

CS 491 NETWORK SECURITY

HOMEWORK-2 REPORT

ANIMESH JAIN

UIN:- 669653208

TASK 1 – USING FIREWALL

MACHINE “ANI” (IP address – 10.0.2.4)



```
[04/09/2018 19:22] seed@ubuntu:~$ ifconfig
eth14      Link encap:Ethernet HWaddr 08:00:27:f5:a2:b0
           inet addr:10.0.2.4 Bcast:10.0.2.255 Mask:255.255.255.0
             inet6 addr: fe80::a00:27ff:fe5:a2b0/64 Scope:Link
                   UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                   RX packets:59319 errors:0 dropped:0 overruns:0 frame:0
                   TX packets:22288 errors:0 dropped:0 overruns:0 carrier:0
                   collisions:0 txqueuelen:1000
                   RX bytes:55187757 (55.1 MB) TX bytes:2477256 (2.4 MB)

lo         Link encap:Local Loopback
           inet addr:127.0.0.1 Mask:255.0.0.0
             inet6 addr: ::1/128 Scope:Host
                   UP LOOPBACK RUNNING MTU:16436 Metric:1
                   RX packets:56306 errors:0 dropped:0 overruns:0 frame:0
                   TX packets:56306 errors:0 dropped:0 overruns:0 carrier:0
                   collisions:0 txqueuelen:0
                   RX bytes:45530230 (45.5 MB) TX bytes:45530230 (45.5 MB)

[04/09/2018 19:22] seed@ubuntu:~$
```

MACHINE “MESH” (IP Address – 10.0.2.5)



A screenshot of a Ubuntu desktop environment within a VirtualBox window titled "MESH [Running] - Oracle VM VirtualBox". The desktop background features the classic Ubuntu logo. A terminal window is open in the foreground, showing the output of the command "ifconfig". The terminal window has a dark theme with white text. The status bar at the bottom right shows the date and time as "04/09/2018 19:23" and "7:24 PM".

```
[04/09/2018 19:23] seed@ubuntu:~$ ifconfig
eth14      Link encap:Ethernet HWaddr 08:00:27:20:42:19
           inet addr:10.0.2.5 Bcast:10.0.2.255 Mask:255.255.255.0
           inet6 addr: fe80::a00:27ff:fe20:4219/64 Scope:Link
                  UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                  RX packets:59476 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:71782 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:45667311 (45.6 MB) TX bytes:57093435 (57.0 MB)

lo         Link encap:Local Loopback
           inet addr:127.0.0.1 Mask:255.0.0.0
           inet6 addr: ::1/128 Scope:Host
                  UP LOOPBACK RUNNING MTU:16436 Metric:1
                  RX packets:66 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:66 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:0
                  RX bytes:4812 (4.8 KB) TX bytes:4812 (4.8 KB)

[04/09/2018 19:23] seed@ubuntu:~$
```

Preventing “ANI” from telnetting “MESH”



A screenshot of a Ubuntu desktop environment within a VirtualBox window titled "ANI [Running] - Oracle VM VirtualBox". The desktop background features the classic Ubuntu logo. A terminal window is open in the foreground, showing the output of the command "sudo ufw status". The terminal window has a dark theme with white text. The status bar at the bottom right shows the date and time as "04/09/2018 19:25" and "7:26 PM".

```
[04/09/2018 19:25] seed@ubuntu:~$ sudo ufw status
Status: active
To                         Action      From
--                         ----       ---
10.0.2.5 23                DENY OUT   10.0.2.4

[04/09/2018 19:25] seed@ubuntu:~$
```

In this task the purpose is to prevent machine “ANI” from telnetting machine “MESH”. To do this we give the following commands on the terminal of machine “ANI”.

- Sudo ufw deny out from 10.0.2.4 to 10.0.2.5 port 23

10.0.2.4 – address od machine “ANI”.

10.0.2.5 – address of machine “MESH”

23 – Telnet port

Also we need to enable the firewall and to do so we give

- Sudo ufw enable

Now when we try to telnet machine “MESH” by giving command “telnet 10.0.2.5” on terminal of machine “ANI” we get the following output.



We have to make sure before telnetting that the firewall is enabled. To check we can use “sudo ufw status” or “sudo ufw status numbered “command.

Preventing MESH from Telnetting ANI

This can be achieved in the similar way as we did in the above task. Commands that we use are:-

- Sudo ufw deny out from 10.0.2.5 to 10.0.2.4 port 23
- Sudo ufw enable
- Sudo ufw status
- telnet 10.0.2.4

After we give the above commands we see that telnetting from machine “MESH” to machine “ANI” is not successful as the machine is unable to connect to the remote host. We get the following output.



Preventing ANI from accessing oistbpl.com

Here I am trying to prevent machine “ANI” from accessing a site to Oriental Institute of Technology, Bhopal in India i.e www.oistbpl.com. So in order to do this I will have to add a rule in the firewall for blocking oistbpl.com from machine “ANI” i.e from 10.0.2.4 and since the connection will be http so I will have to block it on port 80.



To add the rule to firewall we need to have the ip address of the site we are blocking. We can find it out by using the following command:-

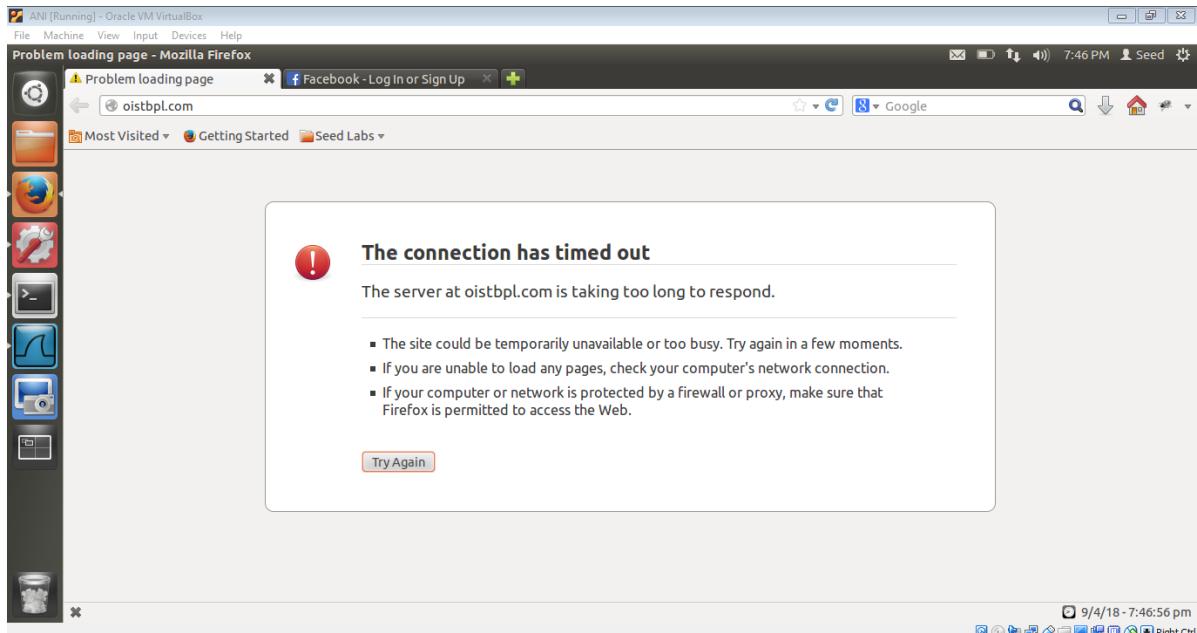
- nslookup oistbpl.com

on doing this we come to know the ip address of oistbpl.com i.e 103.251.24.104

Command that I will use to add the rule to the firewall will be:-

- sudo ufw deny out from 10.0.2.4 to 103.251.24.104 port 80

we will also check the status of the firewall (sudo ufw status) just to be sure whether it is enabled or not. After setting up everything we will try to access oistbpl.com from the Firefox. On doing this I got the following result.

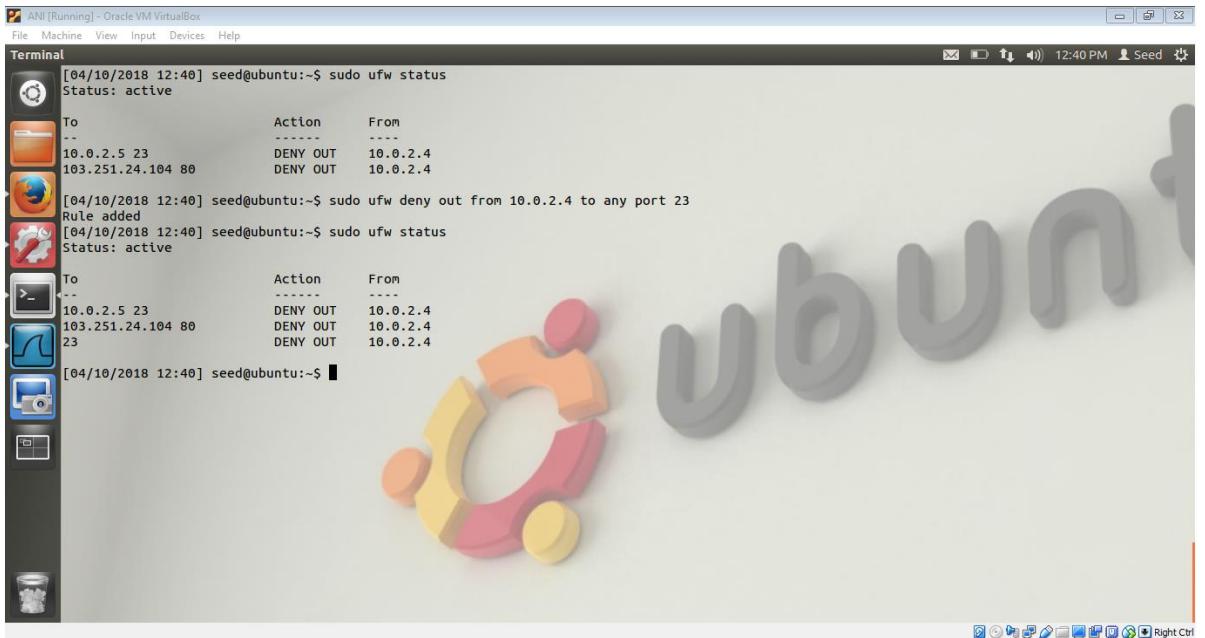


We see that we are unable to access the desired site. Apart from oistbpl.com we can access other sites like www.google.com , www.facebook.com etc. as the rule we added to the firewall just blocks the address that corresponds to www.oistbpl.com .

TASK 2 – EVADING EGRESS FILTERING

Setting up the two block rules:-

1. On machine “ANI” we will block all the outgoing traffic to external telnet servers. Command we use to add this rule to the firewall is:- **sudo ufw deny out from 10.0.2.4 to any port 23**. Also we will introduce a third machine called “JAIN”. We will enable the firewall in machine “ANI” in which we deny all the outgoing traffic to external telnet server. Although we can run telnet server on machine “MESH” and machine “JAIN”. So we will disable the firewall for both machine “MESH” and JAIN” (sudo ufw disable) since there is no restriction on them.



MACHINE “JAIN” (IP address – 10.0.2.6)



2. Blocking all outgoing traffic from machine “ANI” to a website with static IP address. I am again opting to block www.oistbpl.com because it has got a static IP address and the rule to block machine “ANI” from accessing www.oistbpl.com is predefined. Also I deleted the recent search history so as to kill the cached pages.

TASK 2.a:

Egress Filtering is the process of restricting the information flow outsourced from one network to other network. Now the best way to evade the egress filtering is by creating a tunnel. That is exactly what we are going to do here. We will create a tunnel using SSH (Secure Shell) protocol. In this way we can bypass the firewall we created in machine “ANI” without changing any rules. So in a way we

are able to access machine "MESH". Now we since all the firewalls in machine "MESH" and "JAIN" are disabled, we can Telnet machine "JAIN" from machine "MESH".

NOTE:- I have created an empty file in every machine. They are:-

- "this is from ani" – Machine "ANI"
- "this is from mesh" – Machine "MESH"
- "This is JAIN" – Machine "JAIN"

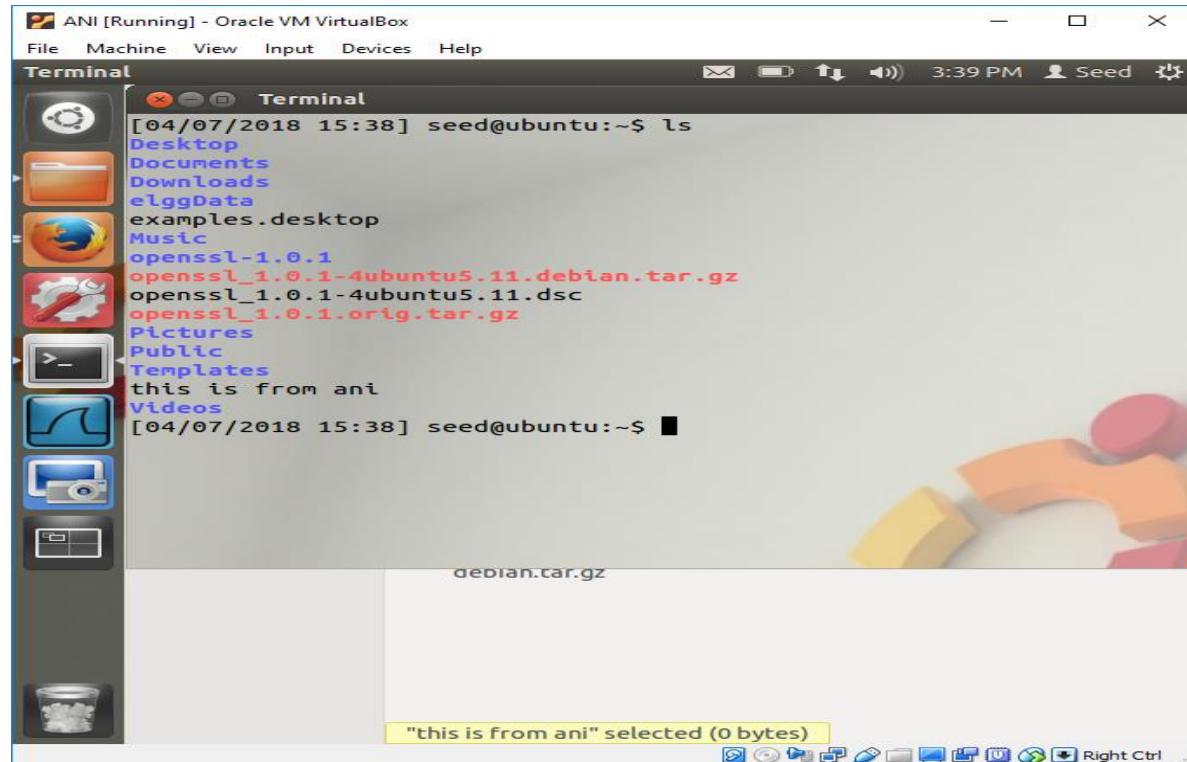
The purpose of creating these empty files in different machine is just to check for which machine I am in after I do SSH and Telnet from one machine to another.

Command that we will use are:-

- sudo ssh seed@10.0.2.5 – for SSH from machine "ANI" to machine "MESH".
- telnet 10.0.2.6 – for Telneting to machine "JAIN" from machine "MESH".

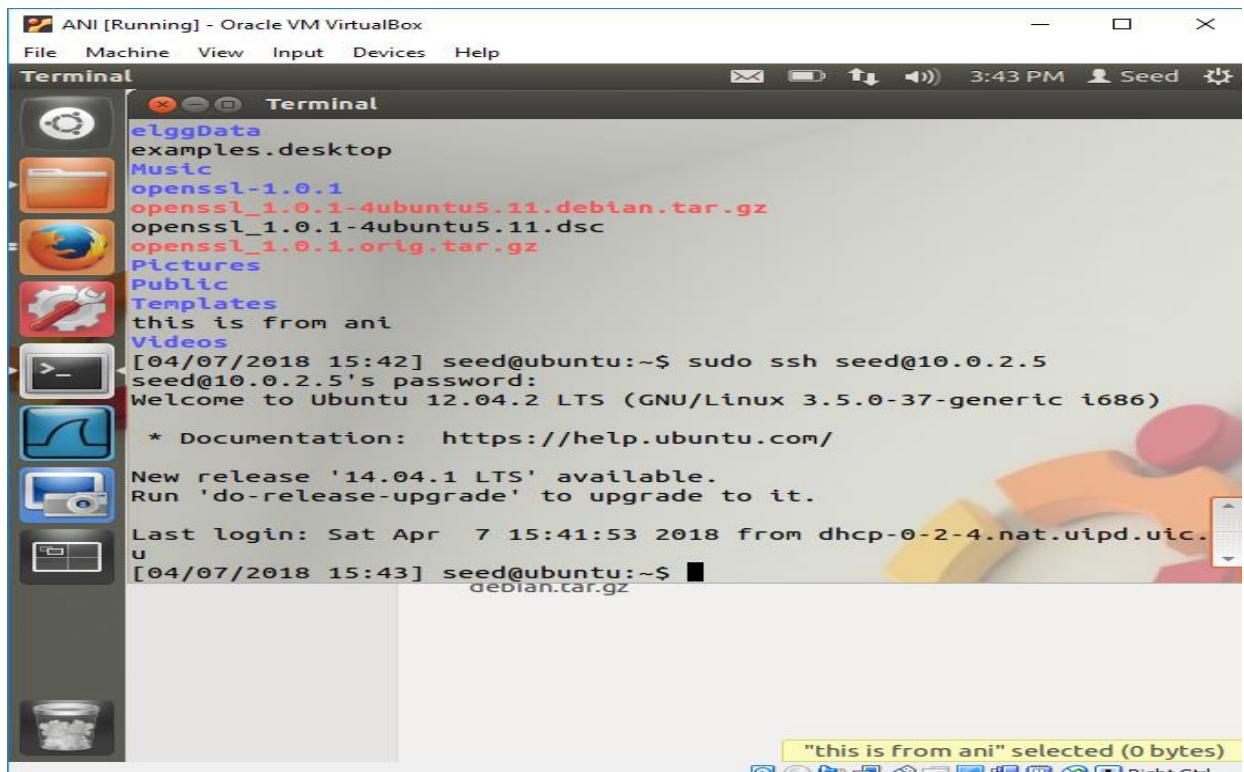
OBSERVATIONS:-

Files in Machine "ANI"

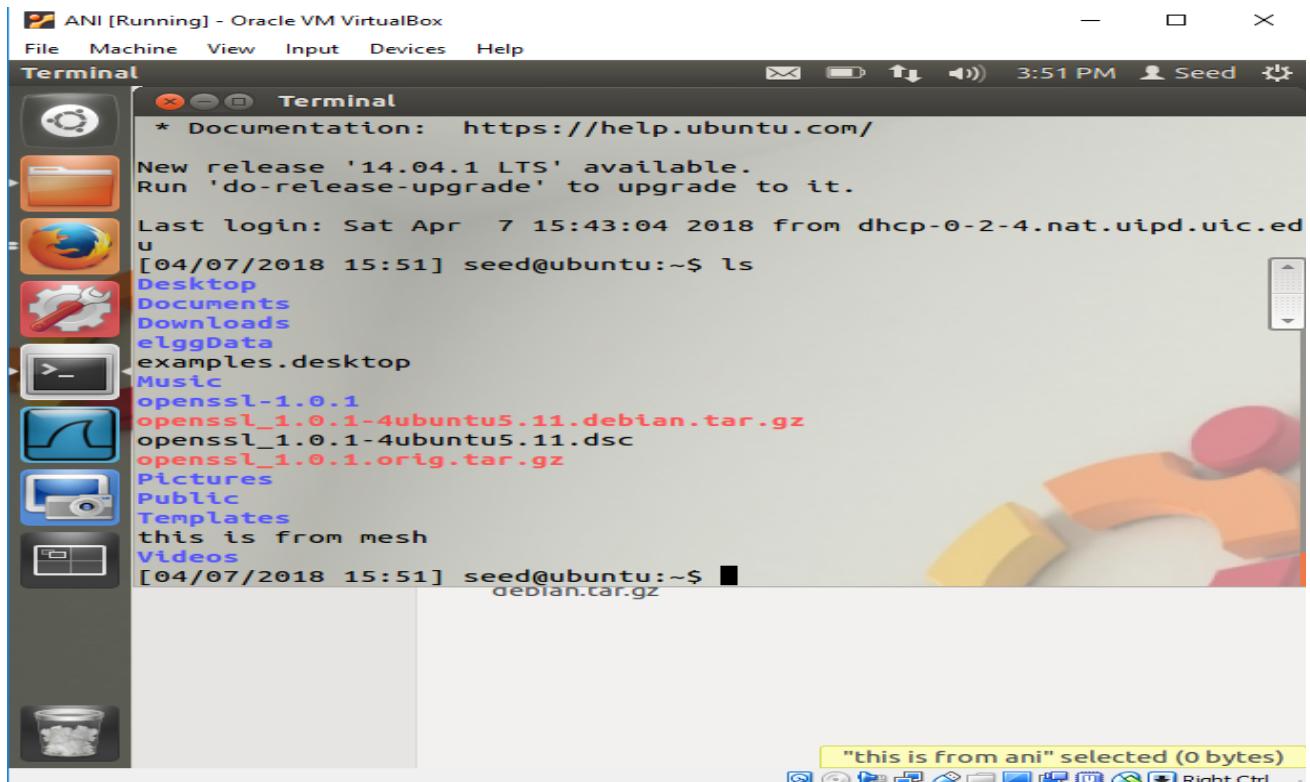


Here we can see that the file "this is from ani" is in machine "ANI"

Machine "ANI" to Machine "MESH" through SSH



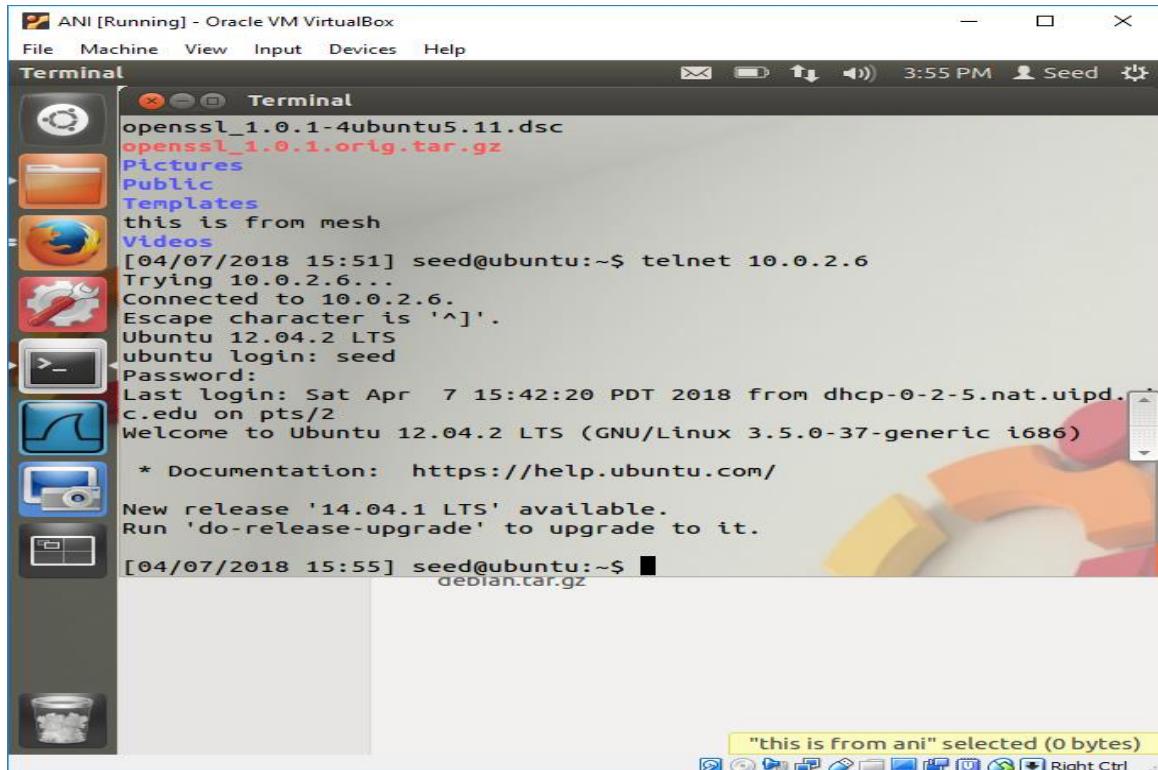
Files in Machine “ANI” after SSH to machine “MESH”



After SSH from machine “ANI” to machine “MESH” we see that the file “this is from mesh” that originally belongs to machine “MESH” is in machine “ANI” i.e I am able to access files in machine

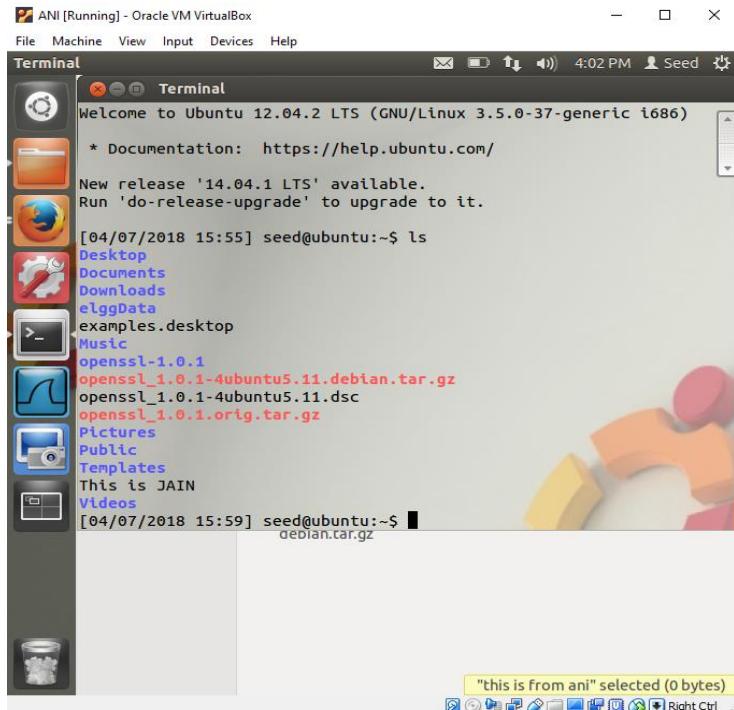
“MESH” from machine “ANI”. Earlier in Task 1 we observed that when Telnet was blocked from machine “ANI” to machine “MESH”, we were not able to gain access to machine “MESH”. But although now we did not change our firewall rules for machine “ANI” but still we are able to evade the firewall and access machine “MESH”. This is because we created a tunnel using SSH protocol.

Machine “MESH” to machine “JAIN” through Telnet



Files in machine “ANI” after Telnet to machine “JAIN” from machine “MESH”

We can see from the following screenshot that there is a file named “This is JAIN”. This is file that belongs to machine “JAIN”. So, it makes clear that machine “ANI” can access the files from machine “JAIN”. So it can be said that packets from machine “JAIN” are reaching to machine “ANI” through Telnet to Machine “MESH” and finally through SSH tunnel.



Capturing Packets through Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
1	2018-04-10 20:22:30.8810.0.2.4	10.0.2.5	TCP	74	55321 > ssh [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK PERM=1 TStamp=4294941193	
2	2018-04-10 20:22:30.8810.0.2.5	10.0.2.4	TCP	66	ssh > 55321 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK PERM=1	
3	2018-04-10 20:22:30.8810.0.2.4	10.0.2.5	TCP	66	55321 > ssh [ACK] Seq=1 Ack=1 Win=14720 Len=0 TStamp=4294941193 TSecr=4528	
4	2018-04-10 20:22:30.8810.0.2.5	10.0.2.4	SSHv2	107	Server Protocol: SSH-2.0-OpenSSH_5.9p1 Debian-SUBUNTU1.1 r	
5	2018-04-10 20:22:30.8810.0.2.4	10.0.2.5	TCP	66	55321 > ssh [ACK] Seq=1 Ack=42 Win=14720 Len=0 TStamp=4294941195 TSecr=452	
6	2018-04-10 20:22:30.8810.0.2.4	10.0.2.5	SSHv2	107	Client Protocol: SSH-2.0-OpenSSH_5.9p1 Debian-SUBUNTU1.1 r	
7	2018-04-10 20:22:30.8810.0.2.5	10.0.2.4	TCP	66	ssh > 55321 [ACK] Seq=42 Ack=42 Win=14592 Len=0 TStamp=452872 TSecr=429494	
8	2018-04-10 20:22:30.8810.0.2.5	10.0.2.4	SSHv2	1050	Server: Key Exchange Init	
9	2018-04-10 20:22:30.8810.0.2.4	10.0.2.5	SSHv2	1338	Client: Key Exchange Init	
10	2018-04-10 20:22:30.9110.0.2.5	10.0.2.4	TCP	66	ssh > 55321 [ACK] Seq=1026 Ack=1314 Win=17408 Len=0 TStamp=452883 TSecr=42	
11	2018-04-10 20:22:30.9110.0.2.4	10.0.2.5	SSHv2	146	Client: Diffie-Hellman Key Exchange Init	
12	2018-04-10 20:22:30.9110.0.2.5	10.0.2.4	TCP	66	ssh > 55321 [ACK] Seq=1026 Ack=1394 Win=17408 Len=0 TStamp=452883 TSecr=42	
13	2018-04-10 20:22:30.9110.0.2.5	10.0.2.4	SSHv2	378	Server: New Keys	
14	2018-04-10 20:22:30.9110.0.2.4	10.0.2.5	SSHv2	82	Client: New Keys	
15	2018-04-10 20:22:30.9110.0.2.5	10.0.2.4	TCP	66	ssh > 55321 [ACK] Seq=1338 Ack=1410 Win=17408 Len=0 TStamp=452897 TSecr=42	
16	2018-04-10 20:22:30.9110.0.2.4	10.0.2.5	TCP	114	[TCP segment of a reassembled PDU]	
17	2018-04-10 20:22:30.9110.0.2.5	10.0.2.4	TCP	66	ssh > 55321 [ACK] Seq=1338 Ack=1458 Win=17408 Len=0 TStamp=452897 TSecr=42	
18	2018-04-10 20:22:30.9110.0.2.5	10.0.2.4	TCP	114	[TCP segment of a reassembled PDU]	
19	2018-04-10 20:22:30.9110.0.2.4	10.0.2.5	TCP	130	[TCP segment of a reassembled PDU]	
20	2018-04-10 20:22:30.9110.0.2.5	128.248.171.50	DNS	81	Standard query PTR 4.2.0.10.in-addr.arpa	
21	2018-04-10 20:22:31.0110.0.2.5	10.0.2.4	TCP	66	ssh > 55321 [ACK] Seq=1386 Ack=1522 Win=17408 Len=0 TStamp=452907 TSecr=42	
22	2018-04-10 20:22:31.01128.248.171.50	10.0.2.5	DNS	294	Standard query response PTR dhcp-0-2-4.nat.uipd.uic.edu	
23	2018-04-10 20:22:31.0110.0.2.5	128.248.171.50	DNS	87	Standard query A dhcp-0-2-4.nat.uipd.uic.edu	
24	2018-04-10 20:22:31.01128.248.171.50	10.0.2.5	DNS	275	Standard query response A 10.0.2.4	
25	2018-04-10 20:22:31.0110.0.2.5	10.0.2.4	TCP	130	[TCP segment of a reassembled PDU]	

ANI [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

eth14 [Wireshark 1.6.7]

Filter: Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Length	Info
200	2018-04-09 17:34:20.5610.0.2.5	10.0.2.6	TCP	66	39890 > telnet [ACK] Seq=93 Ack=424 Win=15744 Len=0 TSval=37351 TSecr=159	
201	2018-04-09 17:34:20.5610.0.2.6	10.0.2.5	TELNET	67	Telnet Data ...	
202	2018-04-09 17:34:20.5610.0.2.5	10.0.2.6	TCP	66	39890 > telnet [ACK] Seq=93 Ack=425 Win=15744 Len=0 TSval=37351 TSecr=159	
203	2018-04-09 17:34:20.5610.0.2.5	10.0.2.4	TCP	146	[TCP segment of a reassembled PDU]	
204	2018-04-09 17:34:20.5610.0.2.4	10.0.2.5	TCP	66	51406 > ssh [ACK] Seq=3010 Ack=3658 Win=24448 Len=0 TSval=77589 TSecr=373	
205	2018-04-09 17:34:20.5610.0.2.6	10.0.2.5	TELNET	69	Telnet Data ...	
206	2018-04-09 17:34:20.5610.0.2.5	10.0.2.6	TCP	66	39890 > telnet [ACK] Seq=93 Ack=428 Win=15744 Len=0 TSval=37351 TSecr=159	
207	2018-04-09 17:34:20.5610.0.2.5	10.0.2.4	TCP	114	[TCP segment of a reassembled PDU]	
208	2018-04-09 17:34:20.5610.0.2.4	10.0.2.5	TCP	66	51406 > ssh [ACK] Seq=3010 Ack=3706 Win=24448 Len=0 TSval=77589 TSecr=373	
209	2018-04-09 17:34:20.5610.0.2.6	10.0.2.5	TELNET	173	Telnet Data ...	
210	2018-04-09 17:34:20.5610.0.2.5	10.0.2.6	TCP	66	39890 > telnet [ACK] Seq=93 Ack=535 Win=15744 Len=0 TSval=37351 TSecr=159	
211	2018-04-09 17:34:20.5610.0.2.5	10.0.2.4	TCP	210	[TCP segment of a reassembled PDU]	
212	2018-04-09 17:34:20.5610.0.2.4	10.0.2.5	TCP	66	51406 > ssh [ACK] Seq=3010 Ack=3850 Win=26496 Len=0 TSval=77589 TSecr=373	
213	2018-04-09 17:34:20.5610.0.2.6	10.0.2.5	TELNET	451	Telnet Data ...	
214	2018-04-09 17:34:20.5610.0.2.5	10.0.2.6	TCP	66	39890 > telnet [ACK] Seq=93 Ack=920 Win=16768 Len=0 TSval=37352 TSecr=159	
215	2018-04-09 17:34:20.5610.0.2.5	10.0.2.4	TCP	498	[TCP segment of a reassembled PDU]	
216	2018-04-09 17:34:20.5610.0.2.4	10.0.2.5	TCP	66	51406 > ssh [ACK] Seq=3010 Ack=4282 Win=28416 Len=0 TSval=77589 TSecr=373	
217	2018-04-09 17:34:20.5610.0.2.6	10.0.2.5	TELNET	68	Telnet Data ...	
218	2018-04-09 17:34:20.5610.0.2.5	10.0.2.6	TCP	66	39890 > telnet [ACK] Seq=93 Ack=922 Win=16768 Len=0 TSval=37352 TSecr=159	
219	2018-04-09 17:34:20.5610.0.2.5	10.0.2.4	TCP	114	[TCP segment of a reassembled PDU]	
220	2018-04-09 17:34:20.5610.0.2.4	10.0.2.5	TCP	66	51406 > ssh [ACK] Seq=3010 Ack=4330 Win=28416 Len=0 TSval=77589 TSecr=373	
221	2018-04-09 17:34:20.5610.0.2.6	10.0.2.5	TELNET	100	Telnet Data ...	
222	2018-04-09 17:34:20.5610.0.2.5	10.0.2.6	TCP	66	39890 > telnet [ACK] Seq=93 Ack=956 Win=16768 Len=0 TSval=37352 TSecr=159	
223	2018-04-09 17:34:20.5610.0.2.5	10.0.2.4	TCP	146	[TCP segment of a reassembled PDU]	
224	2018-04-09 17:34:20.5610.0.2.4	10.0.2.5	TCP	66	51406 > ssh [ACK] Seq=3010 Ack=4410 Win=28416 Len=0 TSval=77590 TSecr=373	

File: "/tmp/wireshark.eth14_2018... Packets: 224 Displayed: 224 Marked: 0 Dropped: 0

Profile: Default

From the above two Wireshark captures we observe that machine "ANI" successfully established SSH connection with machine "MESH" and via machine "MESH" we successfully telneted machine "JAIN".

Task 2.b:

- 1) Accessing oistbpl.com through machine "ANI" once the entire setup is done (i.e after changing the proxy settings and doing SSH from machine "ANI" to machine "MESH". We observe that we are able to access www.oistbpl.com

ANI [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

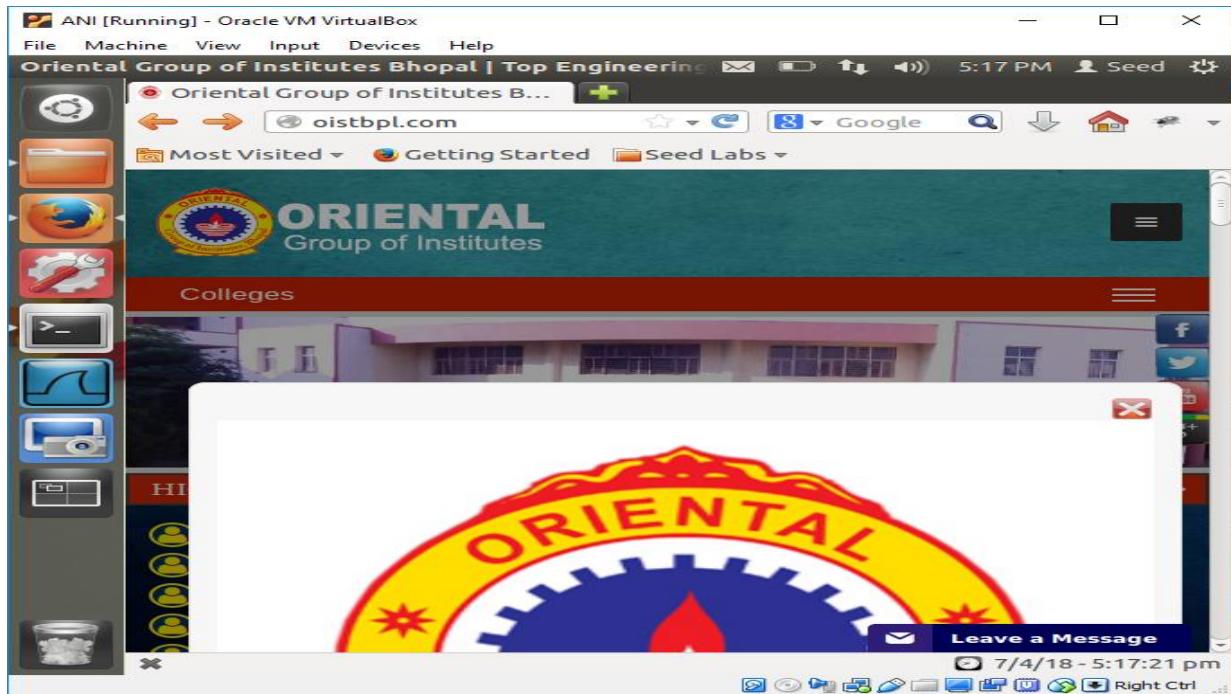
Terminal

```
Terminal
elggData
examples.desktop
Music
openssl-1.0.1
openssl_1.0.1-4ubuntu5.11.debian.tar.gz
openssl_1.0.1-4ubuntu5.11.dsc
openssl_1.0.1.orig.tar.gz
Pictures
Public
Templates
this is from ani
Videos
[04/07/2018 17:09] seed@ubuntu:~$ sudo ssh -D 9000 -C seed@10.0.2.5's password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-37-generic i686)

 * Documentation:  https://help.ubuntu.com/
New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Apr  7 17:02:52 2018 from dhcp-0-2-4.nat.uipd.uic.edu
[04/07/2018 17:10] seed@ubuntu:~$ debian.tar.gz
```

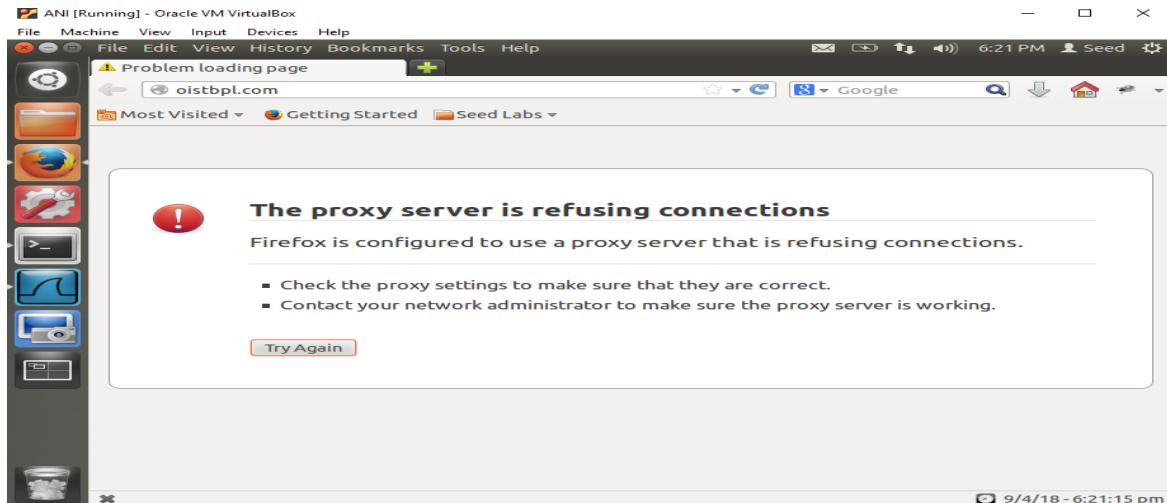
"this is from ani" selected (0 bytes)



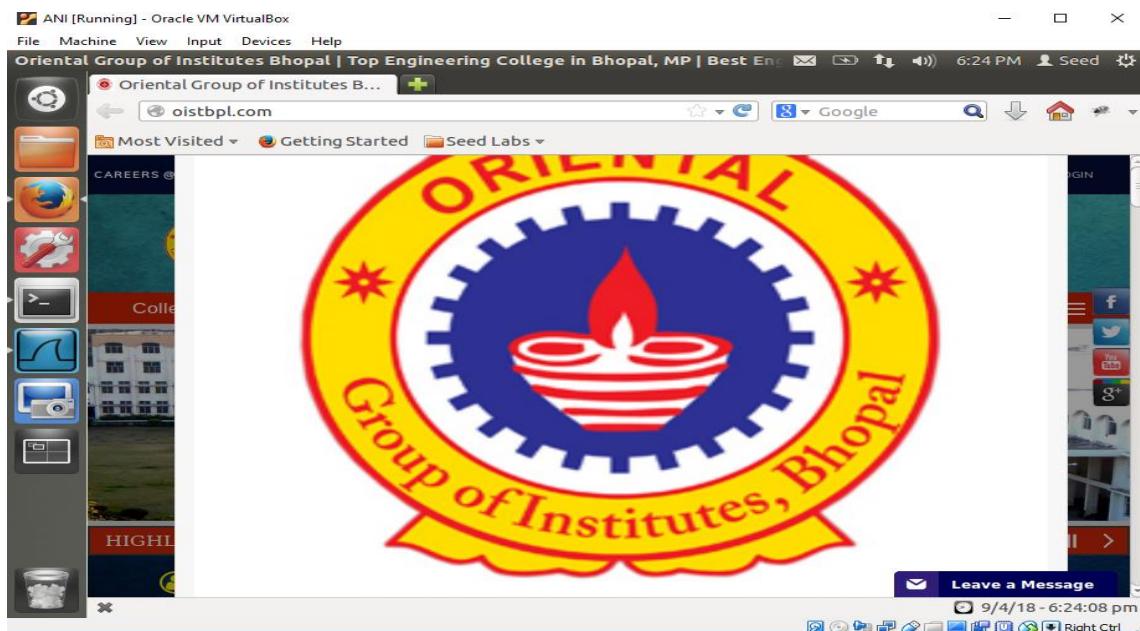
No.	Time	Source	Destination	Protocol	Length	Info
1	2018-04-09 18:48:39.64 CadmusCo 09:34:a8	Broadcast	ARP	60	Who has 10.0.2.3? Tell 10.0.2.6	
2	2018-04-09 18:48:39.64 CadmusCo 09:34:a8	CadmusCo 09:34:a8	ARP	60	10.0.2.3 is at 08:00:27:99:6f:7e	
3	2018-04-09 18:48:39.64 10.0.2.6	10.0.2.3	DHCP	342	DHCP Request - Transaction ID 0x9d0cb83b	
4	2018-04-09 18:48:39.65 10.0.2.3	255.255.255.255	DHCP	590	DHCP ACK - Transaction ID 0x9d0cb83b	
5	2018-04-09 18:48:40.87 10.0.2.4	10.0.2.5	TCP	74	52281 > ssh [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK Perm=1 TSval=119266 TSecr=1152812	
6	2018-04-09 18:48:40.87 10.0.2.5	10.0.2.4	TCP	74	ssh > 52281 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK Perm=1 TSval=119266 TSecr=1152812	
7	2018-04-09 18:48:40.87 10.0.2.4	10.0.2.5	TCP	66	52281 > ssh [ACK] Seq=1 Ack=1 Win=14720 Len=0 TSval=1192660 TSecr=1152812	
8	2018-04-09 18:48:40.87 10.0.2.5	10.0.2.4	SSHV2	107	Server Protocol: SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1.1v1r	
9	2018-04-09 18:48:40.87 10.0.2.4	10.0.2.5	TCP	66	52281 > ssh [ACK] Seq=1 Ack=42 Win=14720 Len=0 TSval=1192662 TSecr=1152812	
10	2018-04-09 18:48:40.87 10.0.2.4	10.0.2.5	SSHV2	107	Client Protocol: SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1.1v1r	
11	2018-04-09 18:48:40.87 10.0.2.5	10.0.2.4	TCP	66	ssh > 52281 [ACK] Seq=42 Ack=42 Win=14592 Len=0 TSval=1152813 TSecr=11926	
12	2018-04-09 18:48:40.87 10.0.2.5	10.0.2.4	SSHV2	1050	Server: Key Exchange Init	
13	2018-04-09 18:48:40.87 10.0.2.4	10.0.2.5	SSHV2	1338	Client: Key Exchange Init	
14	2018-04-09 18:48:40.91 10.0.2.5	10.0.2.4	TCP	66	ssh > 52281 [ACK] Seq=1026 Ack=1314 Win=17408 Len=0 TSval=1152824 TSecr=1152812	
15	2018-04-09 18:48:40.91 10.0.2.4	10.0.2.5	SSHV2	146	Client: Diffie-Hellman Key Exchange Init	
16	2018-04-09 18:48:40.91 10.0.2.5	10.0.2.4	TCP	66	ssh > 52281 [ACK] Seq=1026 Ack=1394 Win=17408 Len=0 TSval=1152824 TSecr=1152812	
17	2018-04-09 18:48:40.91 10.0.2.5	10.0.2.4	SSHV2	378	Server: New Keys	
18	2018-04-09 18:48:40.91 10.0.2.4	10.0.2.5	SSHV2	82	Client: New Keys	
19	2018-04-09 18:48:40.91 10.0.2.5	10.0.2.4	TCP	66	ssh > 52281 [ACK] Seq=1338 Ack=1410 Win=17408 Len=0 TSval=1152837 TSecr=1152812	
20	2018-04-09 18:48:40.91 10.0.2.4	10.0.2.5	TCP	114	[TCP segment of a reassembled PDU]	
21	2018-04-09 18:48:40.91 10.0.2.5	10.0.2.4	TCP	66	ssh > 52281 [ACK] Seq=1338 Ack=1458 Win=17408 Len=0 TSval=1152837 TSecr=1152812	
22	2018-04-09 18:48:40.91 10.0.2.5	10.0.2.4	TCP	114	[TCP segment of a reassembled PDU]	
23	2018-04-09 18:48:40.91 10.0.2.4	10.0.2.5	TCP	130	[TCP segment of a reassembled PDU]	
24	2018-04-09 18:48:40.97 10.0.2.5	128.248.171.50	DNS	81	Standard query PTR 4.2.0.10.in-addr.arpa	
25	2018-04-09 18:48:41.06 10.0.2.5	10.0.2.4	TCP	66	ssh > 52281 [ACK] Seq=1386 Ack=1522 Win=17408 Len=0 TSval=1152847 TSecr=1152812	

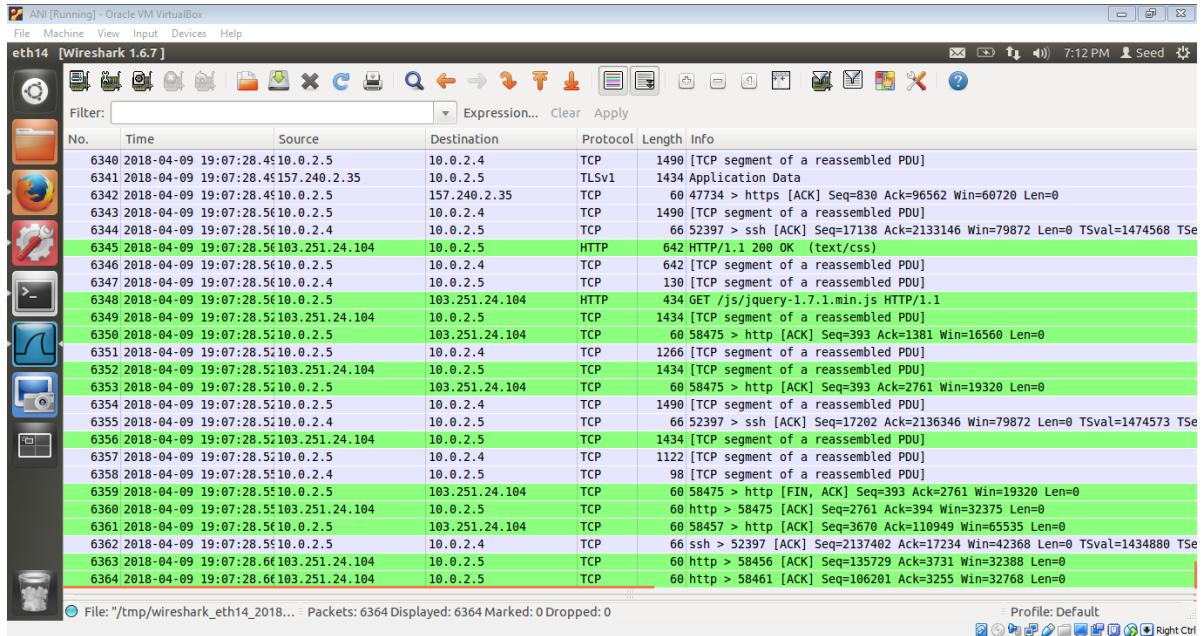
From the Wireshark pictures we observe that SSH connection between Machine “ANI” and machine “MESH” is established.

2) After breaking the SSH tunnel and clearing the Firefox cache we observe that on searching for oistbpl.com on machine “ANI” we are not able to access it as the proxy server refuses the connection.



3) On establishing the SSH tunnel connection again we observe that we are again able to access www.oistbpl.com on machine “ANI”.





From the above wireshark capture we observe that the HTTP request is successful and that is why we are able to access the website again.

TASK 3

WEB PROXY (APPLICATION FIREWALL)

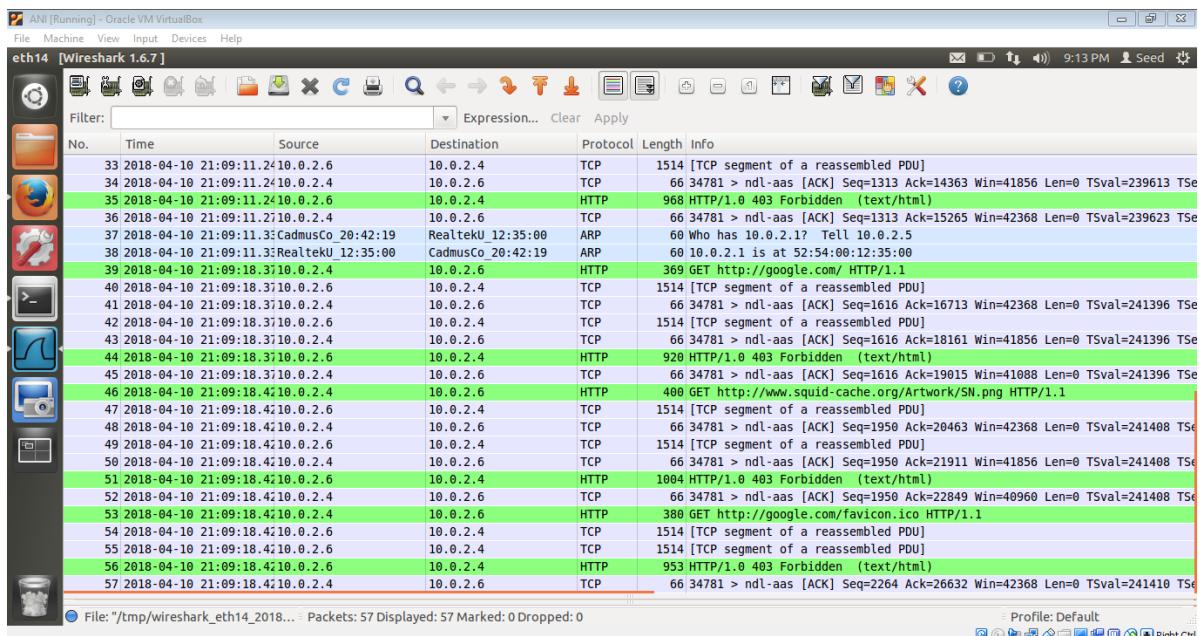
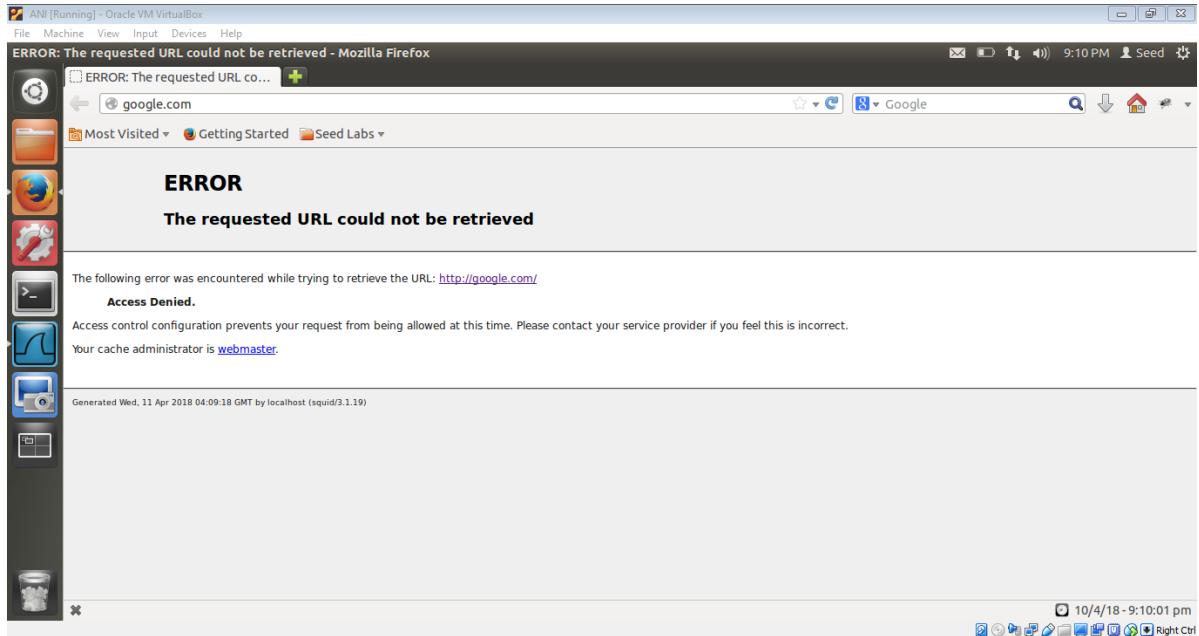
In this task we have to setup a web proxy. In this task we are using a free web proxy software called Squid. Commands we use are:-

- sudo apt-get install squid (if the software is not installed already)
- sudo service squid3 start (to start the server)
- sudo service squid3 restart (to restart the server)

After locating the squid.conf file we can start changing the firewall policies according to our will. Only thing we have to keep in our mind is to restart the configuration file every time after making changes to it. On machine “ANI” we will restrict the browsing behavior. We will run the web proxy on machine “MESH”. We will configure the Firefox setting in machine “ANI” as told in question manual.

OBSERVATIONS:-

- 1). After the entire setup when I try to access www.google.com on machine “ANI”’s Firefox browser we see we are not able to access it. The result that we get is in the following screenshot.



From the wireshark we observe that all the HTTP request from machine “ANI” is being blocked by the squid server in machine “MESH”. It is giving a message 403 Forbidden.

2). When we have a look at the configuration file (under the http_access tag), we see that http access is denied to all the sites. So this is the reason why we were not able to access www.google.com in the previous case.

`http_access deny all` – denies http access to all the sites.



```
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
# Example rule allowing access from your local networks.
# Adapt localnet in the ACL section to list your (internal) IP networks
# from where browsing should be allowed
#http_access allow localnet
#http_access allow localhost
#
# And finally deny all other access to this proxy
http_access deny all
#
# TAG: adapted_http_access
#   Allowing or Denying access based on defined access lists
#
# Essentially identical to http_access, but runs after redirectors
# and ICAP/eCAP adaptation. Allowing access control based on their
# output.
#
# If not set then only http_access is used.
#Default:
#none
#
# TAG: http_reply_access
#   Allow replies to client requests. This is complementary to http_access.
#
# http_reply_access allow|deny [...] aclname ...
#
# NOTE: if there are no access lines present, the default is to allow
# all replies
```

GNOME Terminal - IAIN [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminal

GNU nano 2.2.6 File: /etc/squid3/squid.conf

9:14 PM Seed

Get Help WriteOut Read File Prev Page Cut Text Cur Pos

Exit Justify Where Is Next Page Uncut Text To Spell Right Ctrl

3). Making changes to the configuration file such that all the websites are allowed. We will replace **http_access deny all** by **http_access allow all**. This way we can access www.google.com as well as all other websites.



```
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
# Example rule allowing access from your local networks.
# Adapt localnet in the ACL section to list your (internal) IP networks
# from where browsing should be allowed
#http_access allow localnet
#http_access allow localhost
#
# And finally deny all other access to this proxy
http_access allow all
#
# TAG: adapted_http_access
#   Allowing or Denying access based on defined access lists
#
# Essentially identical to http_access, but runs after redirectors
# and ICAP/eCAP adaptation. Allowing access control based on their
# output.
#
# If not set then only http_access is used.
#Default:
#none
#
# TAG: http_reply_access
#   Allow replies to client requests. This is complementary to http_access.
#
# http_reply_access allow|deny [...] aclname ...
#
# NOTE: if there are no access lines present, the default is to allow
# all replies
```

GNOME Terminal - IAIN [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminal

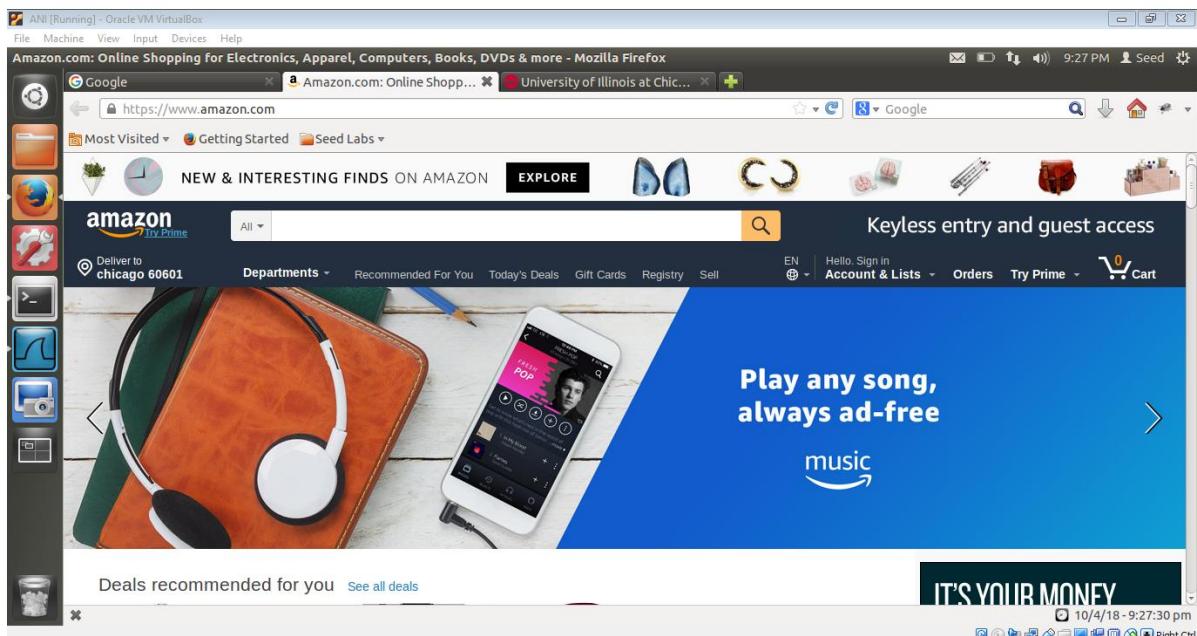
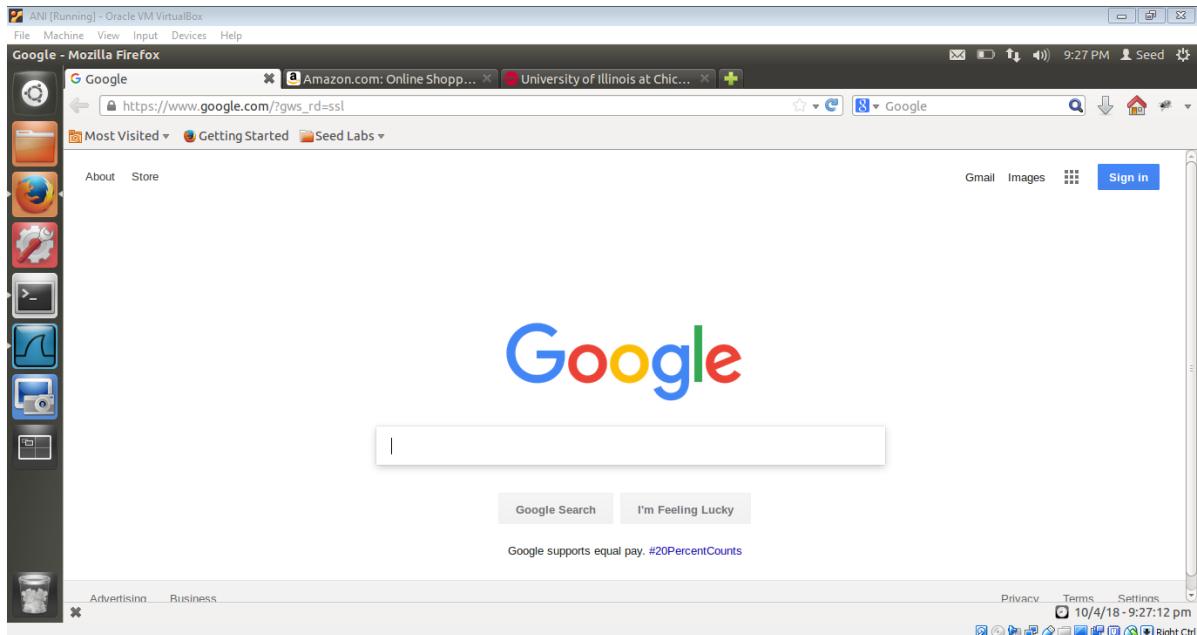
GNU nano 2.2.6 File: /etc/squid3/squid.conf

9:15 PM Seed Modified

Get Help WriteOut Read File Prev Page Cut Text Cur Pos

Exit Justify Where Is Next Page Uncut Text To Spell Right Ctrl

After making changes we can access google.com. Here we observe that along with www.google.com we are able to access other websites as well like www.amazon.com etc.



4). Making changes to the configuration file such that we allow access to google.com and blocking all other sites. Here we observe that we are able to block just www.google.com. Other sites we were able to access in the previous case are no more accessible.

[ANI [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminal

GNU nano 2.2.6 File: /etc/squid3/squid.conf Modified

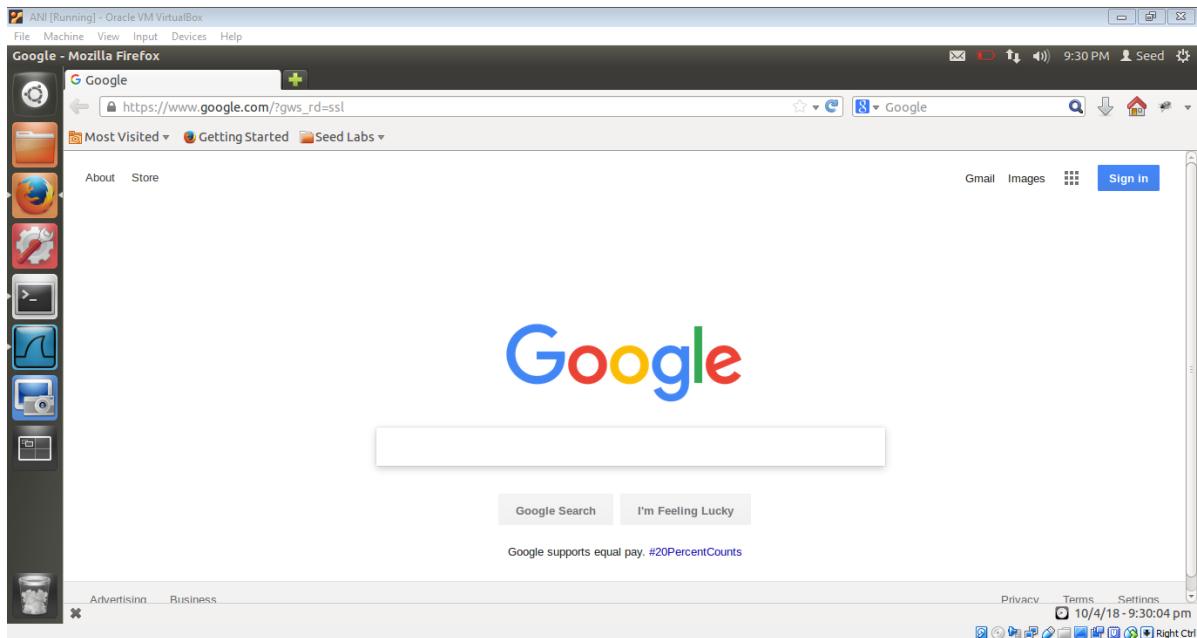
```
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
# Example rule allowing access from your local networks.
# Adapt localnet in the ACL section to list your (internal) IP networks
# from where browsing should be allowed
#http_access allow localnet
http_access allow localhost

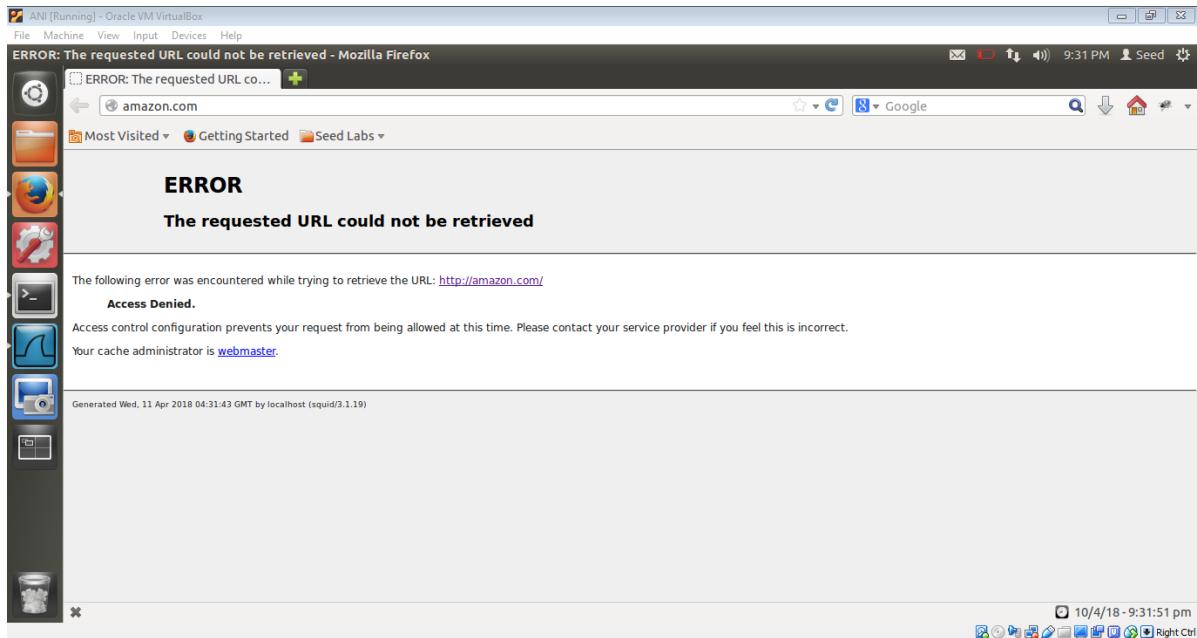
# And finally deny all other access to this proxy
acl access_google dstdomain google.com
http_access allow access_google
http_access deny all

# TAG: adapted_http_access
#     Allowing or Denying access based on defined access lists
#
# Essentially identical to http_access, but runs after redirectors
# and ICAP/eCAP adaptation. Allowing access control based on their
# output.
#
# If not set then only http_access is used.
#Default:
# none

# TAG: http_reply_access
#     Allow replies to client requests. This is complementary to http_access.
#
# http_reply_access allow|deny [...] aclname ...

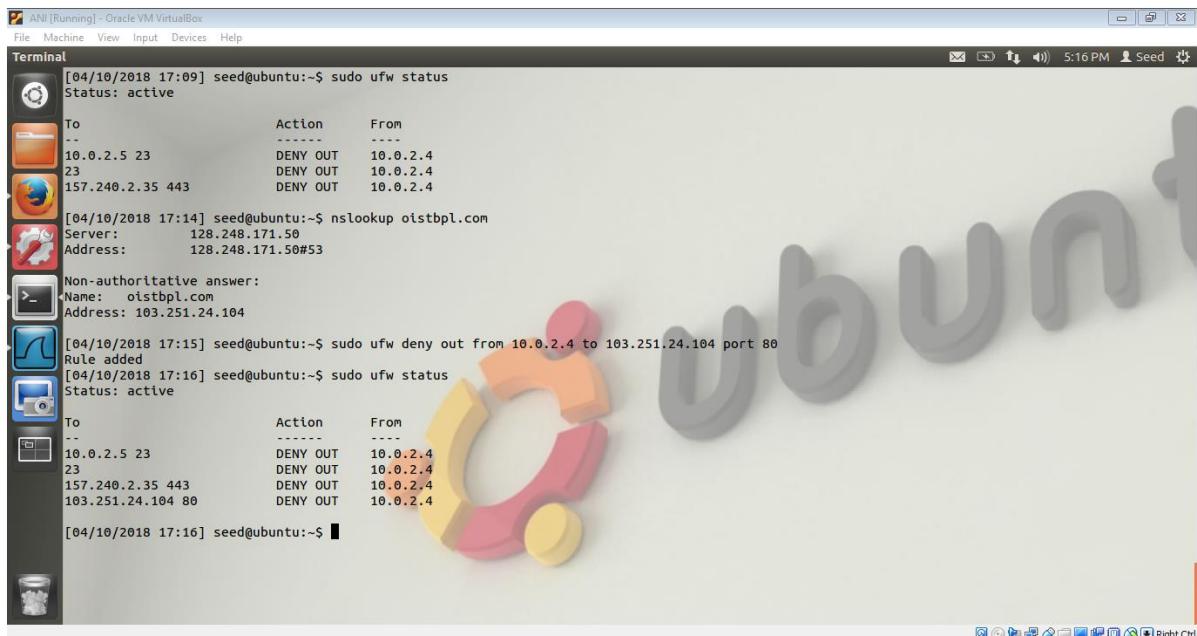
^C Get Help ^O WriteOut ^R Read File ^Y Prev Page ^X Cut Text
^X Exit ^J Justify ^W Where Is ^V Next Page ^U Uncut Text ^A Cur Pos
^T To Spell ^I Right Ctrl
```

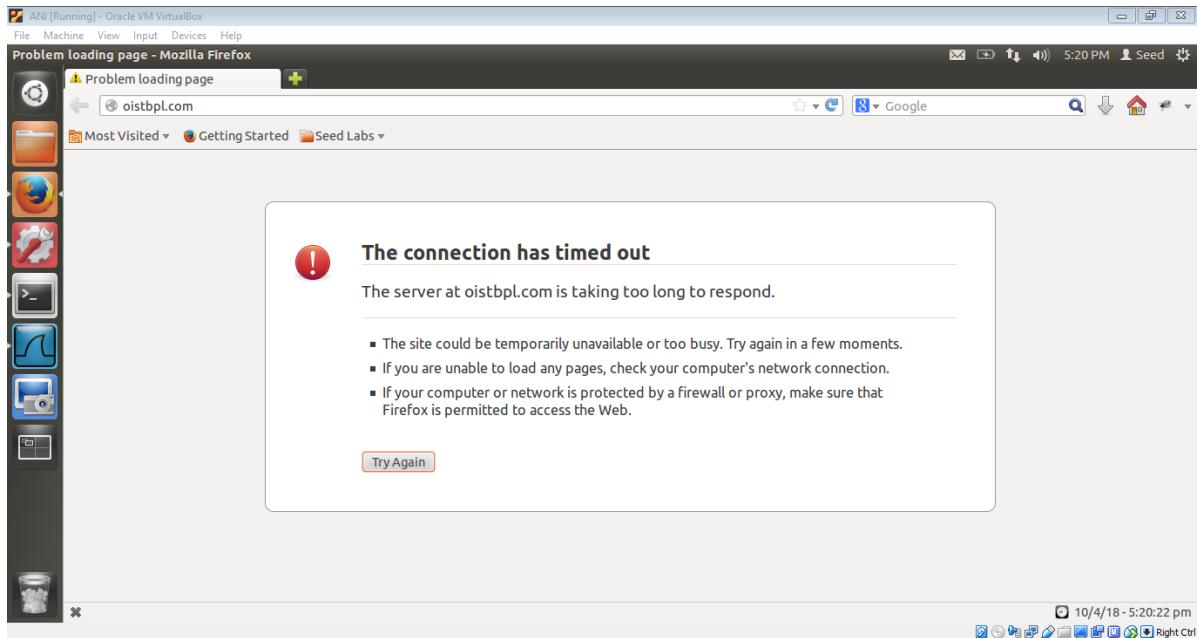




Task 3.b: Using Web Proxy to evade Firewall.

In this task we will add a rule on machine “ANI” to block www.oistbpl.com using **sudo ufw deny out** from **10.0.2.4** to **103.251.24.104** port **80** command.





Now we want to access www.oistbpl.com on machine “ANI” using web proxy. So we will change the configuration in squid.conf file on machine “JAIN” such that we are able to access www.oistbpl.com even when we have blocked it on machine “ANI” using ufw command. We will give the command **http_access allow all** to do so. After doing this we observe that we are able to access www.oistbpl.com on machine “ANI”.

The screenshot shows a terminal window titled "JAIN [Running] - Oracle VM VirtualBox". The title bar indicates the file is "/etc/squid3/squid.conf". The terminal content displays the configuration file for the Squid proxy. The "http_access" section is modified to allow all access:

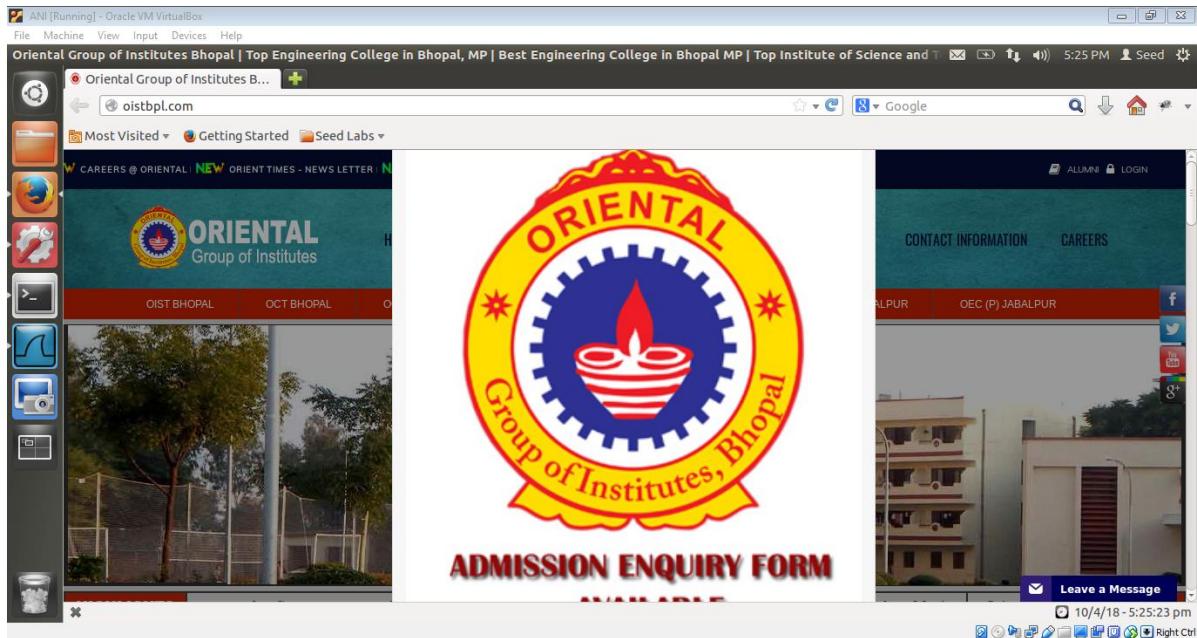
```
# from where browsing should be allowed
#http_access allow localnet
http_access allow localhost

# And finally deny all other access to this proxy
http_access allow all

# TAG: adapted_http_access
#   Allowing or Denying access based on defined access lists
#
#   Essentially identical to http_access, but runs after redirectors
#   and ICAP/eCAP adaptation. Allowing access control based on their
#   output.
#
#   if not set then only http_access is used.
#Default:
# none

# TAG: http_reply_access
#   Allow replies to client requests. This is complementary to http_access.
#
#   http_reply_access allow|deny [!] aclname ...
#
# NOTE: if there are no access lines present, the default is to allow
#       all replies
#
# If none of the access lines cause a match the opposite of the
# last line will apply. Thus it is good practice to end the rules
# with an "allow all" or "deny all" entry.
#
```

The bottom of the terminal shows the nano editor's command bar with options like Get Help, WriteOut, Read File, Prev Page, Next Page, Cut Text, Uncut Text, Cur Pos, To Spell, and Exit.



Question 3: - If ufw blocks the tcp port 3128 , can you still use web proxy to evade the firewall?

Answer 3: - Yes we can still use web proxy to evade firewall even if the ufw blocks the tcp port 3128. Squid uses the port 3128 as the default port. If it gets blocked then we can change the port in the squid.conf file.

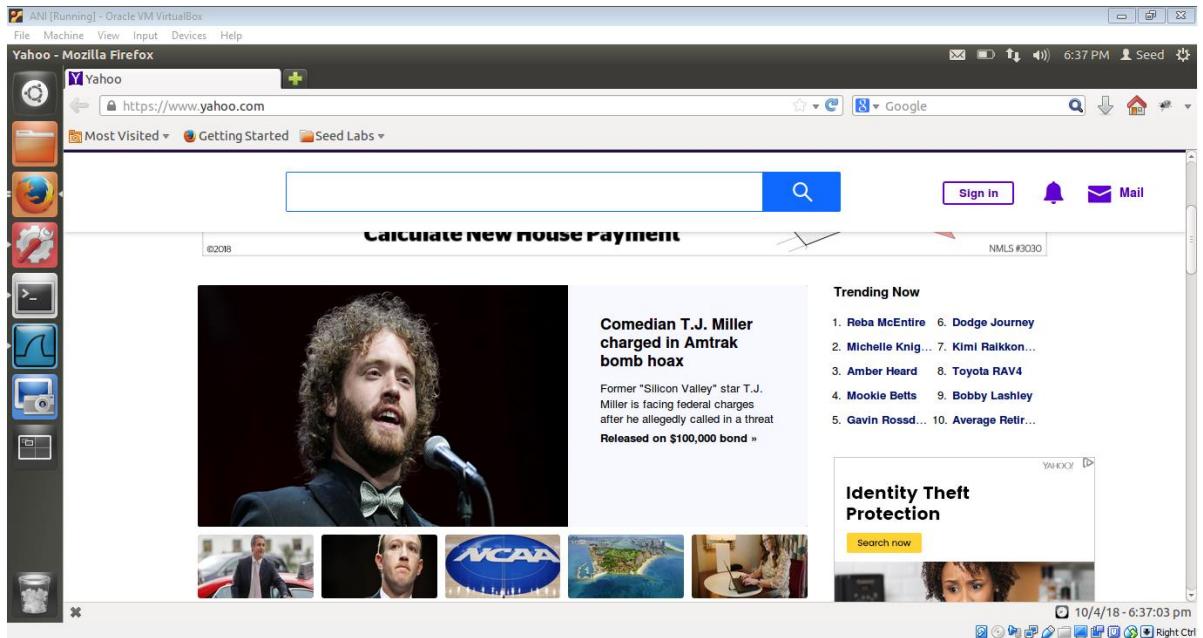
Task 3c:-

- 1) The command in the below screenshot performs redirecting the user on Machine “ANI” from google to www.yahoo.com. Each time we give the url as google.com we will be redirected to www.yahoo.com.

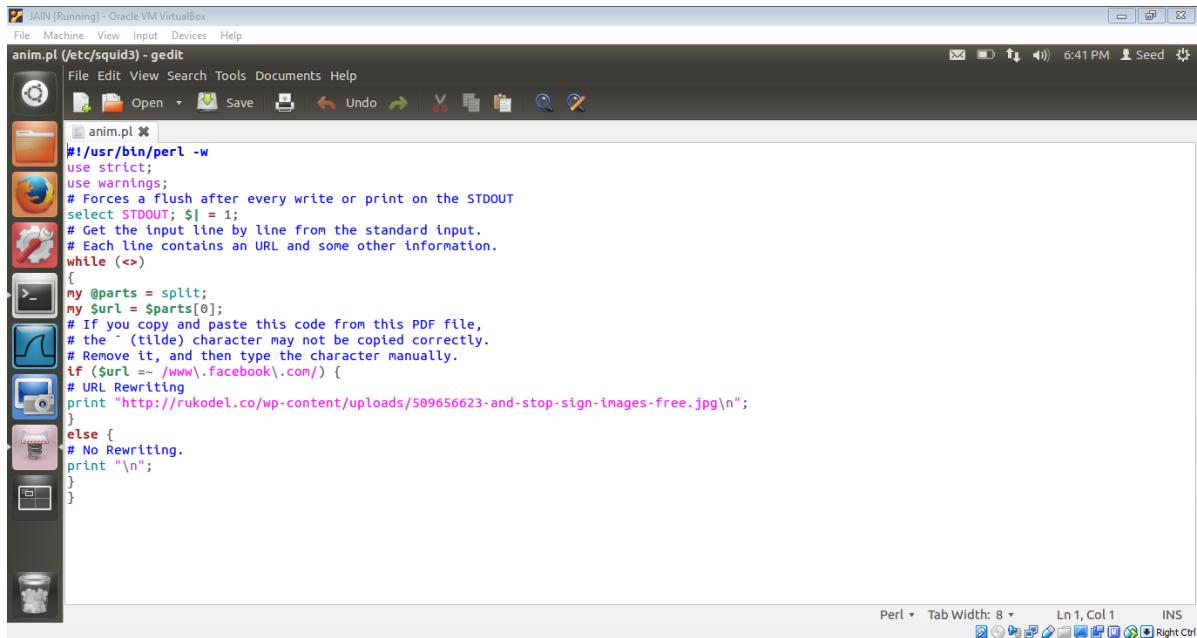
```

#!/usr/bin/perl -w
use strict;
use warnings;
# Forces a flush after every write or print on the STDOUT
select STDOUT; $| = 1;
# Get the input line by line from the standard input.
# Each line contains an URL and some other information.
while (<>)
{
    my @parts = split;
    my $url = $parts[0];
    # If you copy and paste this code from this PDF file,
    # the `~ (tilde) character may not be copied correctly.
    # Remove it, and then type the character manually.
    if ($url =~ /google/) {
        # URI Rewriting
        print "http://www.yahoo.com\n";
    }
    else {
        # No Rewriting.
        print "\n";
    }
}

```



2)

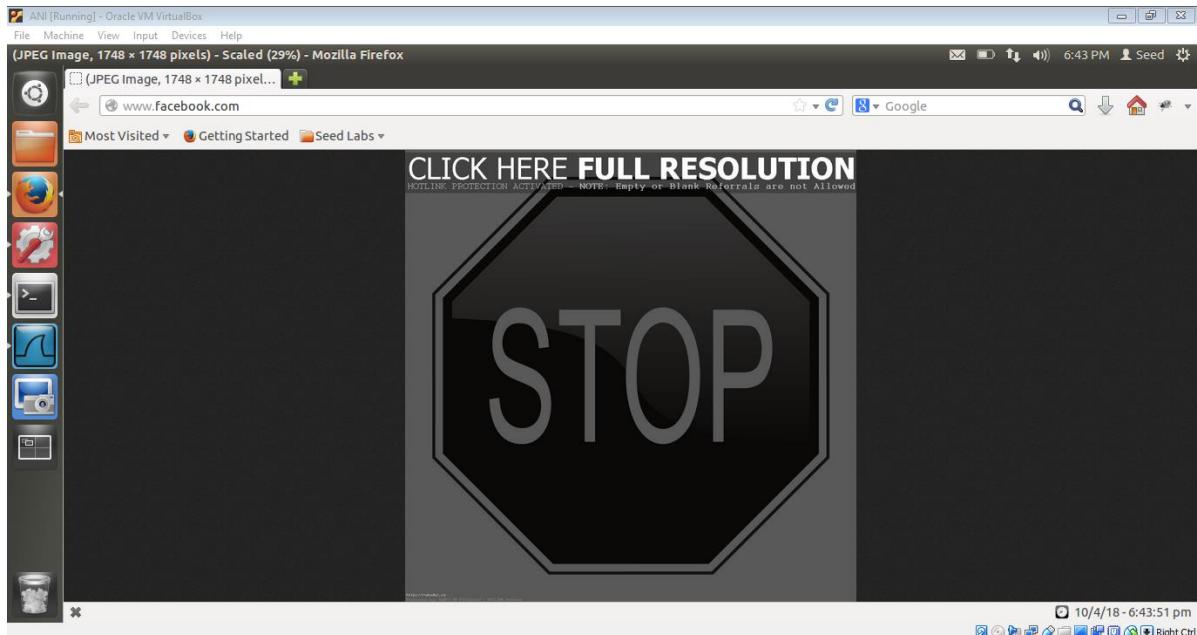


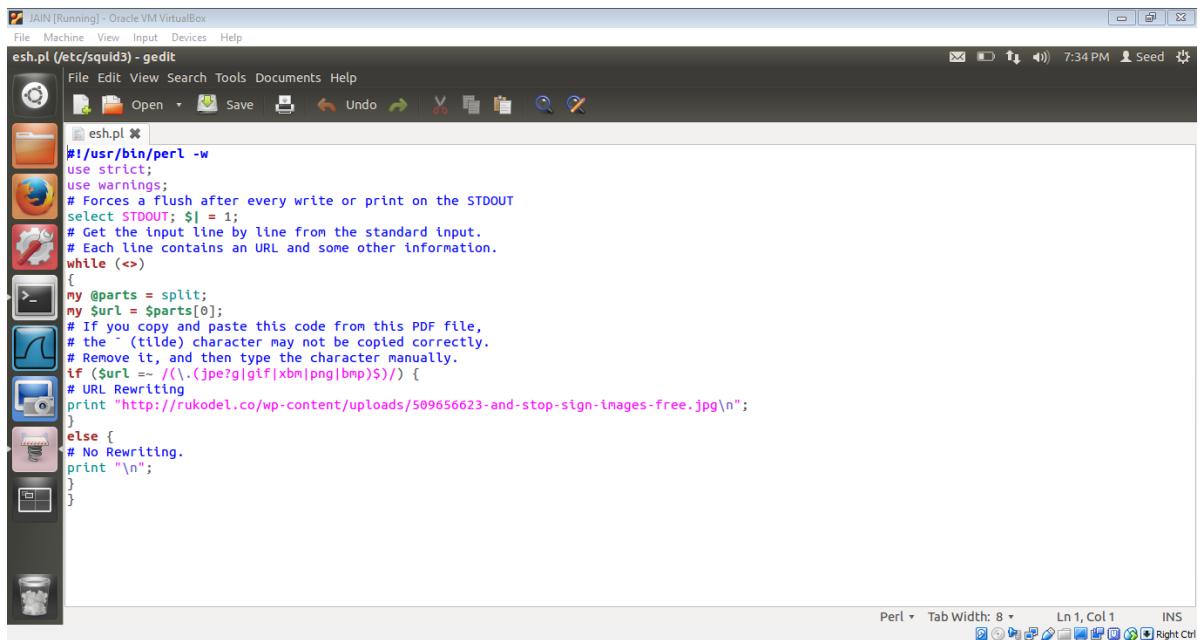
```

anim.pl (/etc/squid3) - gedit
File Edit View Search Tools Documents Help
Open Save Undo Redo Cut Copy Paste Find Replace Select All
anim.pl $|
#!/usr/bin/perl -w
use strict;
use warnings;
# Forces a flush after every write or print on the STDOUT
select STDOUT; $|=1;
# Get the input line by line from the standard input.
# Each line contains an URL and some other information.
while (<>){
{
my @parts = split;
my $url = $parts[0];
# If you copy and paste this code from this PDF file,
# the ~ (tilde) character may not be copied correctly.
# Remove it, and then type the character manually.
if ($url =~ /www\.\.facebook\.com/) {
# URL Rewriting
print "http://rukodel.co/wp-content/uploads/509656623-and-stop-sign-images-free.jpg\n";
}
else {
# No Rewriting.
print "\n";
}
}
Perl Tab Width: 8 Ln 1, Col 1 INS Right Ctrl

```

In this program we changed the program in such a way that every time user on machine “ANI” tries to access www.facebook.com, the user will be redirected to <http://rukodel.co/wp-content/uploads/509656623-and-stop-sign-images-free.jpg> i.e a picture of a stop sign.





```
#!/usr/bin/perl -w
use strict;
use warnings;
# Forces a flush after every write or print on the STDOUT
select STDOUT; $|=1;
# Get the input line by line from the standard input.
# Each line contains an URL and some other information.
while (<>)
{
    my @parts = split;
    my $url = $parts[0];
    # If you copy and paste this code from this PDF file,
    # the `~` character may not be copied correctly.
    # Remove it, and then type the character manually.
    if ($url =~ /(\.(jpe?g|gif|xbm|png|bmp)$)/) {
        # URI Rewriting
        print "http://rukodel.co/wp-content/uploads/509656623-and-stop-sign-images-free.jpg\n";
    }
    else {
        # No Rewriting.
        print "\n";
    }
}
```

When we give the above program, on machine “ANI” when I access any website (in my case it is www.uic.edu), all the pictures on that website will be replaced with the image whose url I provided.

