 00 00 00 00 00 00   ········ ·A······
0030  00 00 00 00 00 00 00 00  00 00 00 00               ········ ····

enp0s3: <live capture in progress>          Packets: 37 · Displayed: 37 (100.0%)          Profile: Default
```



File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.    Time          Source                Destination      Protocol  Length Info
   184 82.174869641 10.0.0.207            224.0.0.251      IGMPv2        60 Membership Report group 22
   185 83.645078885 10.0.0.9              224.0.0.252      IGMPv2        60 Membership Report group 22
   186 84.047882136 10.0.0.111            239.255.255.…    IGMPv2        60 Membership Report group 23
   187 84.071197186 fe80::461c:1…         ff02::1          ICMPv6       174 Router Advertisement from
   188 84.080485865 fe80::a00:27…         ff02::16         ICMPv6       130 Multicast Listener Report
   189 84.080725835 fe80::cd7d:a…         ff02::16         ICMPv6       170 Multicast Listener Report
   190 84.532223364 fe80::a00:27…         ff02::16         ICMPv6       130 Multicast Listener Report
   191 84.624413065 fe80::cd7d:a…         ff02::16         ICMPv6       170 Multicast Listener Report
   192 84.645399301 10.0.0.9              239.192.152.…    IGMPv2        60 Membership Report group 23
   193 86.156758508 10.0.0.1              224.0.0.2        IGMPv2        60 Membership Report group 22

▸ Frame 8: 170 bytes on wire (1360 bits), 170 bytes captured (1360 bits) on interface eth0,
▸ Ethernet II, Src: PcsCompu_d9:41:db (08:00:27:d9:41:db), Dst: IPv6mcast_16 (33:33:00:00:00
▸ Internet Protocol Version 6, Src: fe80::cd7d:a7f5:53bd:4bd5, Dst: ff02::16
▸ Internet Control Message Protocol v6
```

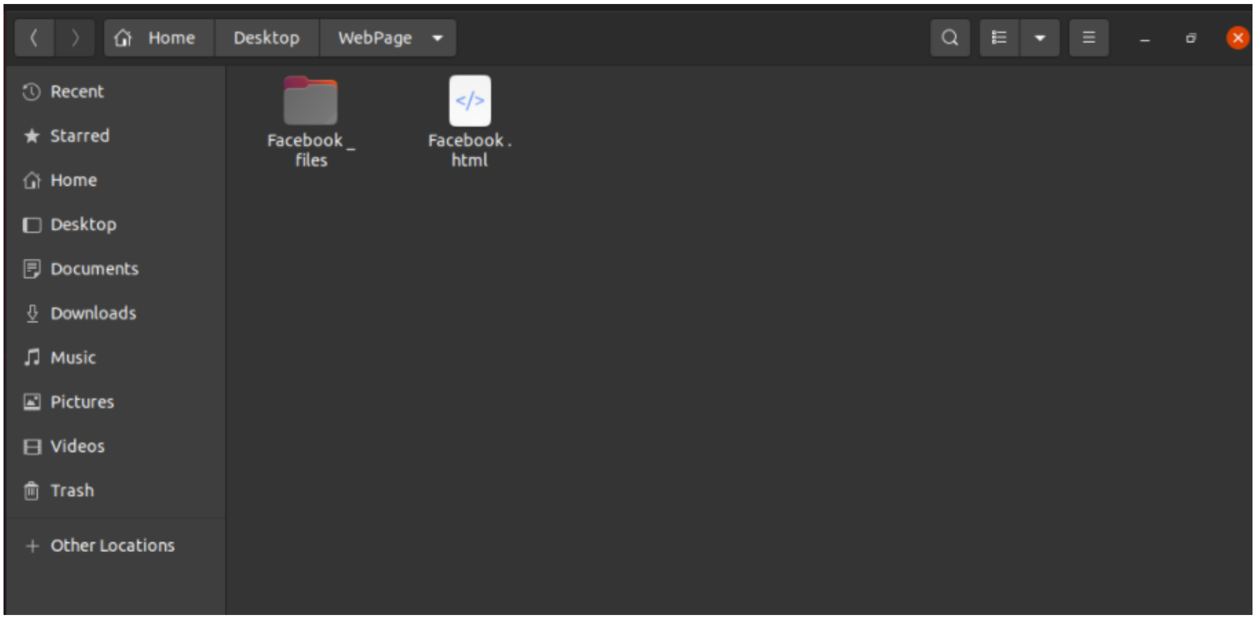
Before the projects involving SSH and the nc listener, let's start a web server using python with any html file. First we need to find(or make) a website to save in a specified directory. Here we have downloaded a Facebook page, and saved the documents into a directory named ‘WebPage’.



https://www.facebook.com

Connect with friends and the world around you on Facebook.

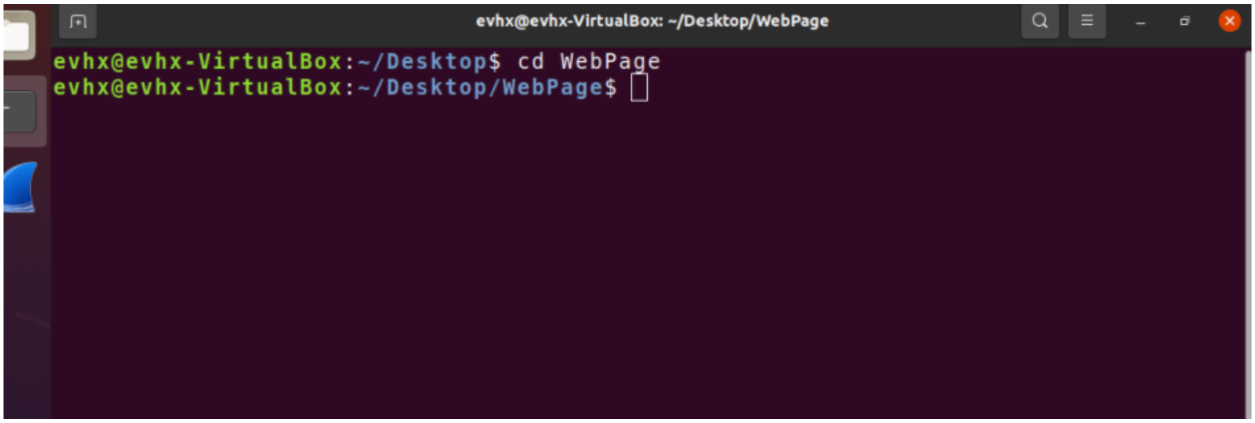
Create a Page for a celebrity, band or business.



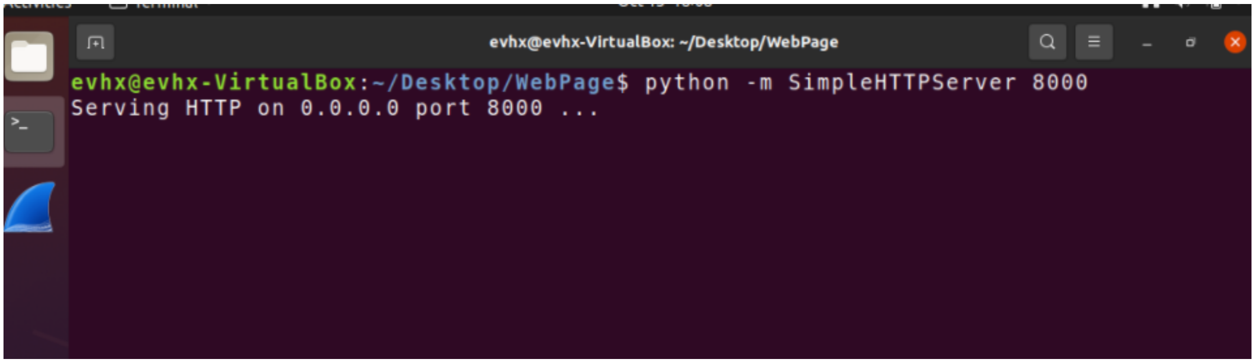
Change the directory to WebPage and start the web server, using the command:

python -m SimpleHTTPServer 8000

Commands used:
python -m SimpleHTTPServer 8000   (web server on port 8000   )



```
evhx@evhx-VirtualBox:~/Desktop$ cd WebPage
evhx@evhx-VirtualBox:~/Desktop/WebPage$ 
```

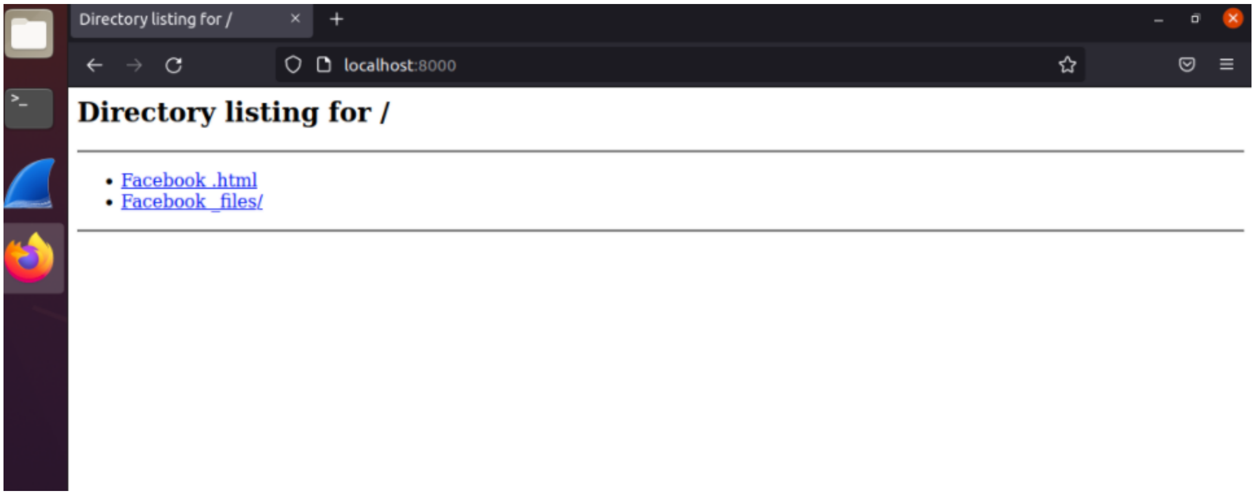


```
evhx@evhx-VirtualBox:~/Desktop/WebPage$ python -m SimpleHTTPServer 8000
Serving HTTP on 0.0.0.0 port 8000 ...
```

To make sure that this webpage located on the port 8000, go into the browser and type:

localhost:8000

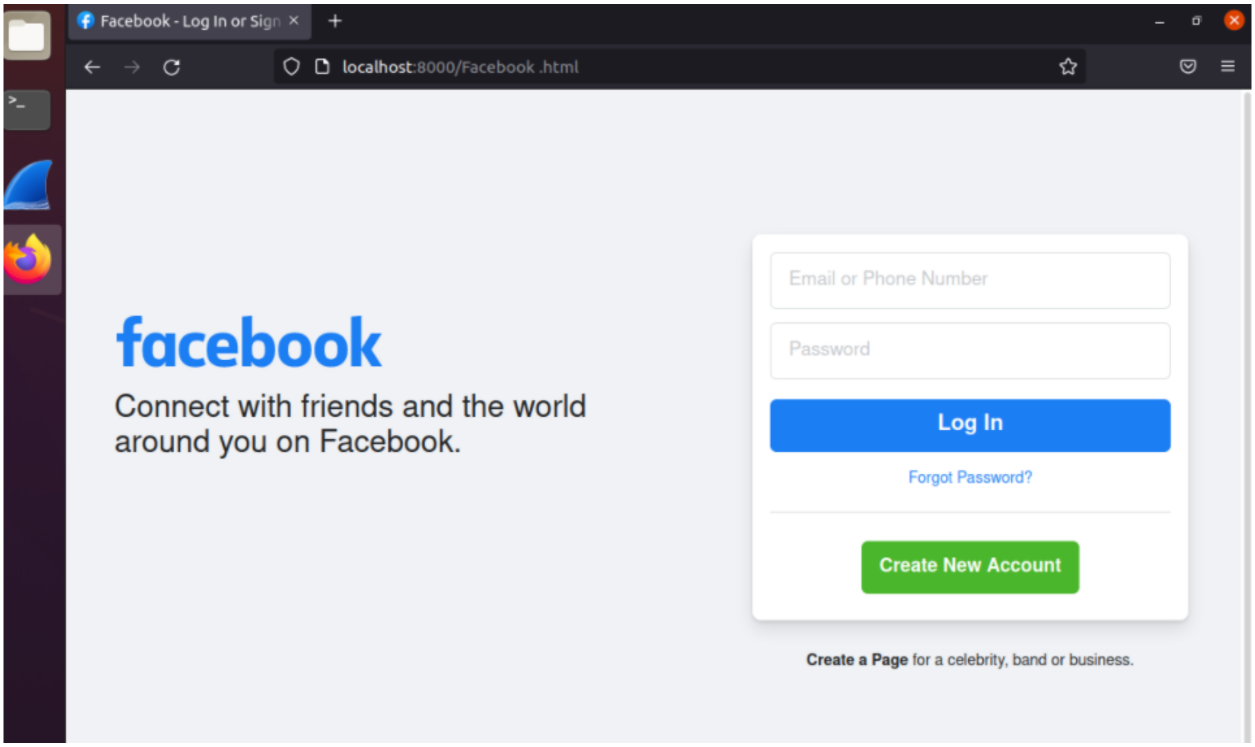
Commands used:
localhost:8000   (local host address on port 8000   )



localhost:8000

Directory listing for /

- Facebook .html
- Facebook _files/



localhost:8000/Facebook .html

Connect with friends and the world around you on Facebook.

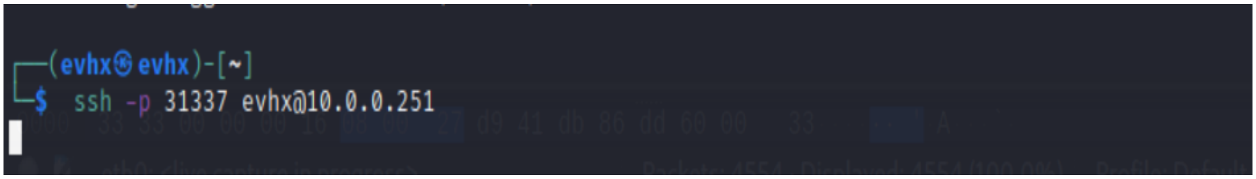
Create a Page for a celebrity, band or business.

First we will connect using SSH from the Kali Linux machine to the Ubuntu machine, by simply using the target ip address and the open port on the Ubuntu machine, using the command

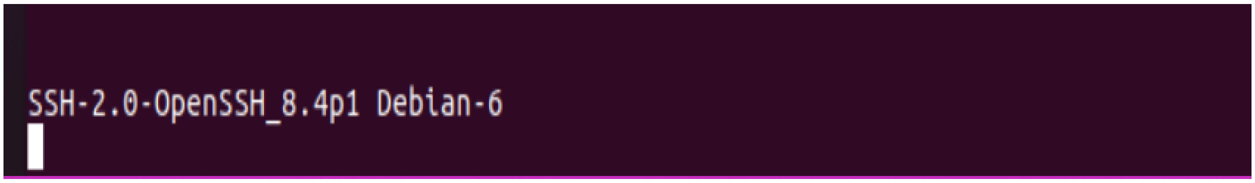
```
ssh -p <port> user@<ip-address-or-hostname>
```

Which reflects as a OpenSSH message in the Ubuntu machine. The packets that specifically come from the Ubuntu machine can be seen in Wireshark by using the filter command: ip.addr==10.0.0.251

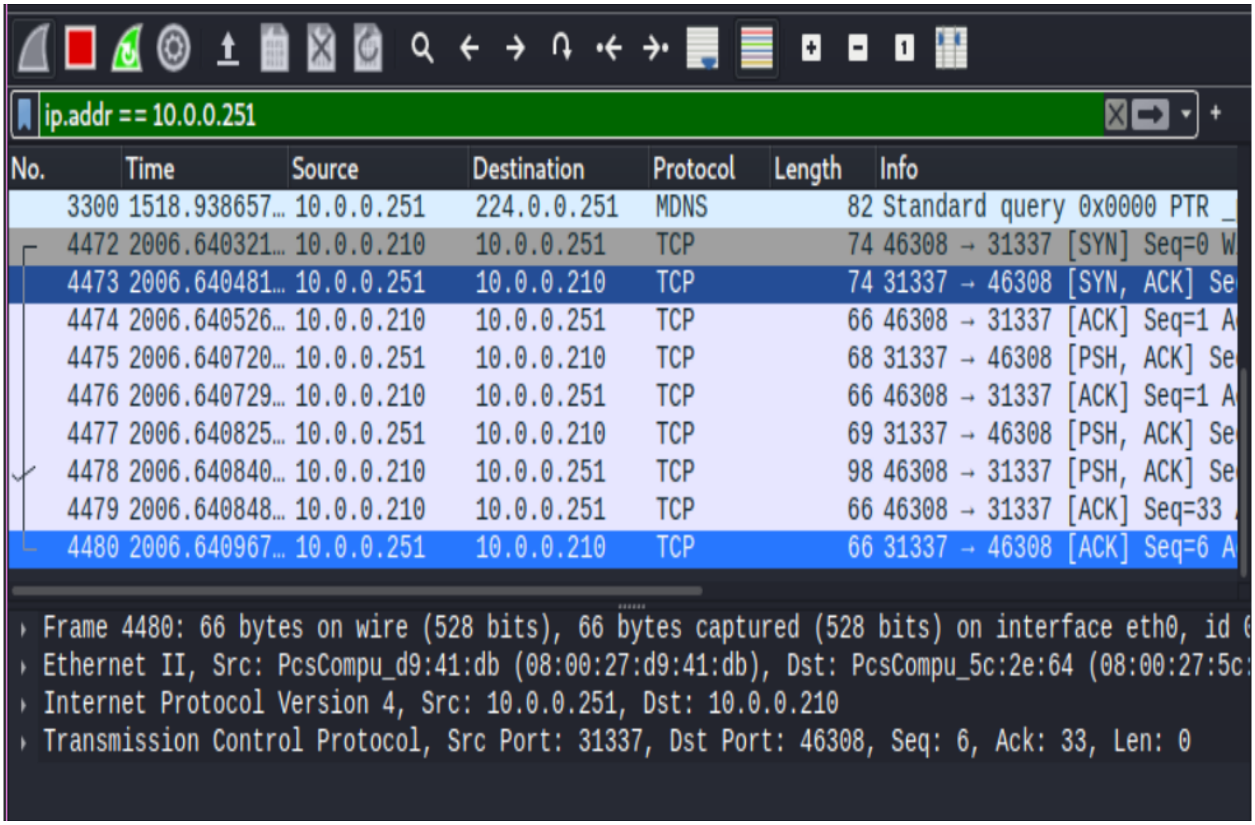
These packets contain data from the messages communicated between machines.



```
┌──(evhx㉿evhx)-[~]
└─$ ssh -p 31337 evhx@10.0.0.251
```



```
SSH-2.0-OpenSSH_8.4p1 Debian-6
```



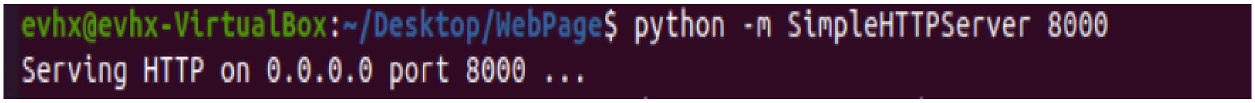
ip.addr == 10.0.0.251

No.	Time	Source	Destination	Protocol	Length	Info
3300	1518.938657…	10.0.0.251	224.0.0.251	MDNS	82	Standard query 0x0000 PTR _
4472	2006.640321…	10.0.0.210	10.0.0.251	TCP	74	46308 → 31337 [SYN] Seq=0 W
4473	2006.640481…	10.0.0.251	10.0.0.210	TCP	74	31337 → 46308 [SYN, ACK] Se
4474	2006.640526…	10.0.0.210	10.0.0.251	TCP	66	46308 → 31337 [ACK] Seq=1 A
4475	2006.640720…	10.0.0.251	10.0.0.210	TCP	68	31337 → 46308 [PSH, ACK] Se
4476	2006.640729…	10.0.0.210	10.0.0.251	TCP	66	46308 → 31337 [ACK] Seq=1 A
4477	2006.640825…	10.0.0.251	10.0.0.210	TCP	69	31337 → 46308 [PSH, ACK] Se
4478	2006.640840…	10.0.0.210	10.0.0.251	TCP	98	46308 → 31337 [PSH, ACK] Se
4479	2006.640848…	10.0.0.210	10.0.0.251	TCP	66	46308 → 31337 [ACK] Seq=33
4480	2006.640967…	10.0.0.251	10.0.0.210	TCP	66	31337 → 46308 [ACK] Seq=6 A

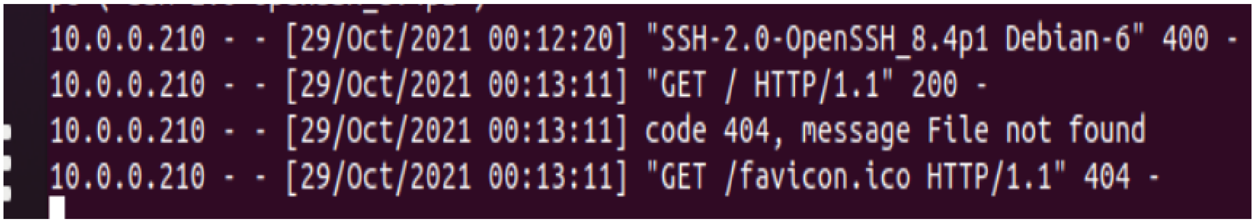
```
› Frame 4480: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0, id 0
› Ethernet II, Src: PcsCompu_d9:41:db (08:00:27:d9:41:db), Dst: PcsCompu_5c:2e:64 (08:00:27:5c:
› Internet Protocol Version 4, Src: 10.0.0.251, Dst: 10.0.0.210
› Transmission Control Protocol, Src Port: 31337, Dst Port: 46308, Seq: 6, Ack: 33, Len: 0
```

To view the webpage from the Kali Linux machine, we have to make sure the previous Ubuntu machine python web server is running first, and then after on the Kali Linux machine, in the browser add the Ubuntu machines ip address, along with the port number the web server is on, in this case would be:

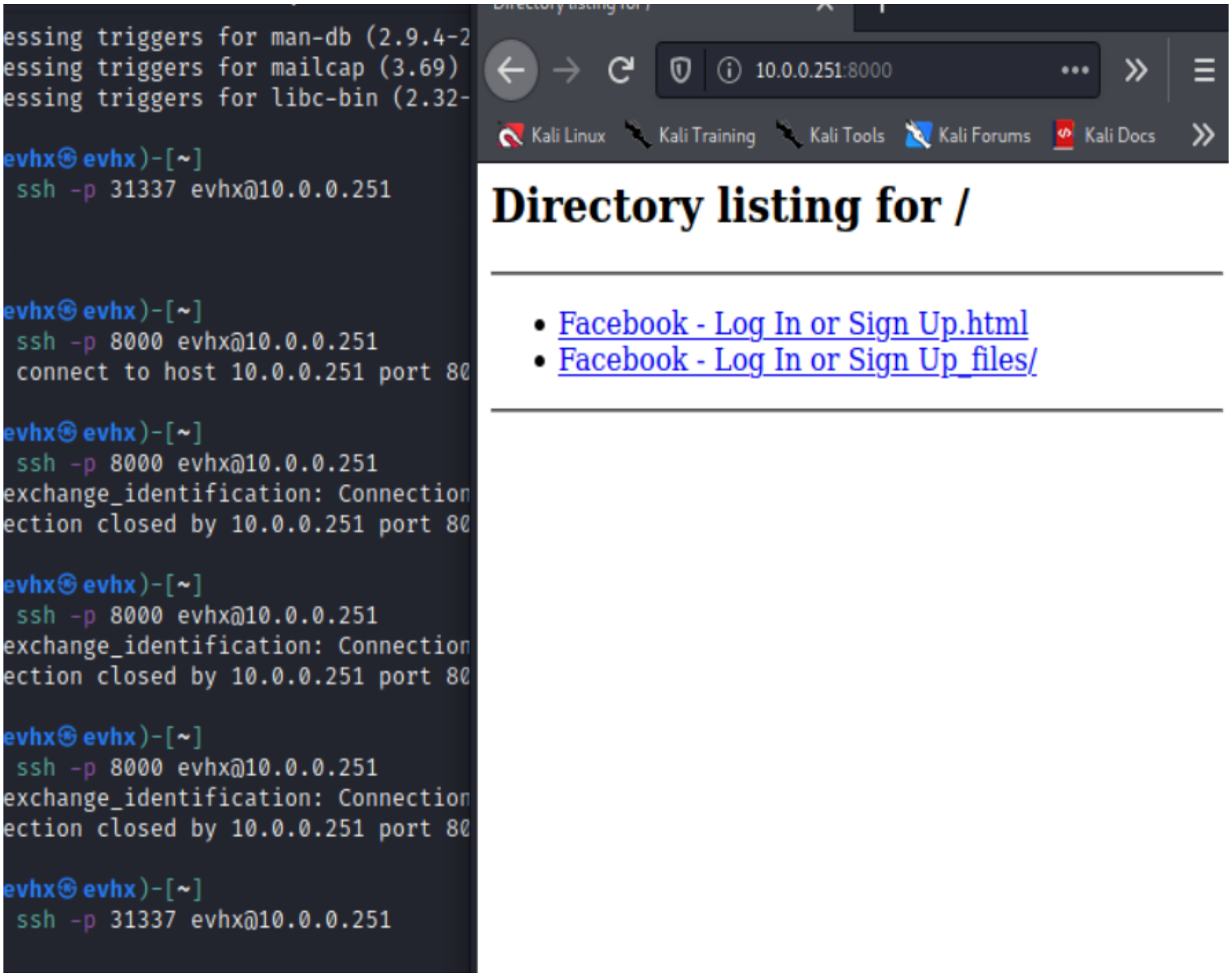
http://10.0.0.251:8000/



```
evhx@evhx-VirtualBox:~/Desktop/WebPage$ python -m SimpleHTTPServer 8000
Serving HTTP on 0.0.0.0 port 8000 ...
```



```
10.0.0.210 - - [29/Oct/2021 00:12:20] "SSH-2.0-OpenSSH_8.4p1 Debian-6" 400 -
10.0.0.210 - - [29/Oct/2021 00:13:11] "GET / HTTP/1.1" 200 -
10.0.0.210 - - [29/Oct/2021 00:13:11] code 404, message File not found
10.0.0.210 - - [29/Oct/2021 00:13:11] "GET /favicon.ico HTTP/1.1" 404 -
```



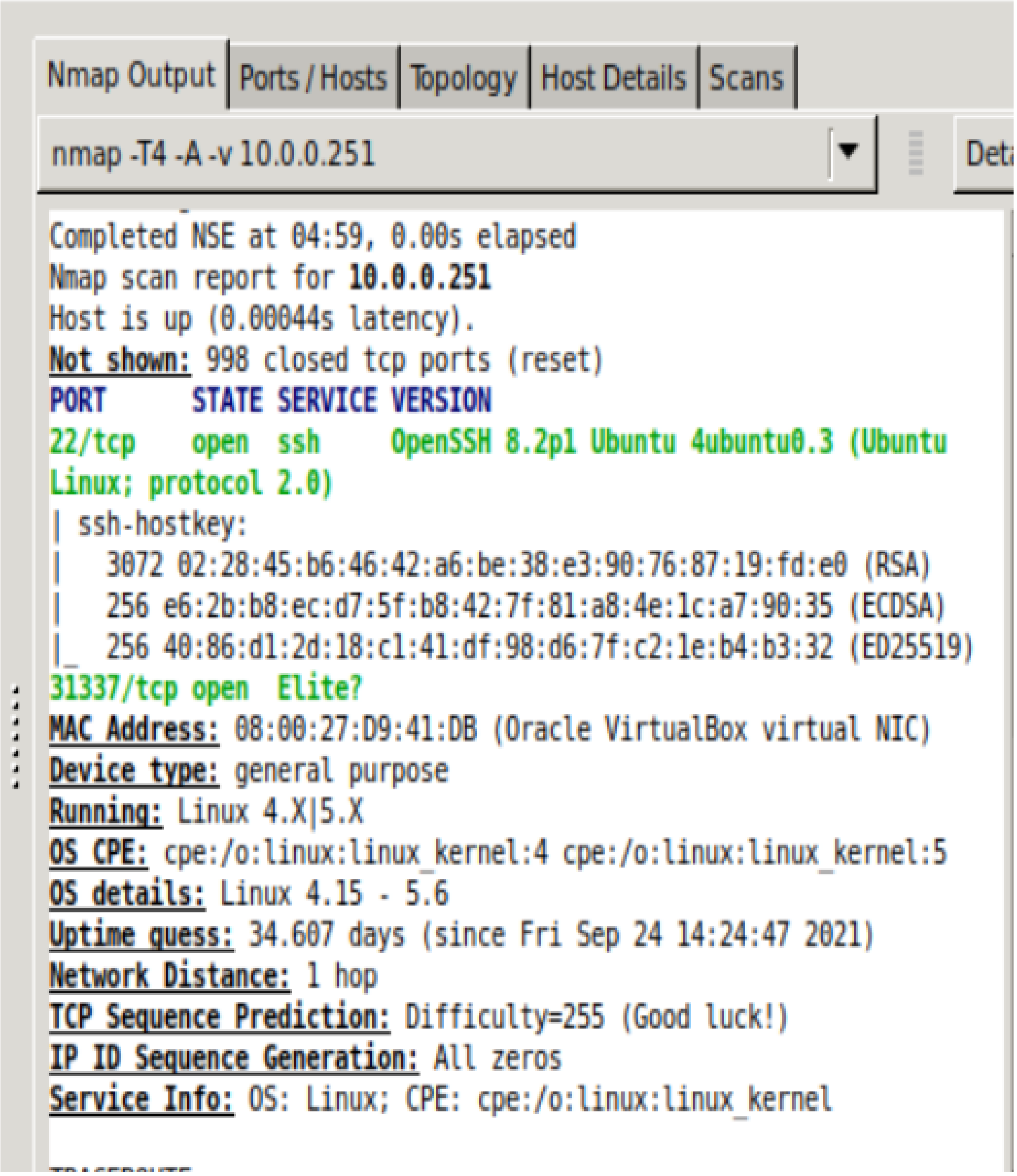
```
essing triggers for man-db (2.9.4-2
essing triggers for mailcap (3.69)
essing triggers for libc-bin (2.32-

evhx㉿evhx)-[~]
 ssh -p 31337 evhx@10.0.0.251


evhx㉿evhx)-[~]
 ssh -p 8000 evhx@10.0.0.251
 connect to host 10.0.0.251 port 80

evhx㉿evhx)-[~]
 ssh -p 8000 evhx@10.0.0.251
exchange_identification: Connection
ection closed by 10.0.0.251 port 80

evhx㉿evhx)-[~]
 ssh -p 8000 evhx@10.0.0.251
exchange_identification: Connection
ection closed by 10.0.0.251 port 80

evhx㉿evhx)-[~]
 ssh -p 8000 evhx@10.0.0.251
exchange_identification: Connection
ection closed by 10.0.0.251 port 80

evhx㉿evhx)-[~]
 ssh -p 31337 evhx@10.0.0.251
```

10.0.0.251:8000

Directory listing for /

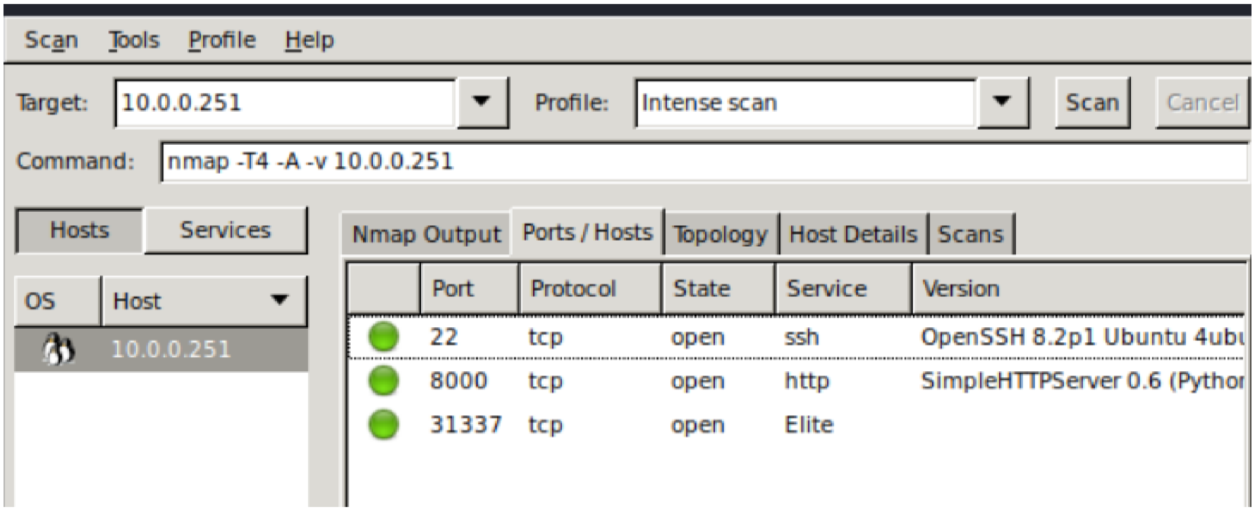
- Facebook - Log In or Sign Up.html
- Facebook - Log In or Sign Up_files/

Connecting to the Ubuntu machine using Zenmap is as simple as adding the target ip address, and then scanning for information. In this case, we used an intense scan, which gives out a multitude of information on the Ubuntu machine.



```
nmap -T4 -A -v 10.0.0.251
```

```
Completed NSE at 04:59, 0.00s elapsed
Nmap scan report for 10.0.0.251
Host is up (0.00044s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu
Linux; protocol 2.0)
| ssh-hostkey:
|   3072 02:28:45:b6:46:42:a6:be:38:e3:90:76:87:19:fd:e0 (RSA)
|   256 e6:2b:b8:ec:d7:5f:b8:42:7f:81:a8:4e:1c:a7:90:35 (ECDSA)
|_  256 40:86:d1:2d:18:c1:41:df:98:d6:7f:c2:1e:b4:b3:32 (ED25519)
31337/tcp open  Elite?
MAC Address: 08:00:27:D9:41:DB (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Uptime guess: 34.607 days (since Fri Sep 24 14:24:47 2021)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=255 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
```



Target: 10.0.0.251

Profile: Intense scan

Command: nmap -T4 -A -v 10.0.0.251

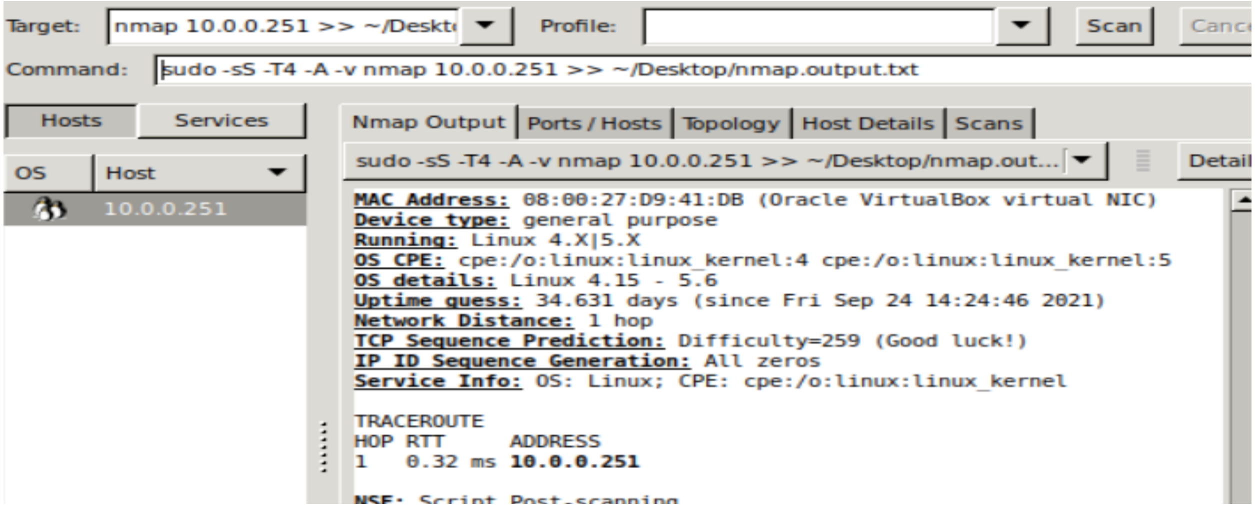
Host: 10.0.0.251

	Port	Protocol	State	Service	Version
●	22	tcp	open	ssh	OpenSSH 8.2p1 Ubuntu 4ubu
●	8000	tcp	open	http	SimpleHTTPServer 0.6 (Pytho
●	31337	tcp	open	Elite	

To retrieve useful information from the target, use the command:
sudo nmap -v -sS -A -T4 10.0.0.251 >> ~/Desktop/nmap.output.txt

Scan a range of ports, here specifically 1-100.
sudo nmap -p 1-100 10.0.0.251 >> ~/Desktop/nmap.output.txt

Scan using TCP connect.
sudo nmap -sT 10.0.0.251 >> ~/Desktop/nmap.output.txt



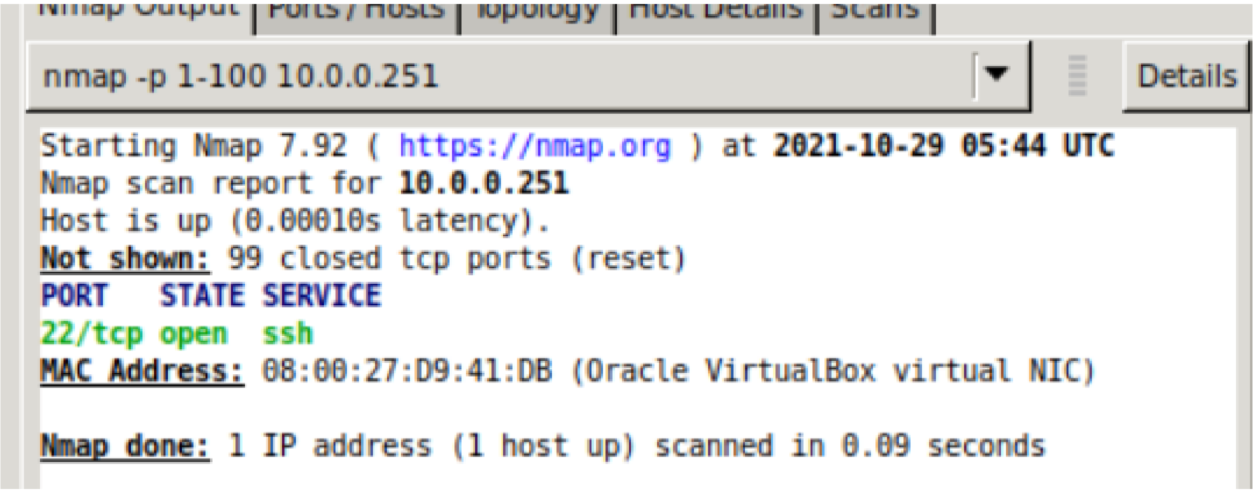
Target: nmap 10.0.0.251 >> ~/Deskt

Command: sudo -sS -T4 -A -v nmap 10.0.0.251 >> ~/Desktop/nmap.output.txt

10.0.0.251

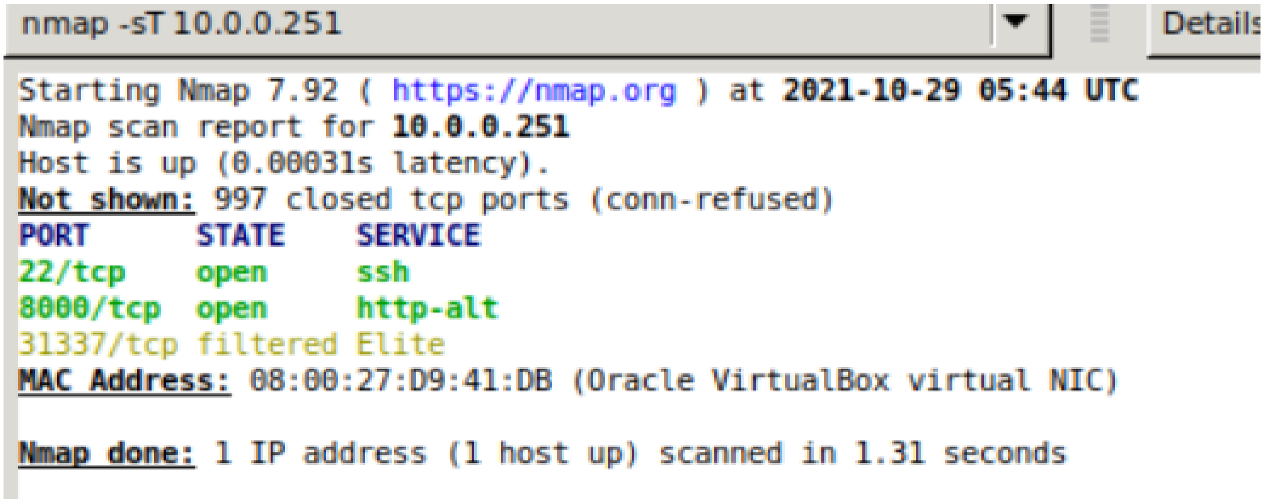
sudo -sS -T4 -A -v nmap 10.0.0.251 >> ~/Desktop/nmap.out...

```
MAC Address: 08:00:27:D9:41:DB (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Uptime guess: 34.631 days (since Fri Sep 24 14:24:46 2021)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=259 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.32 ms 10.0.0.251

NSE: Script Post-scanning
```



nmap -p 1-100 10.0.0.251

```
Starting Nmap 7.92 ( https://nmap.org ) at 2021-10-29 05:44 UTC
Nmap scan report for 10.0.0.251
Host is up (0.00010s latency).
Not shown: 99 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 08:00:27:D9:41:DB (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
```



nmap -sT 10.0.0.251

```
Starting Nmap 7.92 ( https://nmap.org ) at 2021-10-29 05:44 UTC
Nmap scan report for 10.0.0.251
Host is up (0.00031s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT      STATE    SERVICE
22/tcp    open     ssh
8000/tcp  open     http-alt
31337/tcp filtered Elite
MAC Address: 08:00:27:D9:41:DB (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1.31 seconds
```

Conclusion

The main difference between the Nmap packets captured and the Wireshark packets captured is that Nmap is much more organized and precise with the details given out, opposed to Wireshark that provides it all. Although on Wireshark there is still the possibility of filtering these packets, Nmap makes it easier to find port and host information with ease.

Network mapping software affects security in both good and bad ways. It is good because it is a vulnerability scanner and helps with identifying the devices running on their systems, as well as finding open ports that can be used for security risks. On the opposing side, because it is a network mapper, it can also be used for malicious intents, such as finding those open ports in networks.