

UE18CS335 – COMPUTER NETWORK SECURITY

Lab – 3 LINUX FIREWALL EXPLORATION LAB

Date: 21/02/2021

By:

Nitish S

PES2201800368

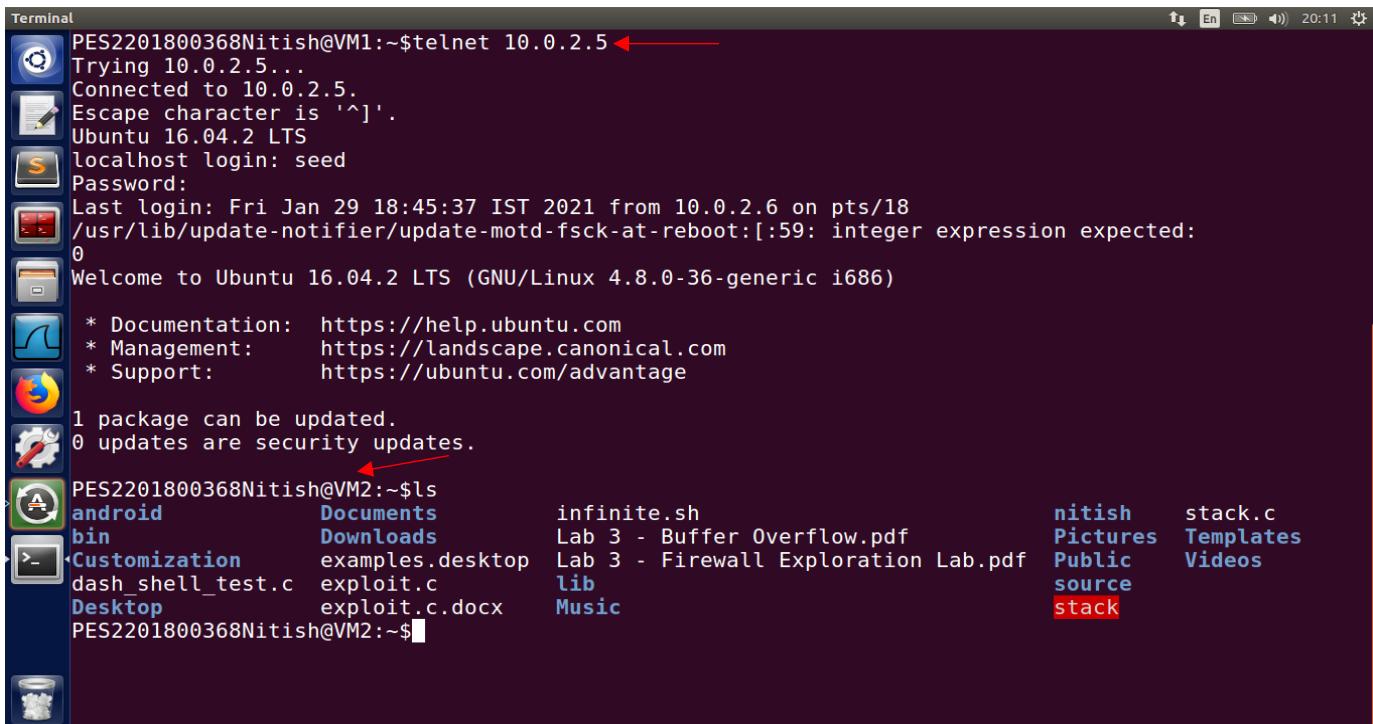
6 'A'

TASK 1: USING FIREWALL

Setup: - VM1 = 10.0.2.4 ; VM2 = 10.0.2.5

PART-1:

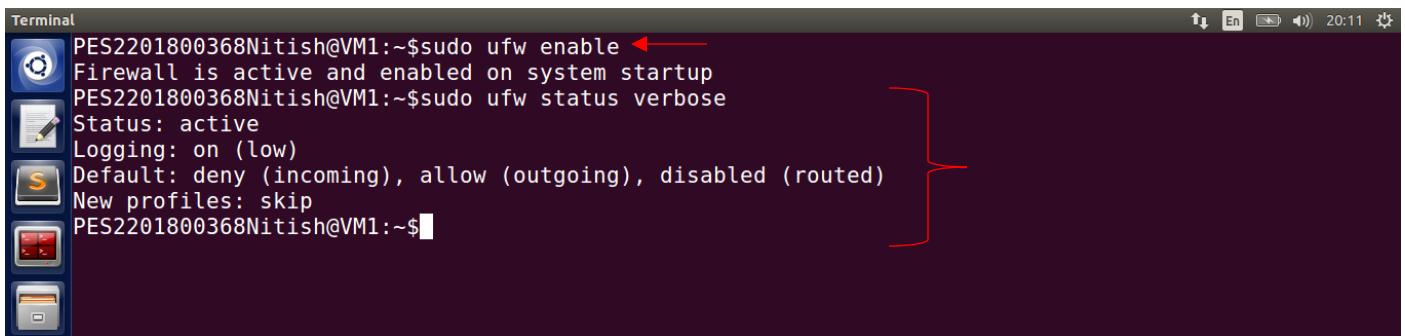
Telnet to VM2 from VM1 to check if it is a success or not.



The screenshot shows a terminal window with a dark background and light-colored text. It displays the output of a Telnet session from VM1 to VM2. The session starts with the command `$ telnet 10.0.2.5`. The terminal then shows the connection to VM2, which is running Ubuntu 16.04.2 LTS. The user logs in as 'seed' and enters a password. The terminal then displays a welcome message for Ubuntu 16.04.2 LTS and provides documentation links. Below this, it shows package update information: 1 package can be updated and 0 updates are security updates. Finally, the user runs the command `ls` to list the contents of their home directory on VM2, showing various files and folders like 'Android', 'bin', 'Customization', 'dash_shell_test.c', 'Desktop', 'Documents', 'Downloads', 'examples.desktop', 'exploit.c', 'exploit.c.docx', 'infinite.sh', 'Lab 3 - Buffer Overflow.pdf', 'Lab 3 - Firewall Exploration Lab.pdf', 'lib', 'Music', 'nitish', 'Pictures', 'Public', 'source', 'stack', 'Templates', 'Videos', and 'stack.c'. A red arrow points to the first line of the Telnet session output, highlighting the connection attempt.

SCREENSHOT SHOWING SUCCESSFUL TELNET CONNECTION FROM VM1 TO VM2

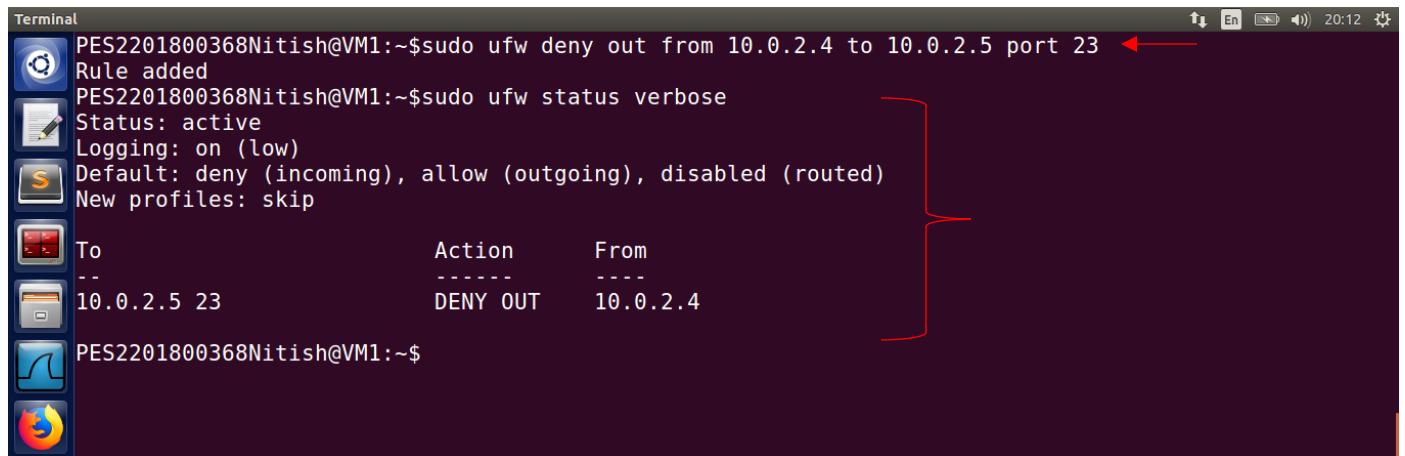
Enabling the firewall and checking the default set rules



```
PES2201800368Nitish@VM1:~$sudo ufw enable
Firewall is active and enabled on system startup
PES2201800368Nitish@VM1:~$sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
PES2201800368Nitish@VM1:~$
```

SCREENTSHOT SHOWING THE DEFAULT FIREWALL RULES

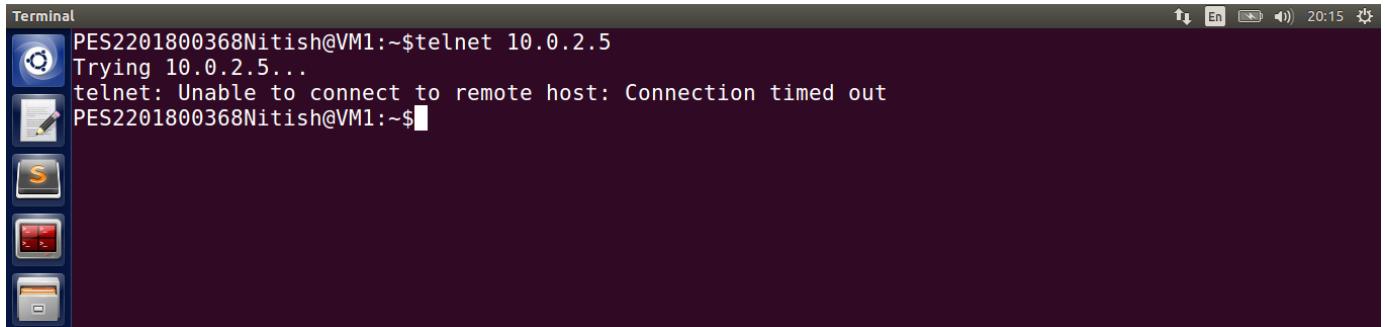
Manually configuring the firewall to deny connection to VM2 using port 23



```
PES2201800368Nitish@VM1:~$sudo ufw deny out from 10.0.2.4 to 10.0.2.5 port 23
Rule added
PES2201800368Nitish@VM1:~$sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
To           Action     From
--           -----     ---
10.0.2.5 23  DENY OUT  10.0.2.4
PES2201800368Nitish@VM1:~$
```

SCREENTSHOT SHOWING THE RULE ADDED TO FIREWALL

Verification of the added rule



```
PES2201800368Nitish@VM1:~$telnet 10.0.2.5
Trying 10.0.2.5...
telnet: Unable to connect to remote host: Connection timed out
PES2201800368Nitish@VM1:~$
```

SCREENTSHOT SHOWING THE FAILURE OF TELNET CONNECTION FROM VM1
TO VM2 DUE TO THE NEWLY ADDED FIREWALL RULE

PART – 2:

Telnet to VM1 from VM2 to check if it's a success or not

The screenshot shows a terminal window titled "Terminal" with the following output:

```
PES2201800368Nitish@VM2:~$telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
localhost login: seed
Password:
Last login: Fri Feb 19 20:15:55 IST 2021 from 10.0.2.4 on pts/18
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

PES2201800368Nitish@VM1:~$ls
android      Desktop    examples.desktop
bin          Documents  Lab 3 - Firewall Exploration Lab.pdf
Customization Downloads lib
Music        Public     Videos
nitish       source
Pictures    Templates
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING SUCCESSFUL TELNET CONNECTION FROM VM2 TO VM1

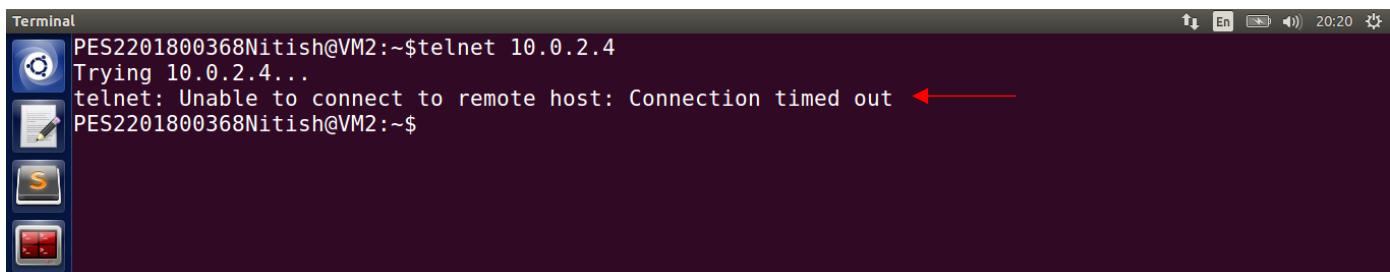
Deleting the existing rules and configuring a new rule to prevent telnet connection from VM2 to VM1

The screenshot shows a terminal window titled "Terminal" with the following output:

```
PES2201800368Nitish@VM1:~$sudo ufw enable
Firewall is active and enabled on system startup
PES2201800368Nitish@VM1:~$sudo ufw delete 1
Deleting:
deny out from 10.0.2.4 to 10.0.2.5 port 23
Proceed with operation (y|n)? y
Rule deleted
PES2201800368Nitish@VM1:~$sudo ufw deny in from 10.0.2.5 to 10.0.2.4 port 23
Rule added
PES2201800368Nitish@VM1:~$sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
To           Action      From
--           ----      ---
10.0.2.4 23  DENY IN   10.0.2.5
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE NEWLY ADDED FIREWALL RULE

Testing the added firewall rule

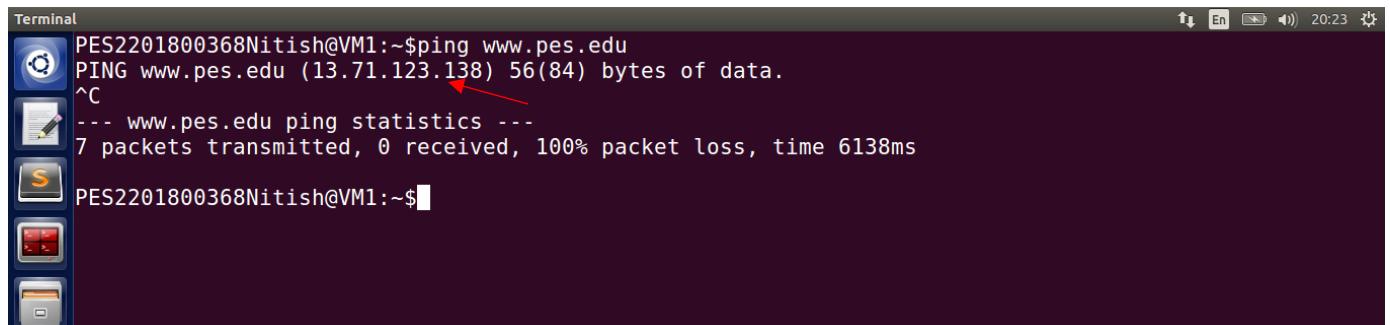


```
Terminal
PES2201800368Nitish@VM2:~$telnet 10.0.2.4
Trying 10.0.2.4...
telnet: Unable to connect to remote host: Connection timed out ←
PES2201800368Nitish@VM2:~$
```

SCREENSHOT SHOWING THE FAILURE OF TELNET FROM VM2 TO VM1 DUE TO THE NEWLY CONFIGURED FIREWALL RULE

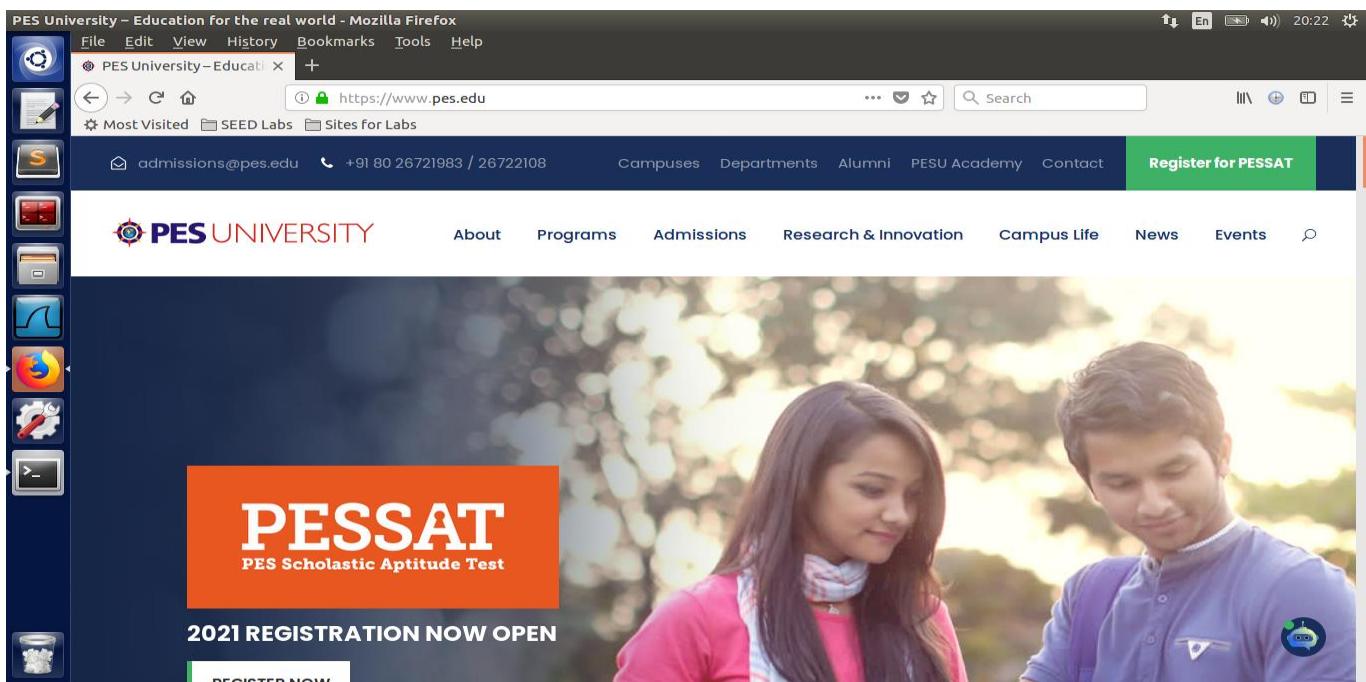
PART – 3:

To obtain the IP address of www.pes.edu using ping command



```
Terminal
PES2201800368Nitish@VM1:~$ping www.pes.edu
PING www.pes.edu (13.71.123.138) 56(84) bytes of data.
^C
--- www.pes.edu ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6138ms
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE IP ADDRESS OF www.pes.edu



SCREENSHOT SHOWING www.pes.edu ACCESSIBLE FROM THE BROWSER

Adding a new rule to the firewall to deny access to www.pes.edu website

The screenshot shows a terminal window with the following command history:

```
PES2201800368Nitish@VM1:~$sudo ufw delete 1  
Deleting:  
deny from 10.0.2.5 to 10.0.2.4 port 23  
Proceed with operation (y|n)? y  
Rule deleted  
PES2201800368Nitish@VM1:~$sudo ufw deny out to 13.71.123.138  
Rule added  
PES2201800368Nitish@VM1:~$sudo ufw status verbose  
Status: active  
Logging: on (low)  
Default: deny (incoming), allow (outgoing), disabled (routed)  
New profiles: skip  
To Action From  
-- ----- -----  
13.71.123.138 DENY OUT Anywhere
```

A red curly brace is positioned to the right of the output, grouping the "Status: active" line and the "To Action From" table.

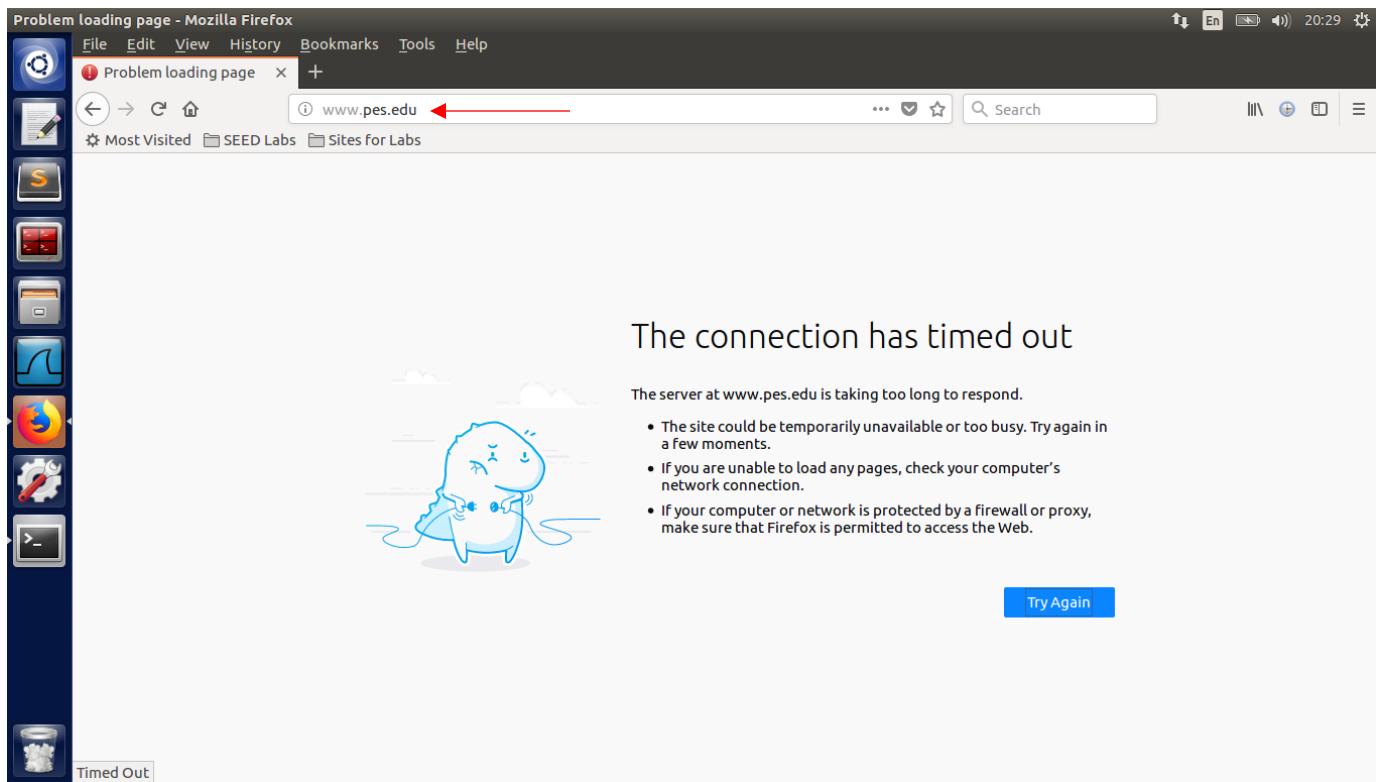
SCREENSHOT SHOWING THE NEWLY CONFIGURED RULE

Testing the newly added rule

The screenshot shows a terminal window with the following command history:

```
PES2201800368Nitish@VM1:~$ping www.pes.edu  
PING www.pes.edu (13.71.123.138) 56(84) bytes of data.  
ping: sendmsg: Operation not permitted  
ping: sendmsg: Operation not permitted  
ping: sendmsg: Operation not permitted  
ping: sendmsg: Operation not permitted
```

SCREENSHOT SHOWING THE FAILURE OF PING COMMAND TO THE BLOCKED WEBSITE



SCREENSHOT SHOWING INACCESSIBILITY OF THE WEBSITE FROM THE BROWSER

Observations from Task 1:

Firewall is a part of a computer system designed to stop unauthorized traffic flowing from one network to another and is used for the purpose of filtering data and protecting against attacks. In the above task, we perform filtering of incoming or outgoing traffic by configuring the allow/deny rules in the firewall. Denying all incoming and outgoing telnet connections to the system i.e. to port 23 was the first part and the same was tested and verified when a connection timed out message was obtained when tried to use telnet. In the second part, we deny an outgoing connection to a website by using its IP address of the website and its verified using the operation failed message when the ping command was run and connection timed out message when the website was accessed in the browser.

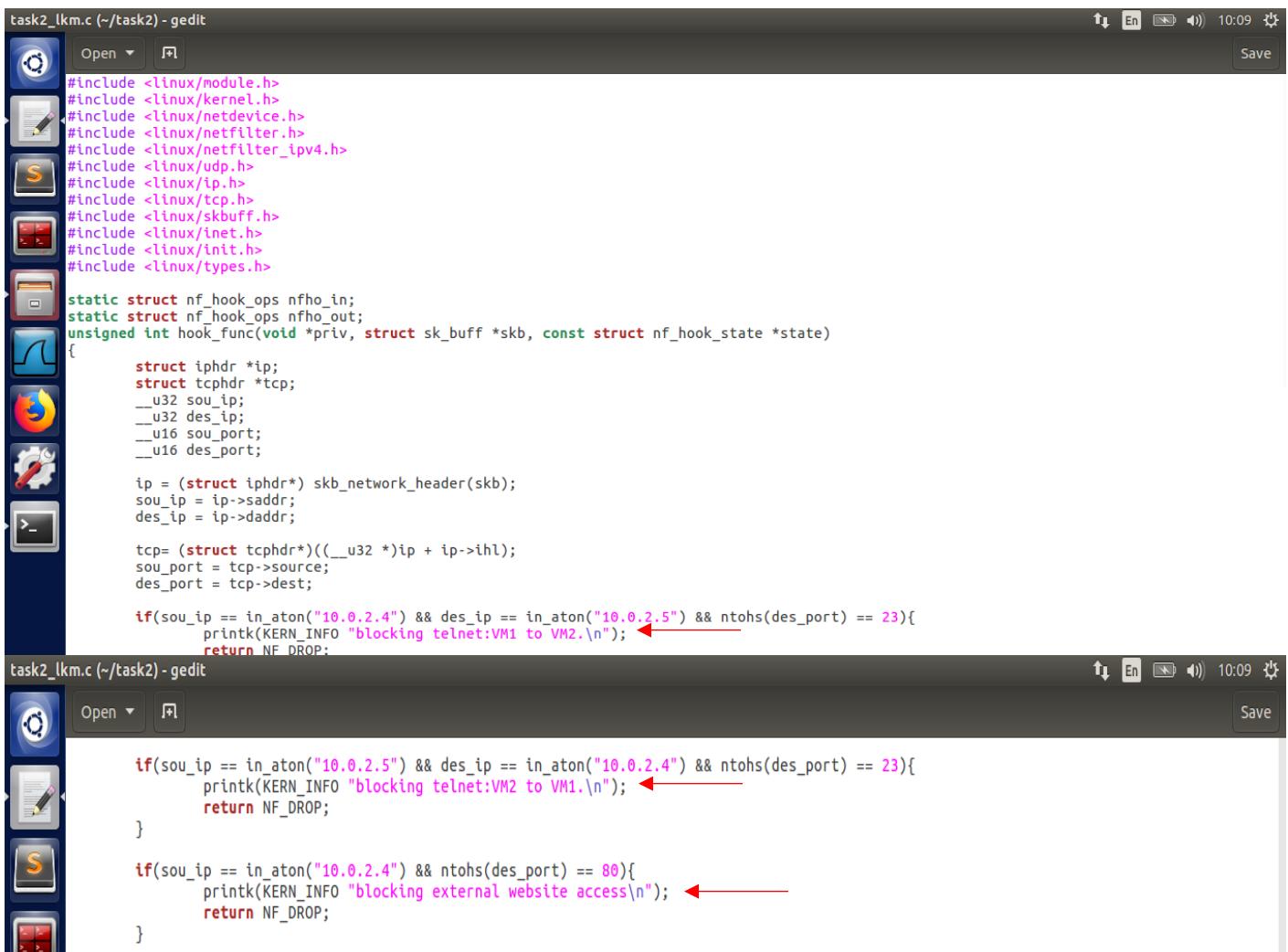
TASK 2: HOW FIREWALL WORKS

In this task, we will develop a firewall using netfilter and LKM. We implement five rules in this firewall:

- Block telnet from VM1 to VM2
- Block telnet from VM2 to VM1
- Block external website access from VM1
- Block ssh from VM1 to VM2
- Block ssh from VM2 to VM1

Setup: VM1 = 10.0.2.4 ; VM2 = 10.0.2.5

task2_lkm.c



```
task2_lkm.c (~/task2) - gedit
Open ▾ Save
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/netdevice.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/udp.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/skbuff.h>
#include <linux/inet.h>
#include <linux/init.h>
#include <linux/types.h>

static struct nf_hook_ops nfho_in;
static struct nf_hook_ops nfho_out;
unsigned int hook_func(void *priv, struct sk_buff *skb, const struct nf_hook_state *state)
{
    struct iphdr *ip;
    struct tcphdr *tcp;
    __u32 sou_ip;
    __u32 des_ip;
    __u16 sou_port;
    __u16 des_port;

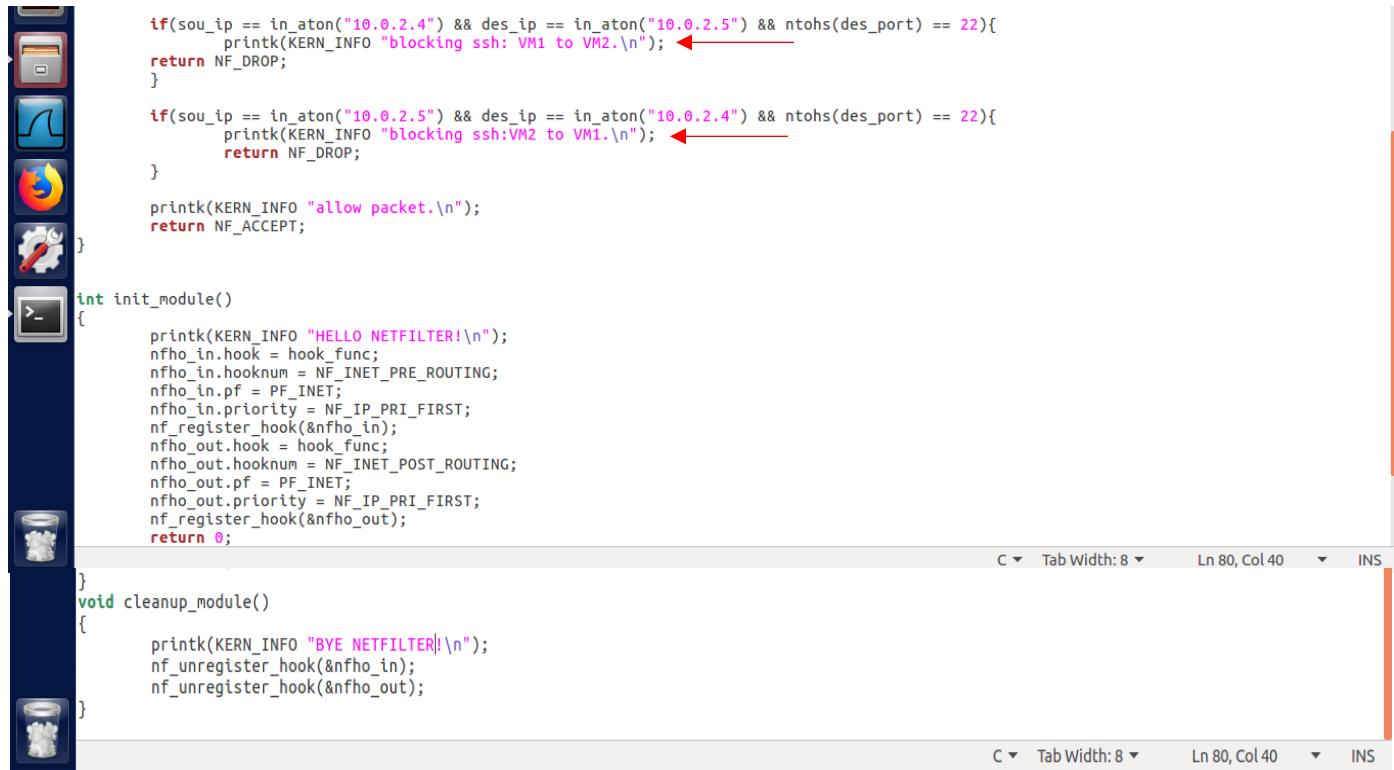
    ip = (struct iphdr*) skb_network_header(skb);
    sou_ip = ip->saddr;
    des_ip = ip->daddr;

    tcp= (struct tcphdr*)((__u32 *)ip + ip->ihl);
    sou_port = tcp->source;
    des_port = tcp->dest;

    if(sou_ip == in_aton("10.0.2.4") && des_ip == in_aton("10.0.2.5") && ntohs(des_port) == 23){
        printk(KERN_INFO "blocking telnet:VM1 to VM2.\n");
        return NF_DROP;
    }
}

task2_lkm.c (~/task2) - gedit
Open ▾ Save
if(sou_ip == in_aton("10.0.2.5") && des_ip == in_aton("10.0.2.4") && ntohs(des_port) == 23){
    printk(KERN_INFO "blocking telnet:VM2 to VM1.\n");
    return NF_DROP;
}

if(sou_ip == in_aton("10.0.2.4") && ntohs(des_port) == 80){
    printk(KERN_INFO "blocking external website access\n");
    return NF_DROP;
}
```



```

if(sou_ip == in_aton("10.0.2.4") && des_ip == in_aton("10.0.2.5") && ntohs(des_port) == 22){
    printk(KERN_INFO "blocking ssh: VM1 to VM2.\n");
    return NF_DROP;
}

if(sou_ip == in_aton("10.0.2.5") && des_ip == in_aton("10.0.2.4") && ntohs(des_port) == 22){
    printk(KERN_INFO "blocking ssh: VM2 to VM1.\n");
    return NF_DROP;
}

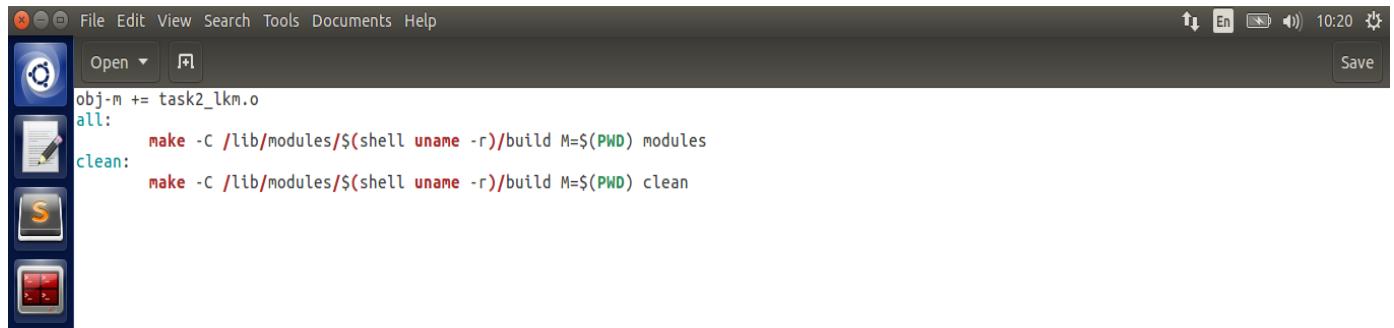
printk(KERN_INFO "allow packet.\n");
return NF_ACCEPT;
}

int init_module()
{
    printk(KERN_INFO "HELLO NETFILTER!\n");
    nfho_in.hook = hook_func;
    nfho_in.hooknum = NF_INET_PRE_ROUTING;
    nfho_in(pf = PF_INET;
    nfho_in.priority = NF_IP_PRI_FIRST;
    nf_register_hook(&nfho_in);
    nfho_out.hook = hook_func;
    nfho_out.hooknum = NF_INET_POST_ROUTING;
    nfho_out(pf = PF_INET;
    nfho_out.priority = NF_IP_PRI_FIRST;
    nf_register_hook(&nfho_out);
    return 0;
}

void cleanup_module()
{
    printk(KERN_INFO "BYE NETFILTER!\n");
    nf_unregister_hook(&nfho_in);
    nf_unregister_hook(&nfho_out);
}

```

makefile

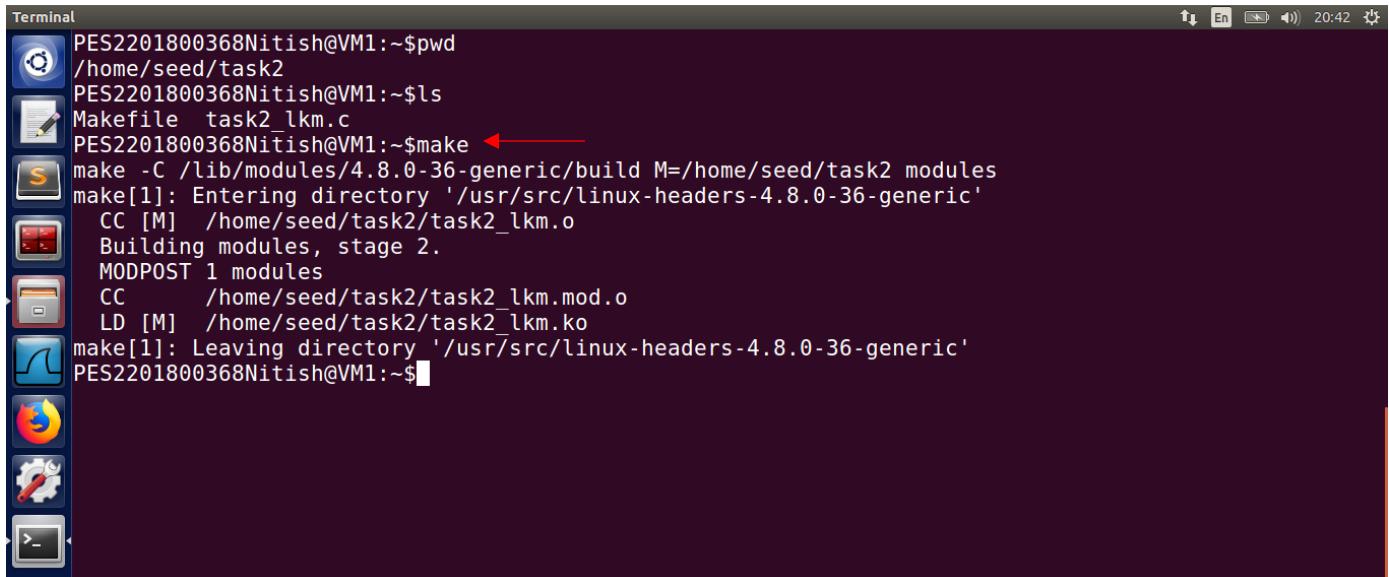


```

obj-m += task2_lkm.o
all:
    make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
clean:
    make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean

```

The program can be compiled using the ***make*** command. (Note: put lkm.c & Makefile into a folder. Open a terminal by right clicking inside that folder. Type ‘*make*’).

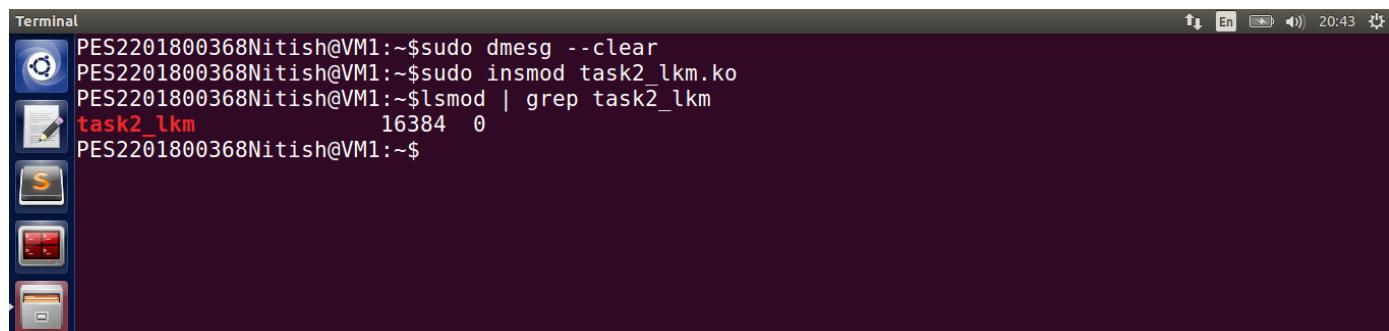
A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window contains the following command-line session:

```
PES2201800368Nitish@VM1:~$pwd  
/home/seed/task2  
PES2201800368Nitish@VM1:~$ls  
Makefile task2_lkm.c  
PES2201800368Nitish@VM1:~$make ←  
make -C /lib/modules/4.8.0-36-generic/build M=/home/seed/task2 modules  
make[1]: Entering directory '/usr/src/linux-headers-4.8.0-36-generic'  
CC [M] /home/seed/task2/task2_lkm.o  
Building modules, stage 2.  
MODPOST 1 modules  
CC /home/seed/task2/task2_lkm.mod.o  
LD [M] /home/seed/task2/task2_lkm.ko  
make[1]: Leaving directory '/usr/src/linux-headers-4.8.0-36-generic'  
PES2201800368Nitish@VM1:$
```

The terminal window has a dark background and light-colored text. The desktop icons on the left include icons for Dash, Home, Applications, and the Dash search bar.

SCREENSHOT SHOWING THE make COMMAND EXECUTION

The compiled kernel module (task2_lkm.ko) can be inserted using insmod:

A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window contains the following command-line session:

```
PES2201800368Nitish@VM1:~$sudo dmesg --clear  
PES2201800368Nitish@VM1:~$sudo insmod task2_lkm.ko  
PES2201800368Nitish@VM1:~$lsmod | grep task2_lkm  
task2_lkm 16384 0  
PES2201800368Nitish@VM1:~$
```

The terminal window has a dark background and light-colored text. The desktop icons on the left are partially visible.

Now we test all the added firewall rules:

- Block telnet from VM1 to VM2
- Block telnet from VM2 to VM1
- Block external website access from VM1
- Block ssh from VM1 to VM2
- Block ssh from VM2 to VM1

Terminal

```
PES2201800368Nitish@VM1:~$telnet 10.0.2.5 ←
Trying 10.0.2.5...
telnet: Unable to connect to remote host: Connection timed out
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING TELNET FROM VM1 to VM2 BLOCKED

Terminal

```
PES2201800368Nitish@VM2:~$telnet 10.0.2.4 ←
Trying 10.0.2.4...
telnet: Unable to connect to remote host: Connection timed out
PES2201800368Nitish@VM2:~$
```

PES2201800368Nitish@VM1:~\$dmesg | tail -10 ←
[1495.217116] allow packet.
[1495.217177] [UFW BLOCK] IN=enp0s3 OUT= MAC=08:00:27:50:43:53:08:00:27:b8:e3:c8:08:00 SRC=10.0.2.7 DST=10.0.2.4 LEN=60 TOS=0x10 PREC=0x00 TTL=64 ID=38607 DF PROTO=TCP SPT=45122 DPT=23 WINDOW=29200 RES=0x00 SYN URGP=0
[1503.405813] allow packet.
[1503.405911] [UFW BLOCK] IN=enp0s3 OUT= MAC=08:00:27:50:43:53:08:00:27:b8:e3:c8:08:00 SRC=10.0.2.7 DST=10.0.2.4 LEN=60 TOS=0x10 PREC=0x00 TTL=64 ID=38608 DF PROTO=TCP SPT=45122 DPT=23 WINDOW=29200 RES=0x00 SYN URGP=0
[1517.164846] blocking telnet:VM2 to VM1.
[1519.525310] allow packet.
[1519.525367] [UFW BLOCK] IN=enp0s3 OUT= MAC=08:00:27:50:43:53:08:00:27:b8:e3:c8:08:00 SRC=10.0.2.7 DST=10.0.2.4 LEN=60 TOS=0x10 PREC=0x00 TTL=64 ID=38609 DF PROTO=TCP SPT=45122 DPT=23 WINDOW=29200 RES=0x00 SYN URGP=0
[1522.382533] blocking telnet:VM2 to VM1.
[1523.408403] blocking telnet:VM2 to VM1.
[1525.424893] blocking telnet:VM2 to VM1.
PES2201800368Nitish@VM1:~\$

SCREENSHOT SHOWING TELNET FROM VM2 to VM1 BLOCKED AND THE COMMAND RUN TO CHECK THE DROPPING OF PACKETS

Firefox Web Browser

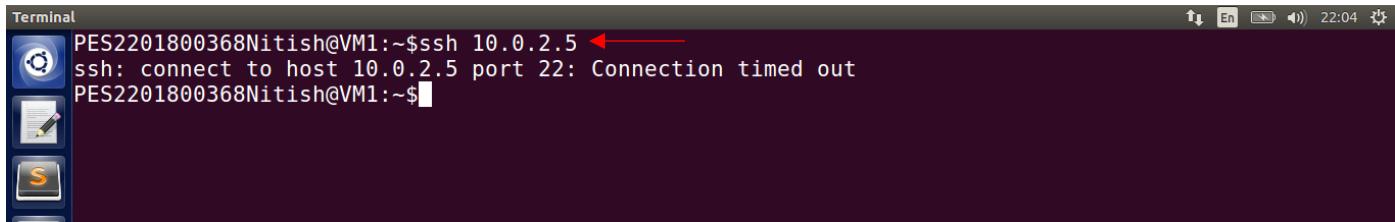
```
PES2201800368Nitish@VM1:~$dmesg | tail -10 ←
[ 3034.303343] allow packet.
[ 3034.303403] allow packet.
[ 3034.362626] allow packet.
[ 3034.362722] allow packet.
[ 3034.362728] allow packet.
[ 3034.362916] blocking external website access
[ 3034.463788] blocking external website access
[ 3034.688177] blocking external website access
[ 3035.071765] blocking external website access
[ 3035.391737] blocking external website access
PES2201800368Nitish@VM1:~$
```

The connection has timed out
The server at www.pes.edu is taking too long to respond.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the Web.

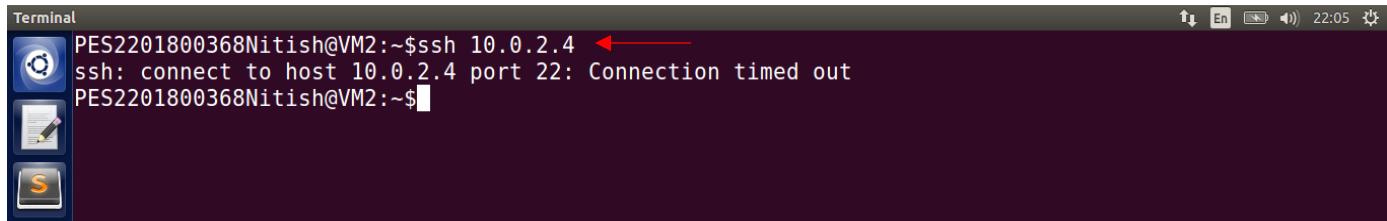
Try Again

SCREENSHOT SHOWING ACCESS TO www.pes.edu BLOCKED



```
Terminal
PES2201800368Nitish@VM1:~$ssh 10.0.2.5 ←
ssh: connect to host 10.0.2.5 port 22: Connection timed out
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE SSH FROM VM1 to VM2 BLOCKED



```
Terminal
PES2201800368Nitish@VM2:~$ssh 10.0.2.4 ←
ssh: connect to host 10.0.2.4 port 22: Connection timed out
PES2201800368Nitish@VM2:~$
```

SCREENSHOT SHOWING THE SSH FROM VM2 to VM1 BLOCKED

Observation from Task2:

The main aim of this task was to implement additional features in the form of kernel modules in Linux and load them dynamically. Kernel Modules are pieces of code that can be loaded and unloaded on-demand at runtime. We write such a kernel module named `task2_lkm.c`, compile it and install the module using makefile and build the same using the commands used as shown in the above screenshots. The kernel module contains the various firewall rules and when the module is loaded and run, we see that the rules get implemented and the firewall is active thereby filtering all the packets and denying connections as mentioned in the rules which is evident from the screenshots of the executions.

TASK 3: EVADING EGRESS FILTERING

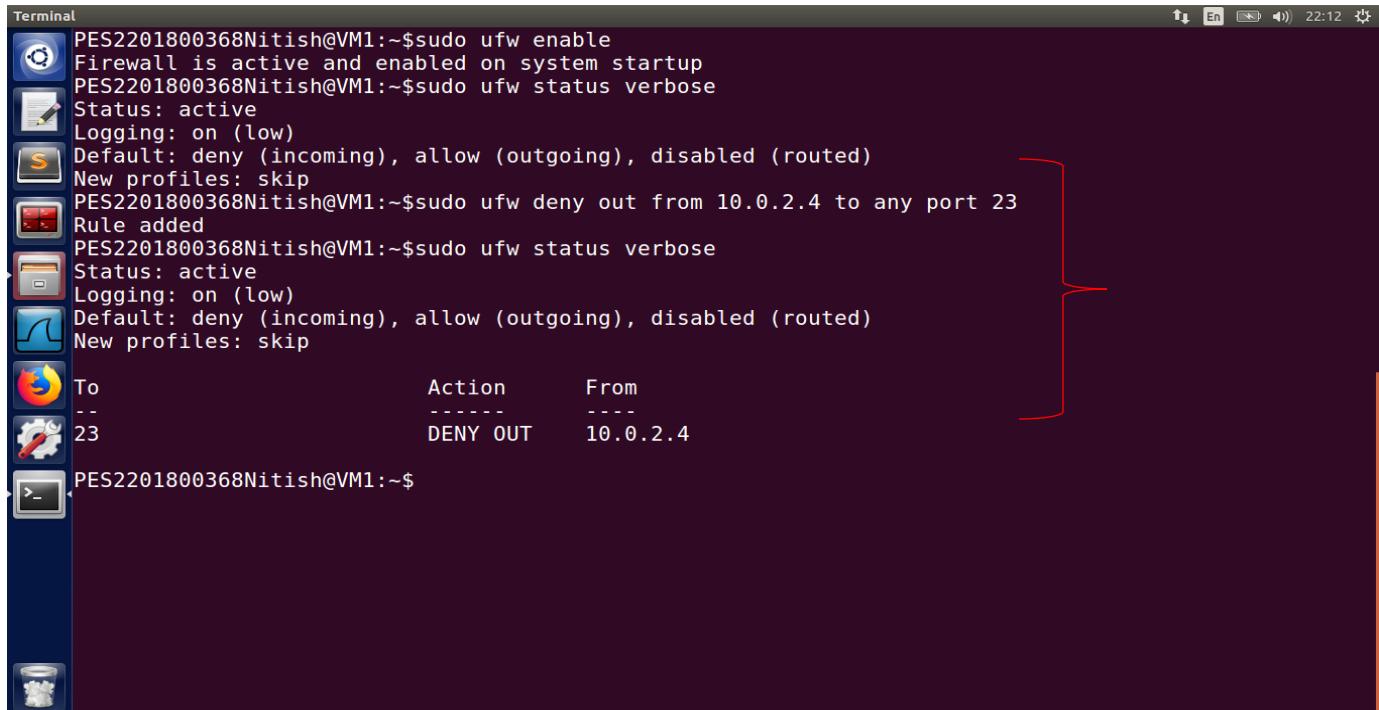
In this task, we will be using a `ssh` tunnel to evade egress filtering.

Setup: VM1 = 10.0.2.4 ; VM2 = 10.0.2.5 ; VM3 = 10.0.2.7

TASK 3.A: TELNET TO MACHINE B THROUGH THE FIREWALL

Aim: In this lab, we will use three VMs. VM1 will be blocked from being able to telnet to VM2. We will utilize a `ssh` tunnel to allow VM1 to telnet to VM2 via VM3. VM1 is analogous to ‘home machine’, VM2 to ‘work machine’ and VM3 to ‘apollo machine’.

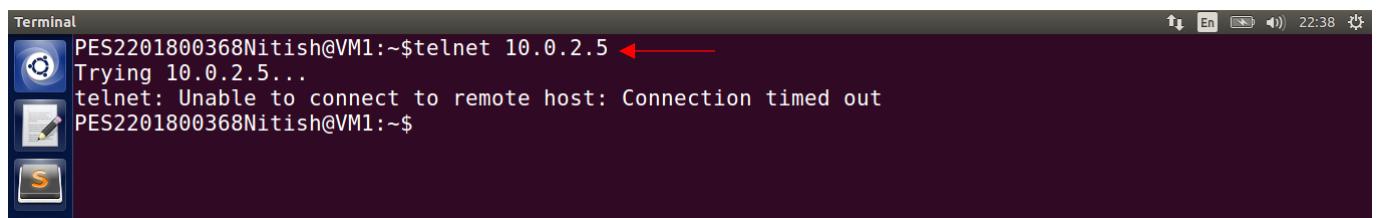
We first delete all the existing firewall rules on VM1 and add a new rule to prevent VM1 from being able to telnet to any other machine.



```
PES2201800368Nitish@VM1:~$sudo ufw enable
Firewall is active and enabled on system startup
PES2201800368Nitish@VM1:~$sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
PES2201800368Nitish@VM1:~$sudo ufw deny out from 10.0.2.4 to any port 23
Rule added
PES2201800368Nitish@VM1:~$sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
To           Action      From
--           -----      ---
23          DENY OUT   10.0.2.4
```

SCREENSHOT SHOWING THE NEWLY ADDED RULES

To verify the effect of the added rule:



```
PES2201800368Nitish@VM1:~$telnet 10.0.2.5 ←
Trying 10.0.2.5...
telnet: Unable to connect to remote host: Connection timed out
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE FAILURE OF TELNET FROM VM1 to VM2

We next setup a *ssh* tunnel between VM1 and VM2 to allow VM1 to telnet via VM2, evading the firewall on VM1. The *ssh* command below allows VM1 to use its local port 8000 to *telnet* to VM3 via VM2.

PES2201800368Nitish@VM1:~\$ssh -L 8000:10.0.2.5:23 seed@10.0.2.7

Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

* Documentation: <https://help.ubuntu.com>
 * Management: <https://landscape.canonical.com>
 * Support: <https://ubuntu.com/advantage>

1 package can be updated.
 0 updates are security updates.

Last login: Fri Feb 19 22:26:40 2021 from 10.0.2.4

PES2201800368Nitish@VM3:~\$

PES2201800368Nitish@VM1:~\$telnet localhost 8000

Trying 127.0.0.1...
 Connected to localhost.
 Escape character is '^'.

Ubuntu 16.04.2 LTS
 localhost login: seed

Password:

Last login: Fri Feb 19 22:27:02 IST 2021 from 1

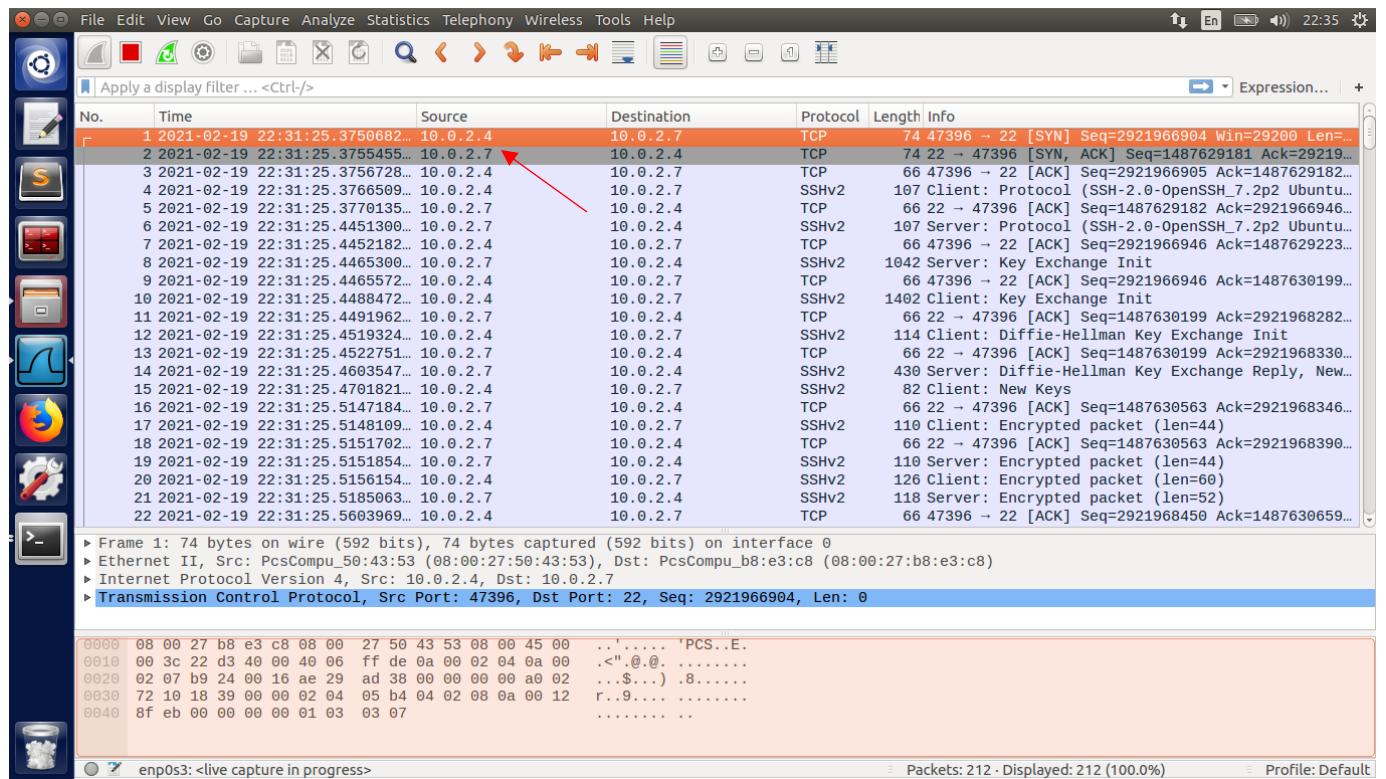
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

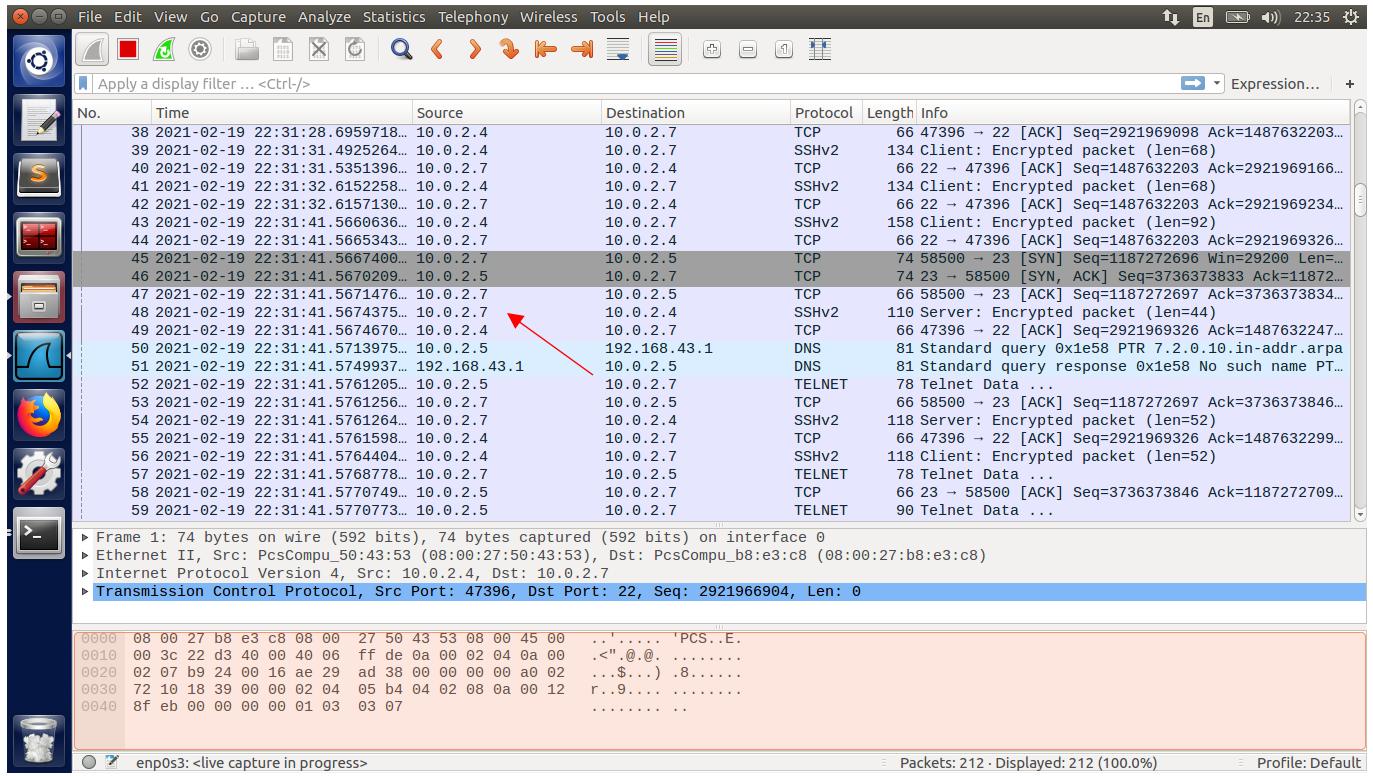
* Documentation: <https://help.ubuntu.com>
 * Management: <https://landscape.canonical.com>
 * Support: <https://ubuntu.com/advantage>

1 package can be updated.
 0 updates are security updates.

PES2201800368Nitish@VM2:~\$

SCREENSHOT SHOWING THE SETTING UP OF ssh TUNNEL and HENCE OBTAINING THE TELNET ACCESS TO VM2 VIA THE ESTABLISHED TUNNEL OF VM3





WIRESHARK CAPTURE

Observations from Task 3a:

The aim of this task to evade firewalls using SSH tunneling. If there exists a firewall to deny telnet connection from ‘home’ machine to ‘work’, then we can build ssh tunnel from ‘home’ to another machine ‘apollo’. This tunnel will forward the TCP data received on port 8000 on ‘home’ to port 23 on ‘work’ thereby evading the firewall from ‘home’ to ‘work’ and establishing successful telnet connection. This is what is done in the above task and is evidently seen in the screenshots and wireshark capture.

TASK 3.B: CONNECTING TO GOOGLE USING SSH TUNNEL

In this task, we will setup a firewall rule to block VM1 from visiting www.google.com but then leverage dynamic forwarding via ssh tunnel to visit www.google.com from VM1 via VM2.

All existing firewall rules are deleted using the ‘`sudo ufw delete <rule_no>`’ command.

Obtaining the IP address of www.google.com using ping command

```
Terminal PES2201800368Nitish@VM1:~$ping www.google.com
PING www.google.com (172.217.163.196) 56(84) bytes of data.
64 bytes from maa05s06-in-f4.1e100.net (172.217.163.196): icmp_seq=1 ttl=111 time=50.1 ms
64 bytes from maa05s06-in-f4.1e100.net (172.217.163.196): icmp_seq=2 ttl=111 time=44.8 ms
64 bytes from maa05s06-in-f4.1e100.net (172.217.163.196): icmp_seq=3 ttl=111 time=34.8 ms
64 bytes from maa05s06-in-f4.1e100.net (172.217.163.196): icmp_seq=4 ttl=111 time=43.6 ms
64 bytes from maa05s06-in-f4.1e100.net (172.217.163.196): icmp_seq=5 ttl=111 time=61.0 ms
64 bytes from maa05s06-in-f4.1e100.net (172.217.163.196): icmp_seq=6 ttl=111 time=42.1 ms
^C
--- www.google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5005ms
rtt min/avg/max/mdev = 34.814/46.098/61.051/8.075 ms
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE IP ADDRESS OF www.google.com

We can setup a firewall rule to block traffic to that IP.

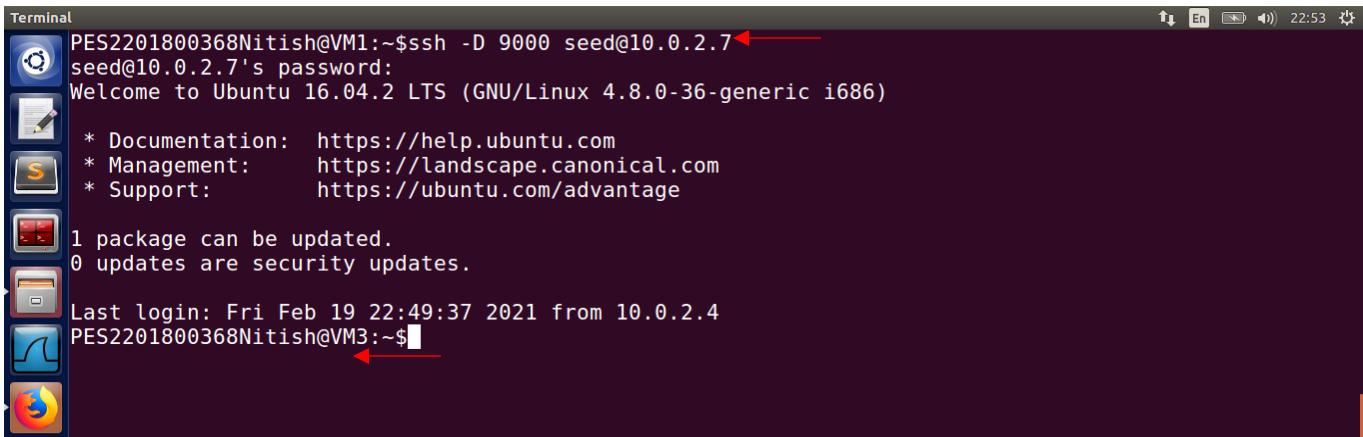
```
PES2201800368Nitish@VM1:~$sudo ufw deny out to 172.217.163.196
Rule added
PES2201800368Nitish@VM1:~$sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
To                      Action      From
--                      ----       ---
172.217.163.196        DENY OUT   Anywhere
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE RULE ADDED

Verifying the added rule

SCREENSHOT SHOWING PING AND BROWSER ACCESS TO www.google.com FAILED

We now setup a ssh tunnel with dynamic port forwarding between VM1 and VM2. With this tunnel setup, VM1 will be able to use its local port 9000 to send a request to www.google.com via VM2.



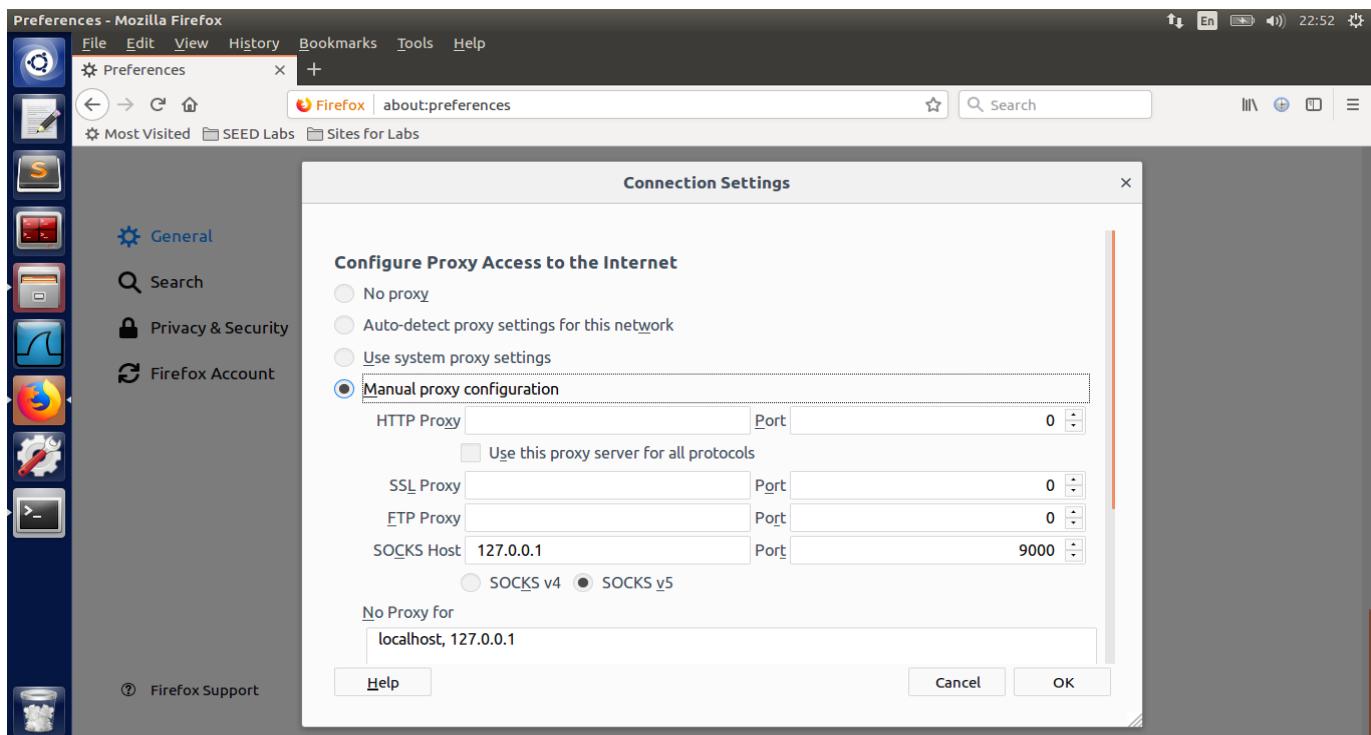
```
Terminal
PES2201800368Nitish@VM1:~$ ssh -D 9000 seed@10.0.2.7
seed@10.0.2.7's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

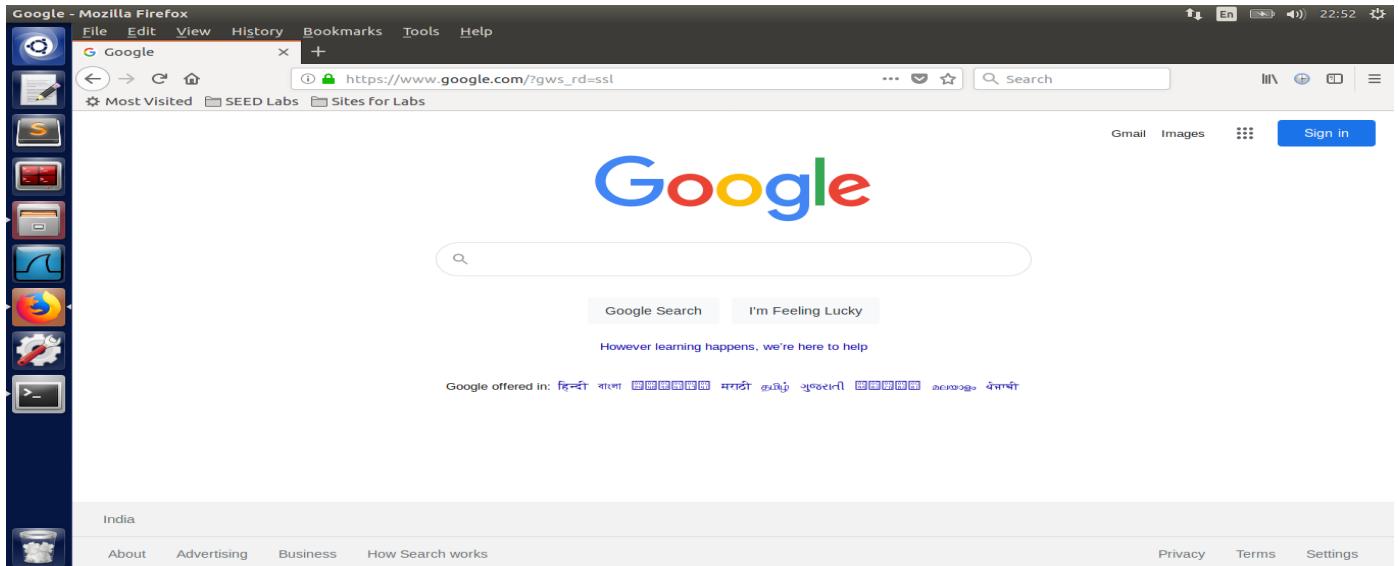
Last login: Fri Feb 19 22:49:37 2021 from 10.0.2.4
PES2201800368Nitish@VM3:~$
```

SCREENSHOT SHOWING THE ESTABLISHING OF TUNNEL TO EVADE THE FIREWALL



SCREENSHOT SHOWING THE PROXY SETTINGS CONFIGURED TO USE THE ESTABLISHED TUNNEL

Testing the established tunnel



SCREENSHOT SHOWING SUCCESSFUL ACCESS TO www.google.com ON CONFIGURING THE PROXY SETTINGS AND ESTABLISHING A TUNNEL TO EVADE THE FIREWALL

Capturing from enp0s3

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-02-19 22:55:14.8002412...	10.0.2.5	10.0.2.3	DHCP	342	DHCP Request - Transaction ID 0x7fa44a6e
2	2021-02-19 22:55:14.8330160...	10.0.2.3	10.0.2.5	DHCP	590	DHCP ACK - Transaction ID 0x7fa44a6e
3	2021-02-19 22:55:16.9510738...	10.0.2.4	10.0.2.7	TCP	74	47850 → 22 [SYN] Seq=1601943567 Win=29200 Len=...
4	2021-02-19 22:55:16.9516713...	10.0.2.7	10.0.2.4	TCP	74 22 → 47850 [SYN, ACK] Seq=1043848920 Ack=16019...	
5	2021-02-19 22:55:16.9517499...	10.0.2.4	10.0.2.7	TCP	66	47850 → 22 [ACK] Seq=1601943568 Ack=1043848921...
6	2021-02-19 22:55:16.9525180...	10.0.2.4	10.0.2.7	SSHv2	107	Client: Protocol (SSH-2.0-OpenSSH_7.2p2 Ubuntu...
7	2021-02-19 22:55:16.9530724...	10.0.2.7	10.0.2.4	TCP	66	22 → 47850 [ACK] Seq=1043848921 Ack=1601943609...
8	2021-02-19 22:55:16.9629308...	10.0.2.7	10.0.2.4	SSHv2	107	Server: Protocol (SSH-2.0-OpenSSH_7.2p2 Ubuntu...
9	2021-02-19 22:55:16.9633082...	10.0.2.4	10.0.2.7	TCP	66	47850 → 22 [ACK] Seq=1601943609 Ack=1043848962...
10	2021-02-19 22:55:16.9645918...	10.0.2.7	10.0.2.4	SSHv2	1042	Server: Key Exchange Init
11	2021-02-19 22:55:16.9645918...	10.0.2.4	10.0.2.7	TCP	66	47850 → 22 [ACK] Seq=1601943609 Ack=1043849938...
12	2021-02-19 22:55:16.96459432...	10.0.2.4	10.0.2.7	SSHv2	1402	Client: Key Exchange Init
13	2021-02-19 22:55:17.0068160...	10.0.2.7	10.0.2.4	TCP	66	22 → 47850 [ACK] Seq=1043849938 Ack=1601944945...
14	2021-02-19 22:55:17.0069051...	10.0.2.4	10.0.2.7	SSHv2	114	Client: Diffie-Hellman Key Exchange Init
15	2021-02-19 22:55:17.0075547...	10.0.2.7	10.0.2.4	TCP	66	22 → 47850 [ACK] Seq=1043849938 Ack=1601944993...
16	2021-02-19 22:55:17.0157235...	10.0.2.7	10.0.2.4	SSHv2	430	Server: Diffie-Hellman Key Exchange Reply, New...
17	2021-02-19 22:55:17.0224724...	10.0.2.4	10.0.2.7	SSHv2	82	Client: New Keys
18	2021-02-19 22:55:17.0674220...	10.0.2.7	10.0.2.4	TCP	66	22 → 47850 [ACK] Seq=1043850302 Ack=1601945009...
19	2021-02-19 22:55:17.0674640...	10.0.2.4	10.0.2.7	SSHv2	110	Client: Encrypted packet (len=44)
20	2021-02-19 22:55:17.0678856...	10.0.2.7	10.0.2.4	TCP	66	22 → 47850 [ACK] Seq=1043850302 Ack=1601945053...
21	2021-02-19 22:55:17.0683366...	10.0.2.7	10.0.2.4	SSHv2	110	Server: Encrypted packet (len=44)
22	2021-02-19 22:55:17.0684230...	10.0.2.4	10.0.2.7	SSHv2	126	Client: Encrypted packet (len=60)

Frame 1: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 0
 ▶ Ethernet II, Src: PcsCompu_74:7b:2e (08:00:27:74:7b:2e), Dst: PcsCompu_ef:b6:51 (08:00:27:ef:b6:51)
 ▶ Internet Protocol Version 4, Src: 10.0.2.5, Dst: 10.0.2.3
 ▶ User Datagram Protocol, Src Port: 68, Dst Port: 67
 ▶ Bootstrap Protocol (Request)

```

0000  08 00 27 ef b6 51 08 00  27 74 7b 2e 08 00 45 00  ..'Q.. 't{...E.
0010  01 48 cf a8 40 00 40 11  51 f5 0a 00 02 05 0a 00  .H..@. Q.....
0020  02 03 00 44 00 43 01 34  92 76 01 06 00 7f a4  ...D.C.4 .V.....
0030  4a 60 00 00 00 00 0a 00  02 05 00 00 00 00 00 00 00  Jn..... .
0040  00 00 00 00 00 00 08 00  27 74 7b 2e 00 00 00 00 00  ..... 't{.....
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  .....
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  .....
  
```

Packets: 42 · Displayed: 42 (100.0%) · Profile: Default

WIRESHARK CAPTURE OF THE SAME

We next disable the tunnel to see its effect. Provide a screenshot of your observations.
 We also clear the browser cache to make sure firefox does not show the webpage it just loaded.

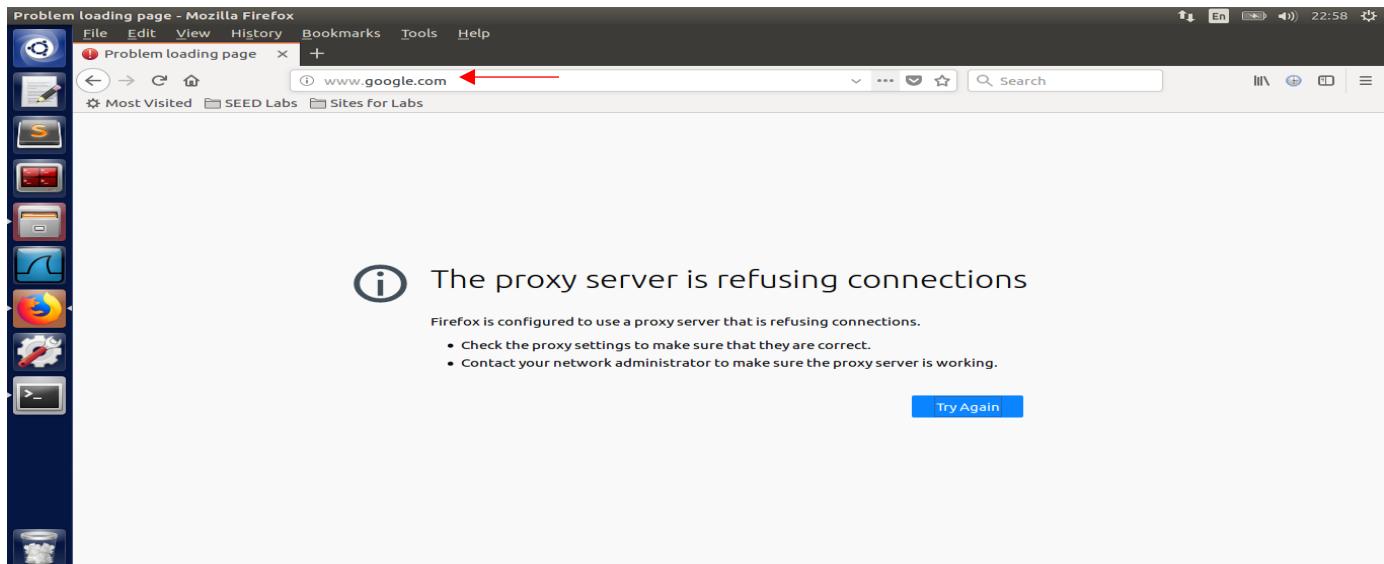
If we try to visit www.google.com again, we observe that the browser informs the proxy is refusing connections. The browser is still configured to use proxy, but with the tunnel not running any more, the browser cannot use local port 9000 to get the result.

The screenshot shows a terminal window with the following content:

```
PES2201800368Nitish@VM1:~$ssh -D 9000 seed@10.0.2.7  
seed@10.0.2.7's password:  
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)  
  
 * Documentation: https://help.ubuntu.com  
 * Management: https://landscape.canonical.com  
 * Support: https://ubuntu.com/advantage  
  
1 package can be updated.  
0 updates are security updates.  
  
Last login: Fri Feb 19 22:53:35 2021 from 10.0.2.4  
PES2201800368Nitish@VM3:~$exit ←  
logout  
Connection to 10.0.2.7 closed.  
PES2201800368Nitish@VM1:~$
```

The terminal window has a dark background and light-colored text. A red arrow points to the exit command in the terminal history.

SCREENSHOT SHOWING THE DISABLING OF THE TUNNEL



SCREENSHOT SHOWING THE RESULT OF DISABLING THE TUNNEL AND RETAINING THE PROXY SETTINGS

On re-enabling the tunnel and retaining the proxy settings, we once again obtain the www.google.com page. If the tunnel is established but the proxy settings are not configured (i.e. left to the default option which is “use system proxy settings”), we once again get a connection timed out message and the webpage isn’t obtained.

Observation from Task 3b:

The above task is another example of evading egress firewall. We have shown above a situation where a firewall rule is configured to prevent access to a website www.google.com from the machine. But in order to evade this firewall and access the website, we establish a ssh tunnel using the command used above and also configure the proxy settings in the browser to use the tunnel. We observe from the screenshots that only when both the tunnel is established and the proxy settings are configured, will the access to browser happen and either one of them not done, will lead to failure of connecting to the website. This method is helpful when the companies or organizations block access to certain websites within the limits using a firewall and this technique can be used to evade the firewall and ensure access to the website.

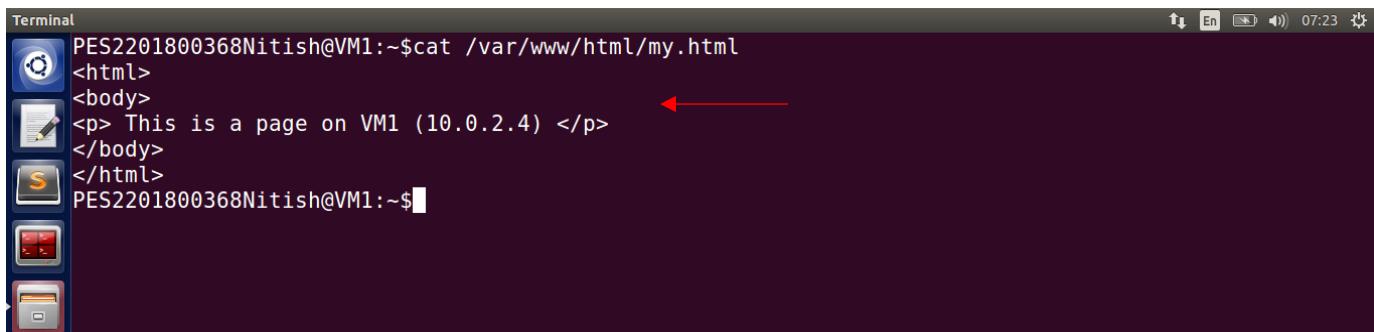
TASK 4: EVADE INGRESS FILTERING

In this task, we will block incoming port 80 and port 22 on VM1, but still access a web page on the web server in VM1 from VM3 by using a reverse *ssh* tunnel.

Setup: VM1 = 10.0.2.4 ; VM3 = 10.0.2.7

We firstly delete all the previously configured firewall rules.

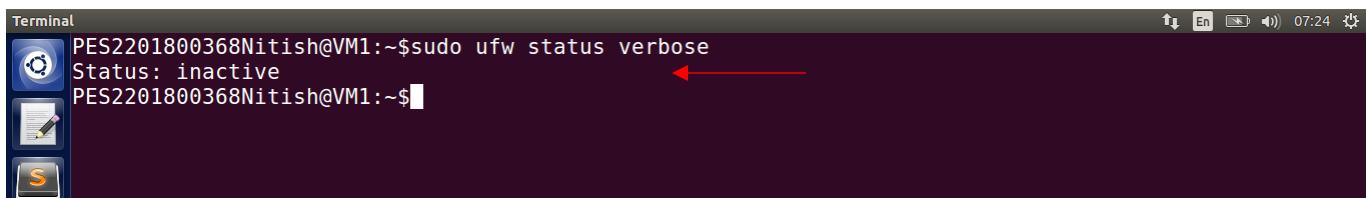
Our goal is to access a secret page on VM1 (my.html) from VM3. The content of the page is shown below:



A screenshot of a terminal window titled "Terminal". The window shows the command "cat /var/www/html/my.html" being run by user Nitish@VM1. The output is an HTML file with a single paragraph: <p> This is a page on VM1 (10.0.2.4) </p>. A red arrow points to the closing tag "</p>".

```
PES2201800368Nitish@VM1:~$ cat /var/www/html/my.html
<html>
<body>
<p> This is a page on VM1 (10.0.2.4) </p>
</body>
</html>
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE my.html WEBPAGE

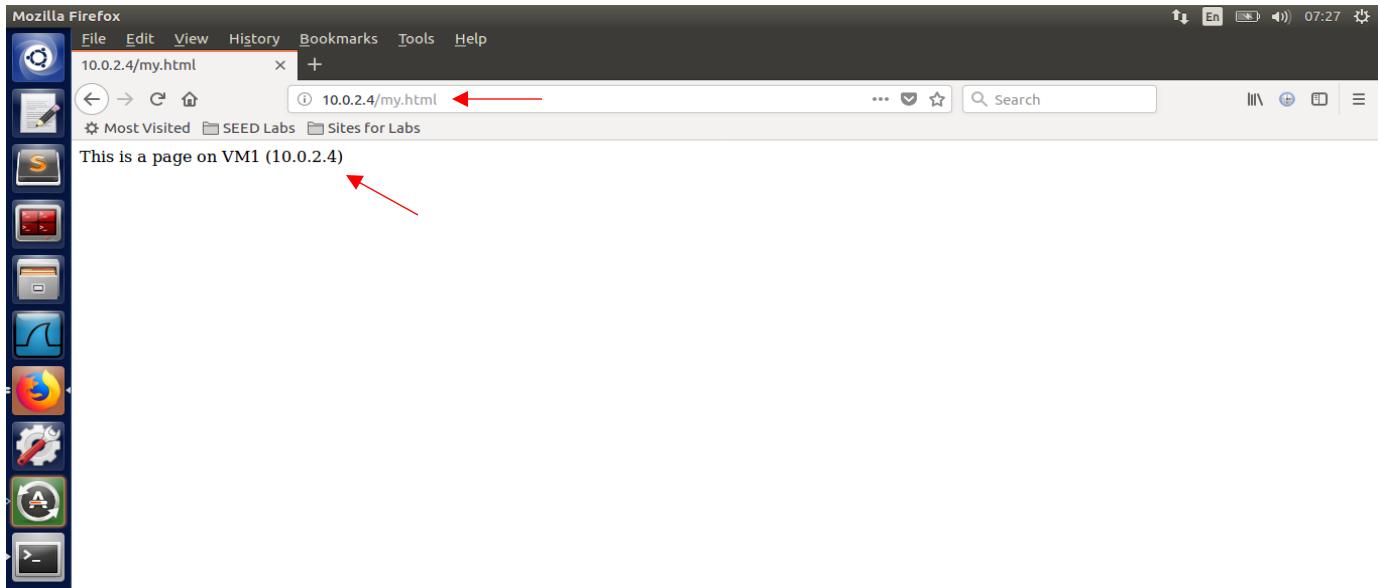


A screenshot of a terminal window titled "Terminal". The command "sudo ufw status verbose" is run by user Nitish@VM1. The output shows the status as "inactive". A red arrow points to the word "inactive".

```
PES2201800368Nitish@VM1:~$ sudo ufw status verbose
Status: inactive
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING NO RULES CONFIGURED IN THE FIREWALL

We now verify that the website is accessible from VM3 when there are no rules configured on the firewall.



SCREENSHOT SHOWING THE WEBPAGE FROM VM1 ACCESSIBLE FROM VM3

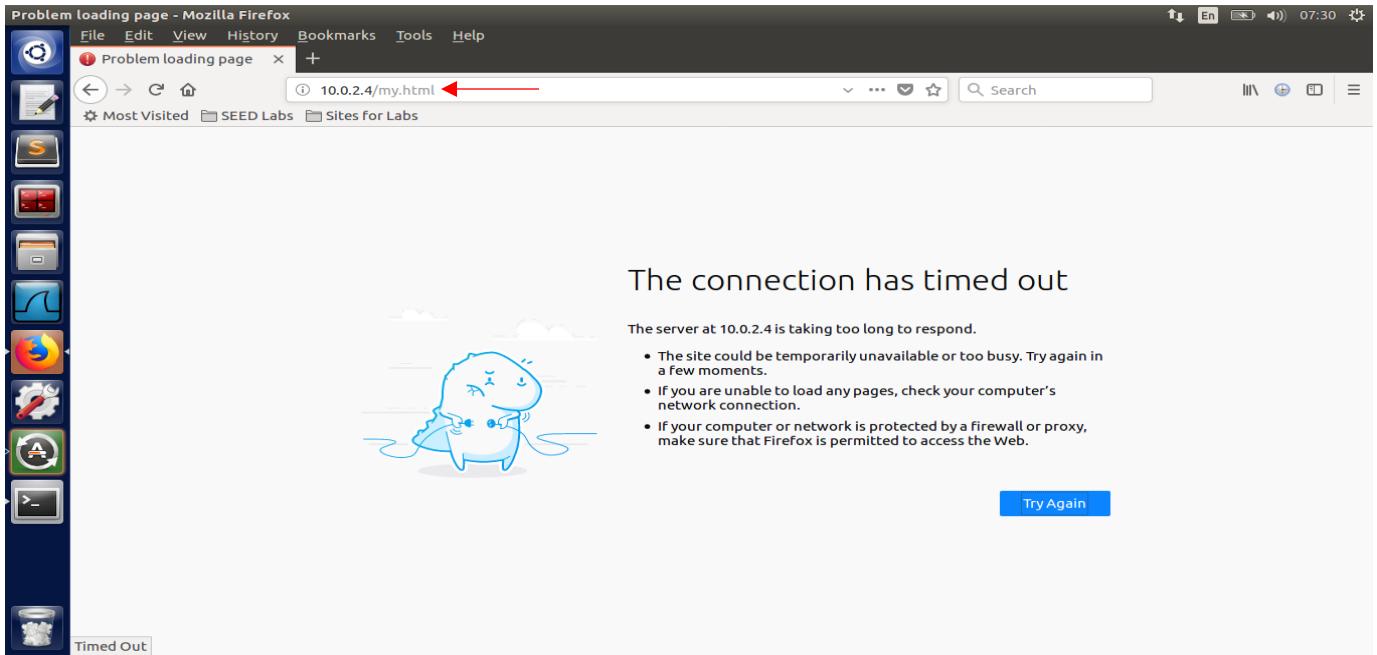
We next block incoming requests on port 80 and port 22 on VM1.

```
PES2201800368Nitish@VM1:~$sudo ufw enable
Firewall is active and enabled on system startup
PES2201800368Nitish@VM1:~$sudo ufw deny in from any to 10.0.2.4 port 80
Rule added
PES2201800368Nitish@VM1:~$sudo ufw deny in from any to 10.0.2.4 port 22
Rule added
PES2201800368Nitish@VM1:~$sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
To           Action      From
--           -----      ---
10.0.2.4 80  DENY IN   Anywhere
10.0.2.4 22  DENY IN   Anywhere
PES2201800368Nitish@VM1:~$
```

SCREENSHOT SHOWING THE NEWLY ADDED RULES

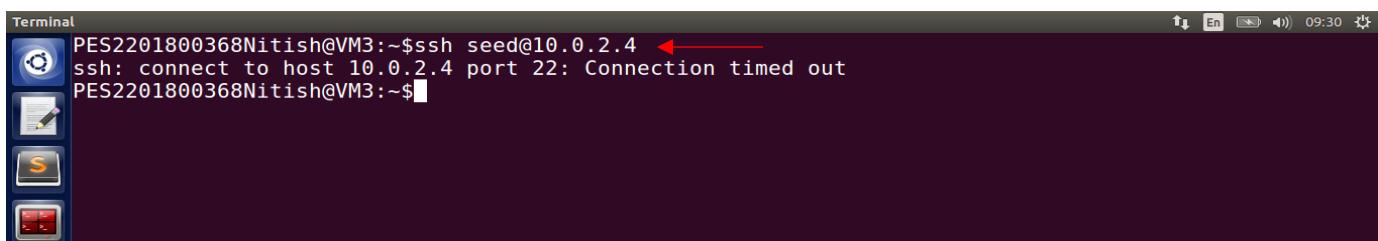
Given the firewall rules in place on VM1, we next check if the page is still accessible. We clear the browser cache.

If we try to access the page again from VM3, the page is no longer accessible.



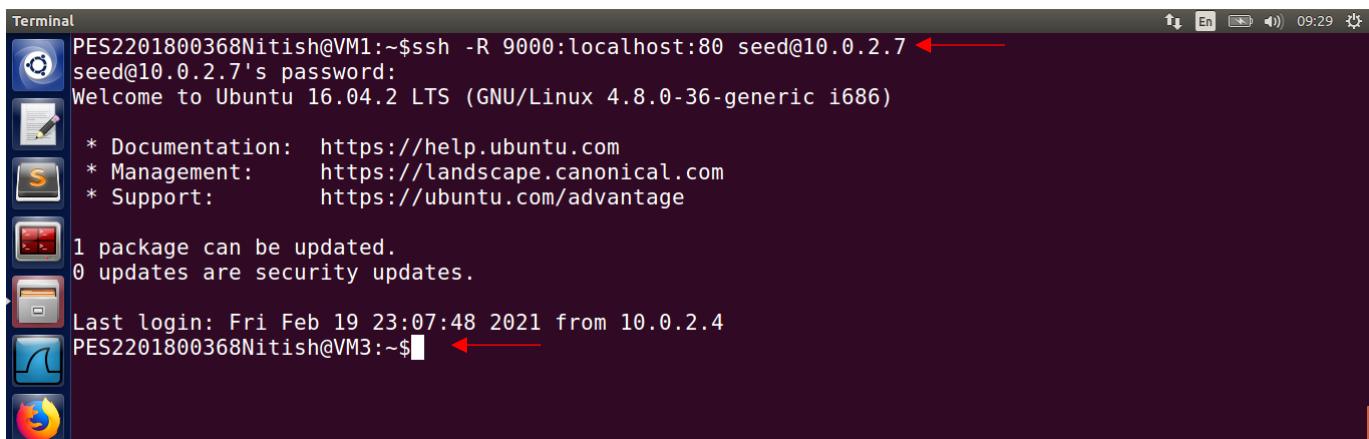
SCREENSHOT SHOWING CONNECTION TIMED OUT ERROR WHEN THE WEBPAGE IS ACCESSED

Also, because port 22 is blocked, VM3 cannot *ssh* to connect to VM1.



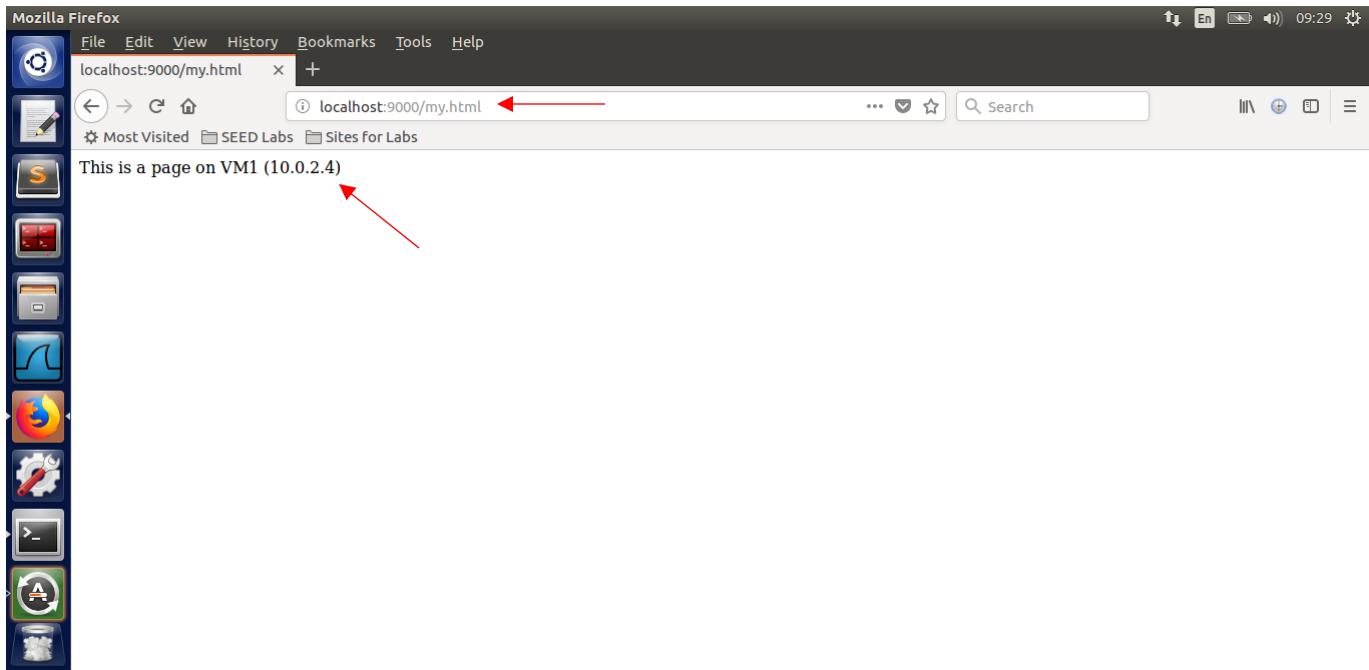
SCREENSHOT SHOWING FAILED ssh CONNECTION TO VM1 DUE TO FIREWALL RULE

We next set up a reverse tunnel. Using this VM3 can use its local port 9000 to access port 80 on VM1.



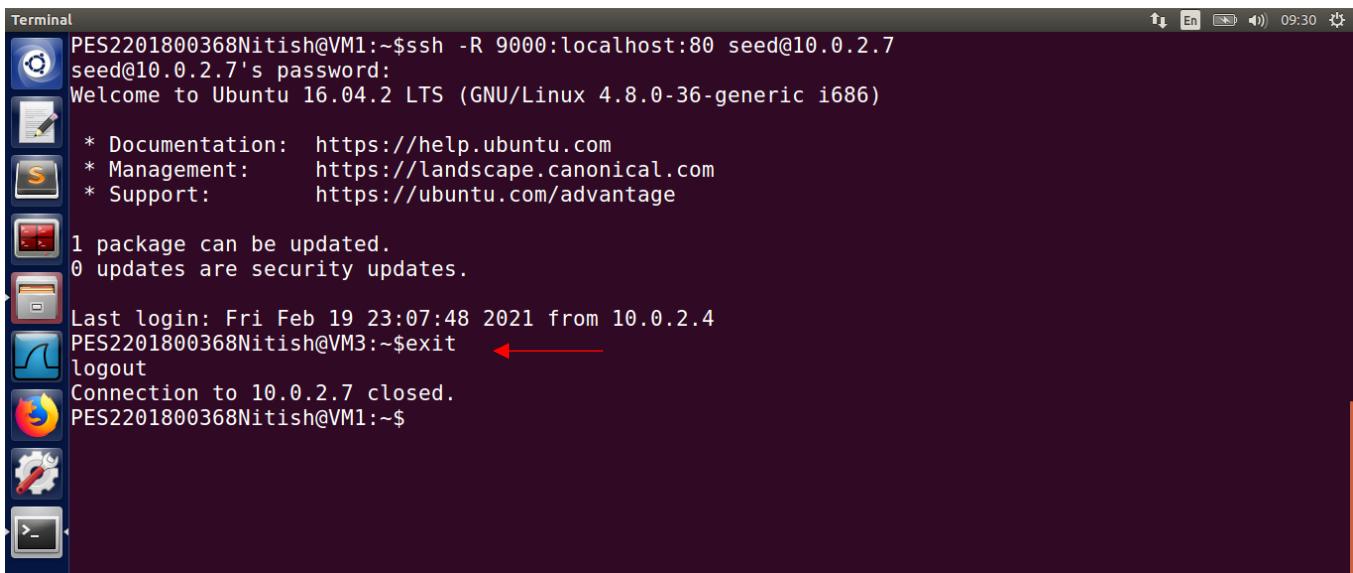
SCREENSHOT SHOWING THE ESTABLISHED REVERSE TUNNEL

With the tunnel setup, we test whether we can access the webpage using port 9000 on VM3.



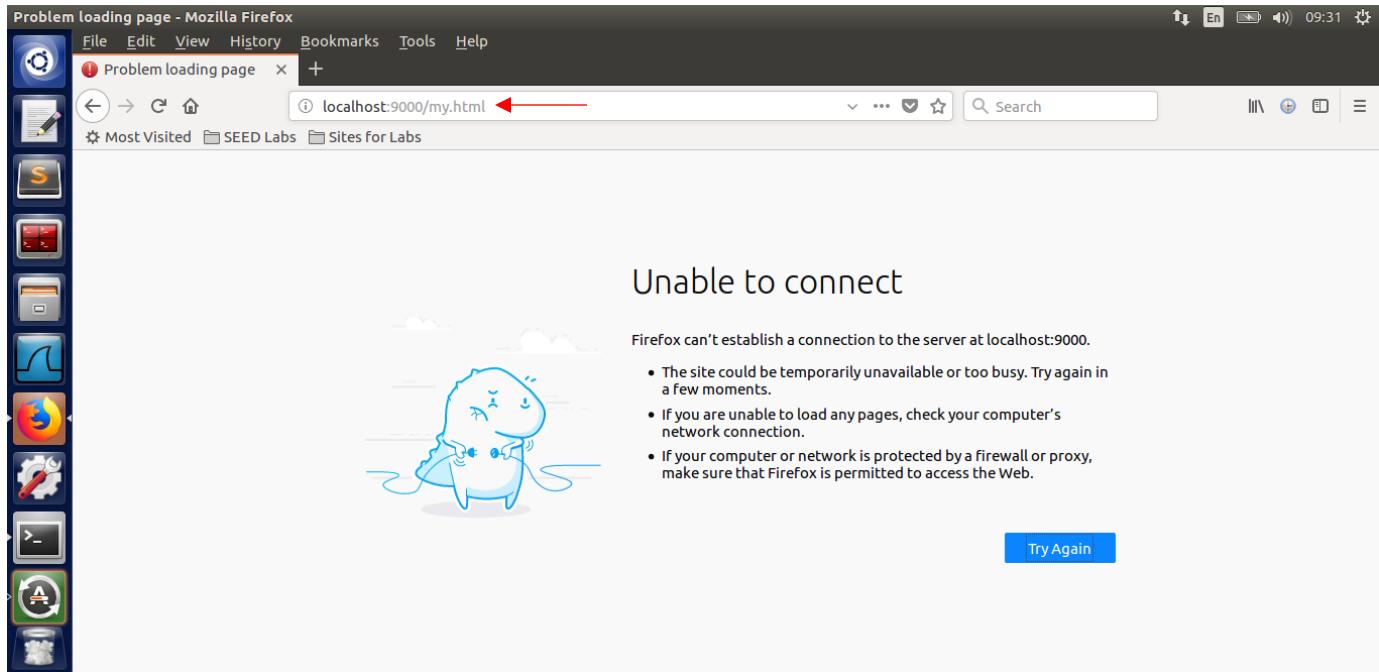
SCREENSHOT SHOWING THE WEBPAGE ACCESS AVAILABLE ON PORT 9000 DUE TO THE ESTABLISHED REVERSE TUNNEL

We next break the tunnel to see its effect (on VM1)



SCREENSHOT SHOWING THE BREAKING OF THE ESTABLISHED REVERSE TUNNEL

Testing the effect of breaking the reverse tunnel



SCREENSHOT SHOWING FAILURE TO ACCESS THE WEBPAGE FROM VM3 DUE TO BREAKING OF THE REVERSE TUNNEL

Observation from Task4:

The aim of the above task was to evade the firewall preventing access to an internal website for which even incoming ssh traffic is blocked. Therefore in order to evade this firewall we create a ssh tunnel from outside and use this tunnel to do the port forwarding. We create a special type of ssh tunnel called the reverse tunnel, so when users from outside send a HTTP request to port 9000 from their browsers, the ssh tunnel will forward the request to the ssh client, which further forwards the request to port 80 of the web-server machine having the webpage and thereby the webpage can be accessed by outsiders.

Without this technique of establishing a reverse tunnel, we cannot access internal webpages of an organization protected by firewall.