New York Institute of Technology

School of Engineering and Computing Sciences

INCS 745 : Intrusion Detection and Hacker Exploits

Linux Firewall Exploration Lab

Harshita Grover (1276595)

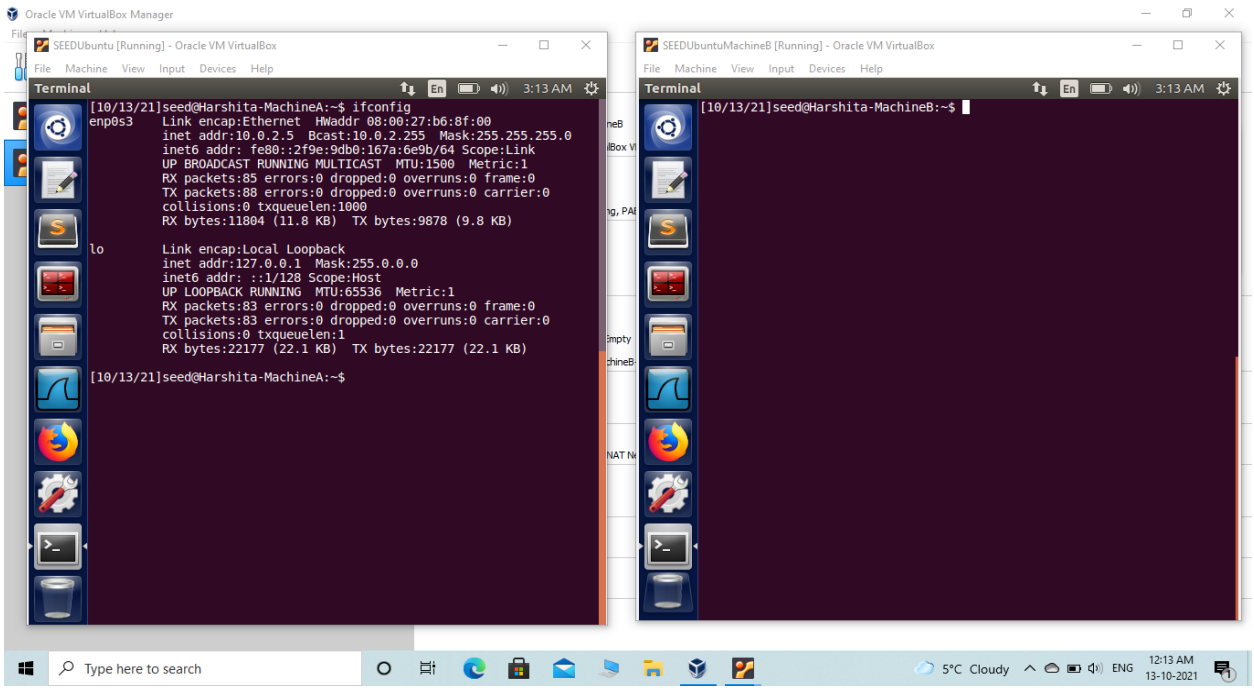
Professor: Yunlong Shao

Date Submitted:

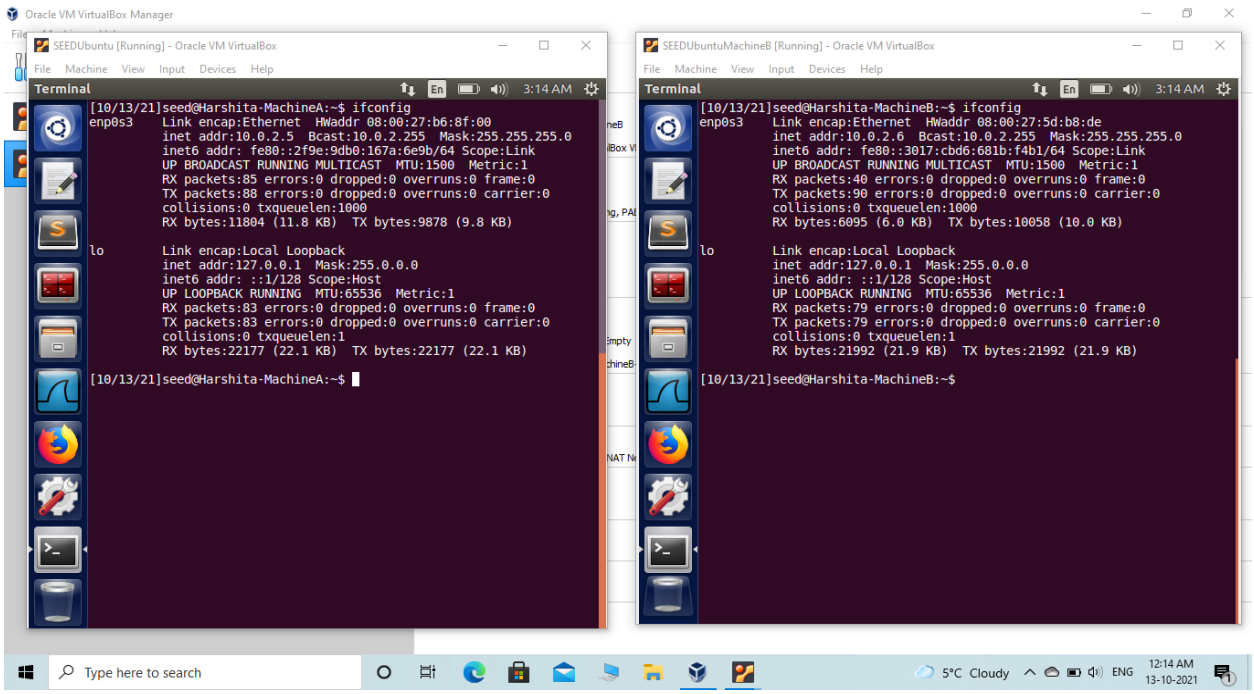
October 15, 2021

Lab 1: Linux Firewall Exploration Lab

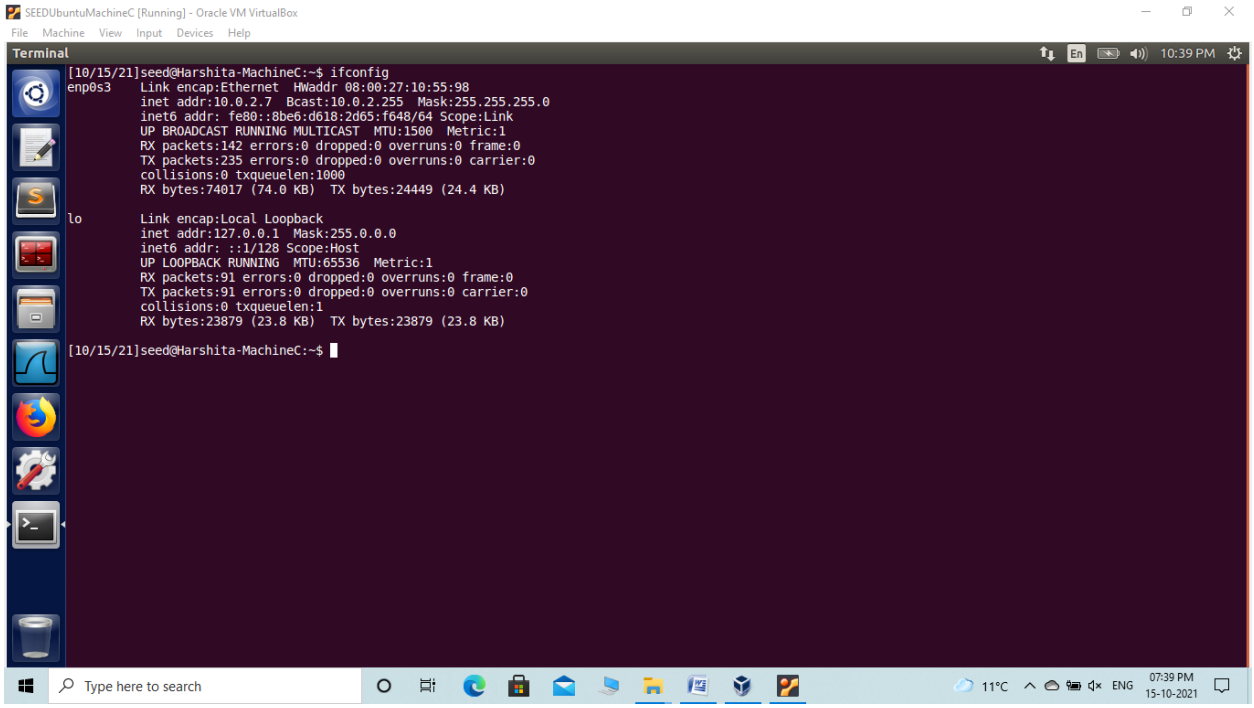
IP address of Machine A : 10.0.2.5



IP address of Machine B : 10.0.2.6



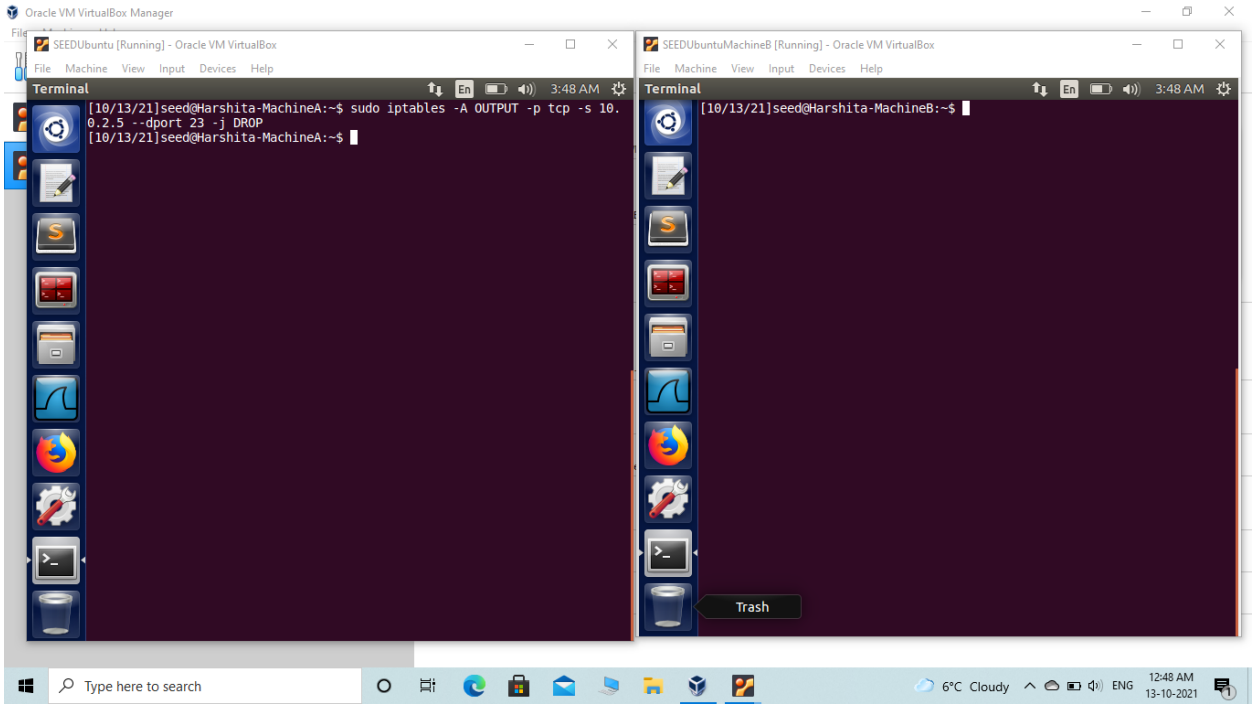
IP address of Machine C: 10.0.2.7



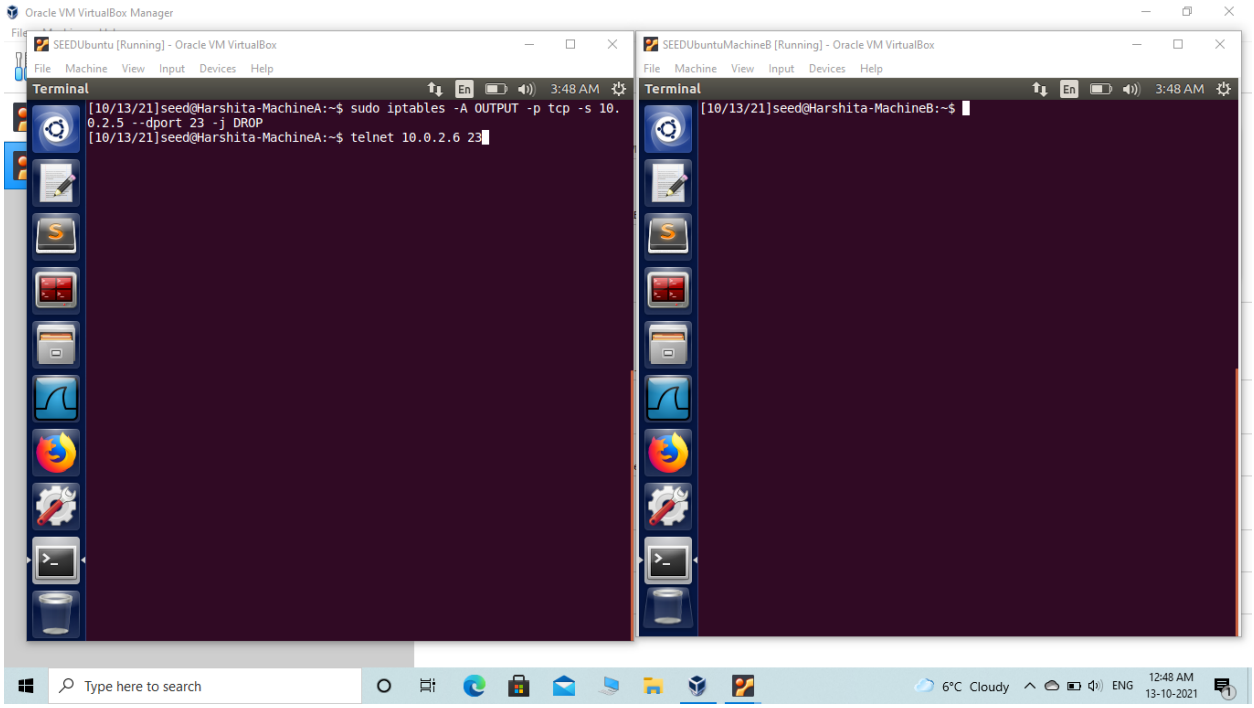
Task 1: Using Firewall

(a) Prevent Machine A from doing telnet to Machine B.

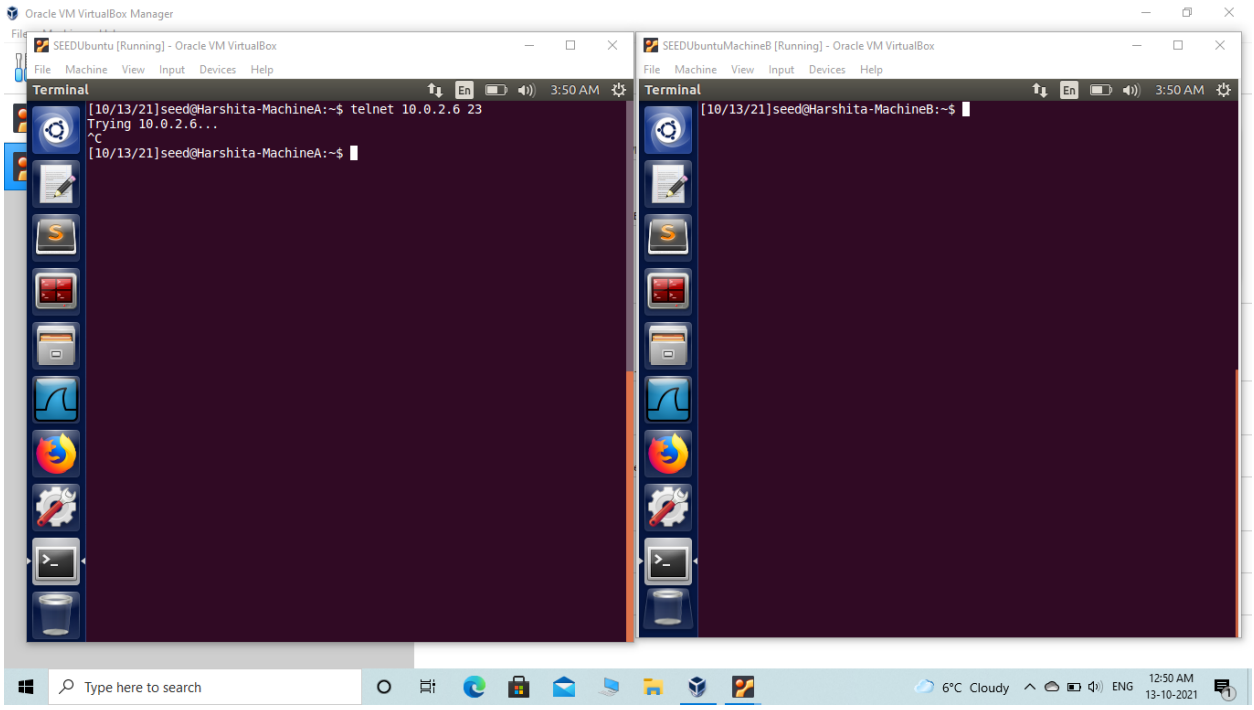
Command: sudo iptables –A OUTPUT –p tcp –s 10.0.2.5 --dport 23 –j DROP



We then try to telnet machine B as shown in the following figure.

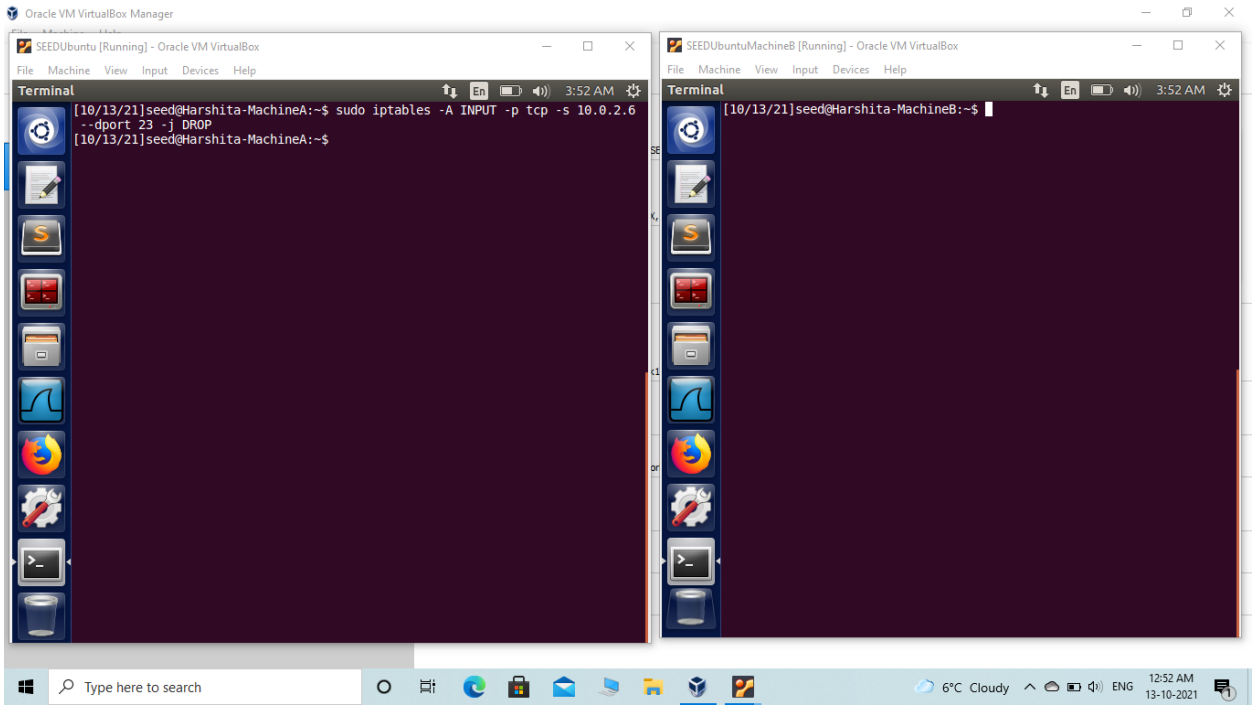


But we are not able to telnet to machine B.

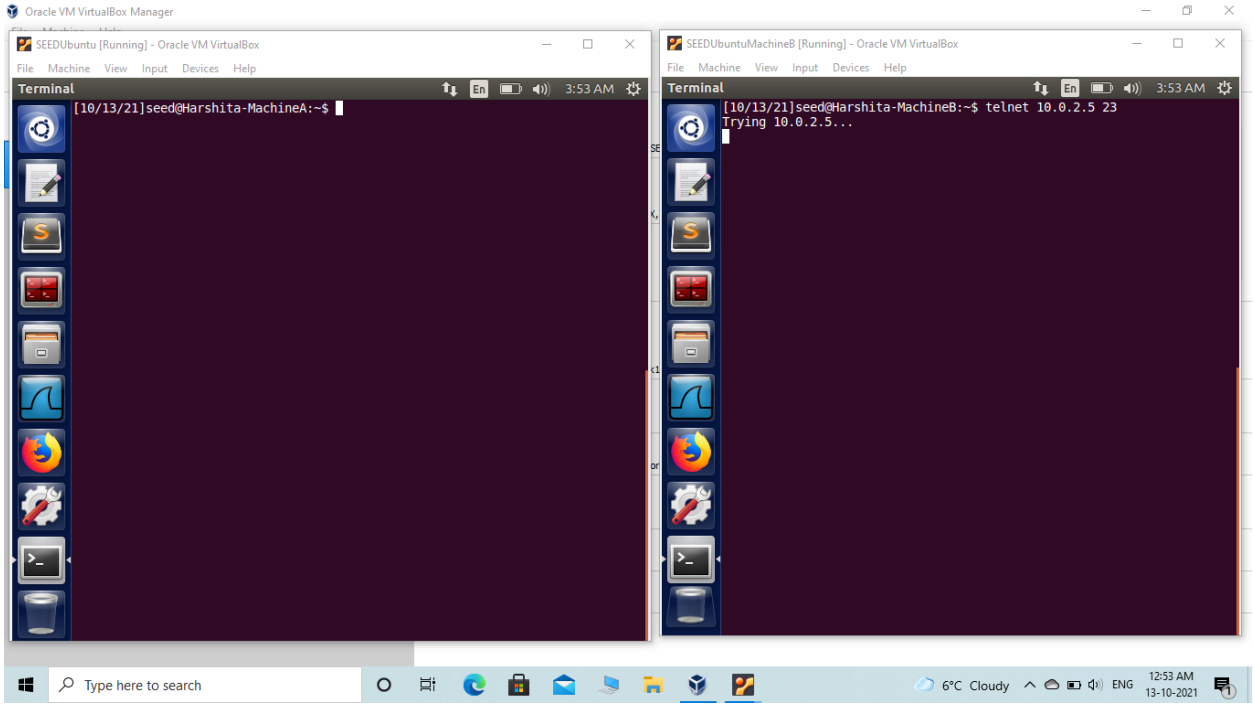


(b) Prevent Machine B from doing telnet to Machine A.

Command: sudo iptables –A INPUT –p tcp –s 10.0.2.6 --dport 23 –j DROP

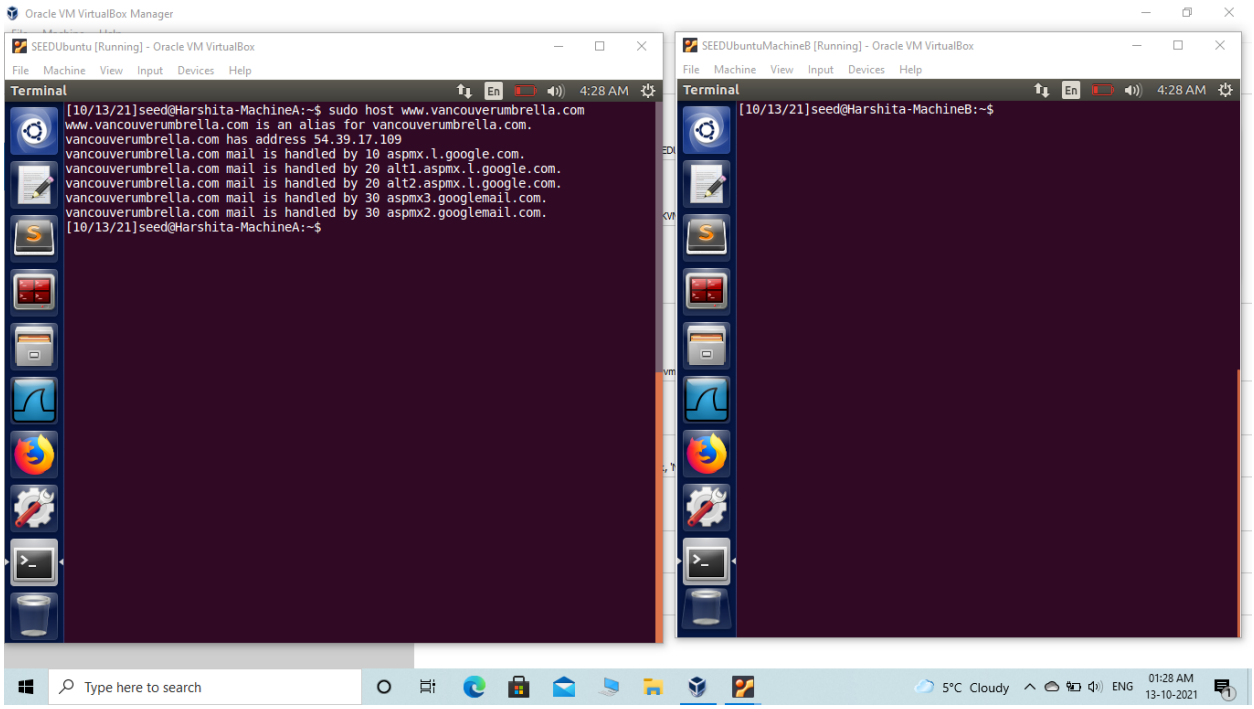


We try to telnet machine A from machine B but we are unable to telnet.

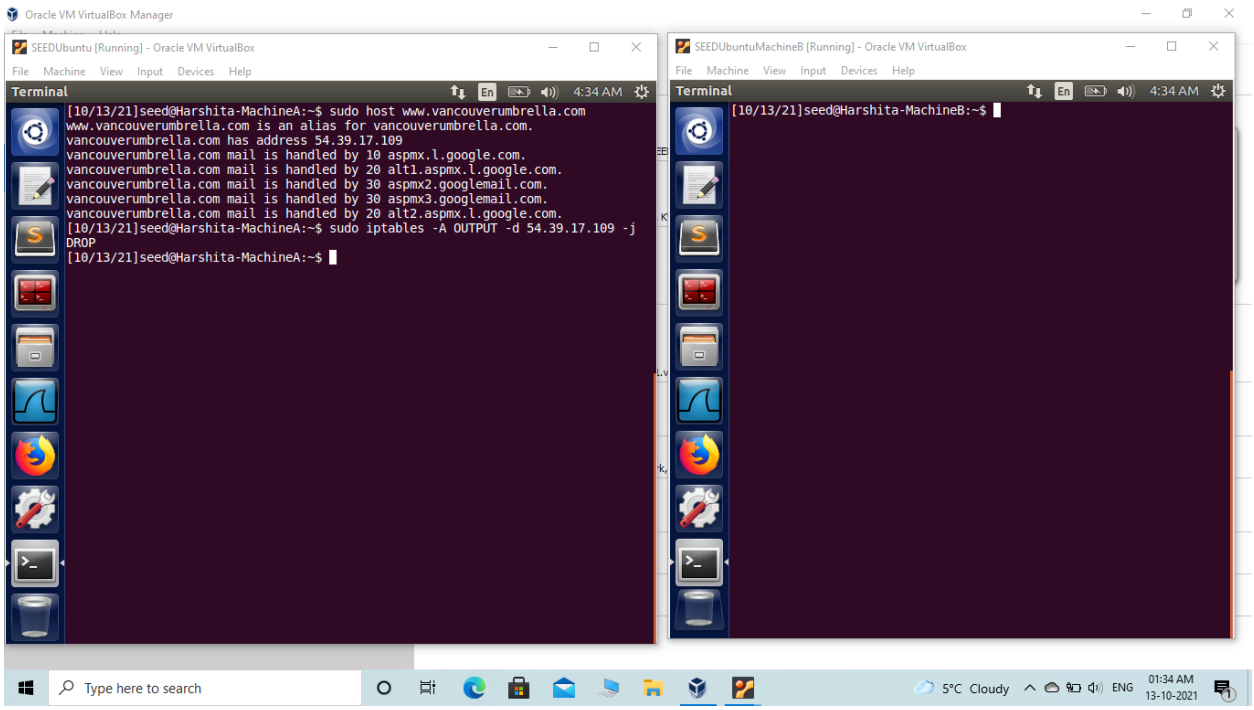


(c) Prevent A from visiting an external web site.

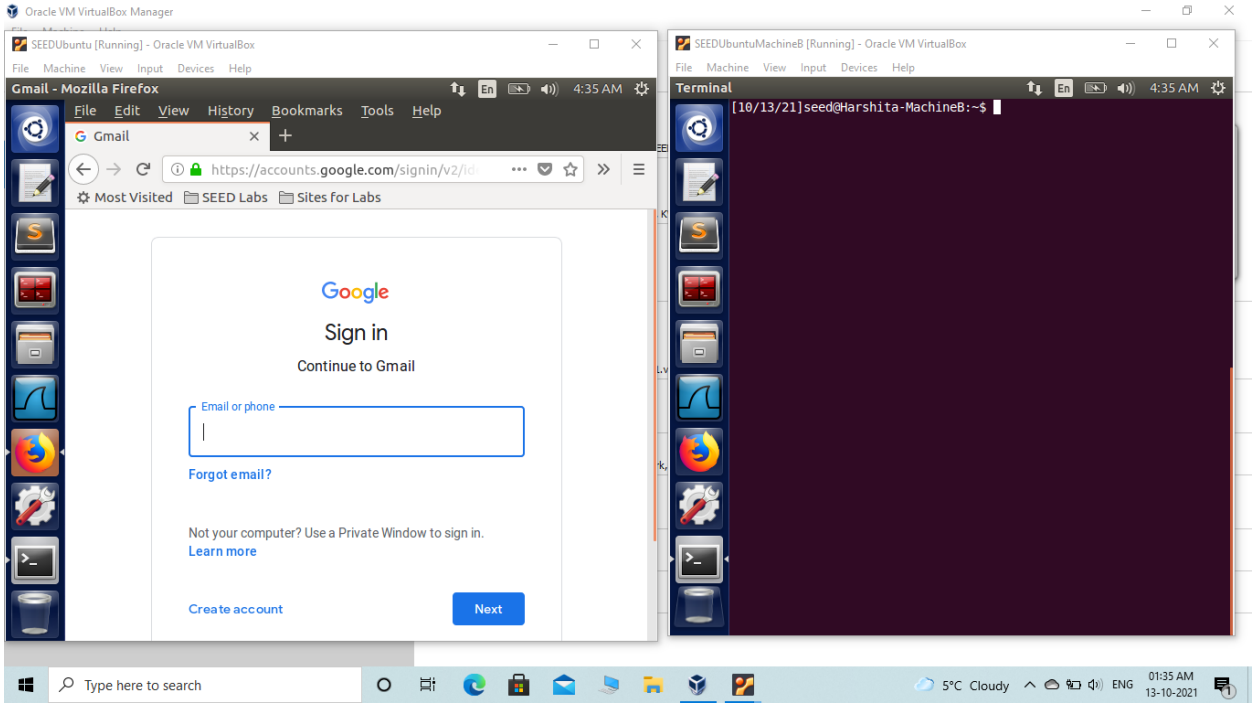
We firstly find ip address of an external website “www.vancouverumbrella.com” .



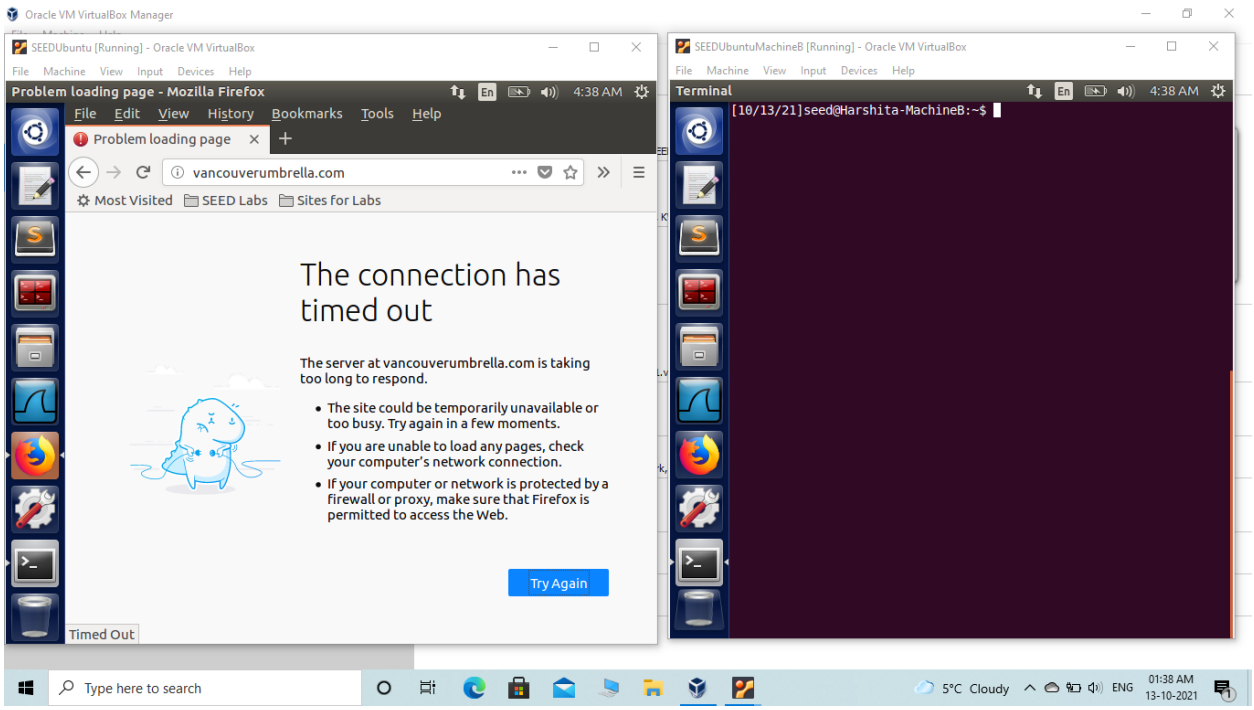
We block access of machine A to external website using iptables.



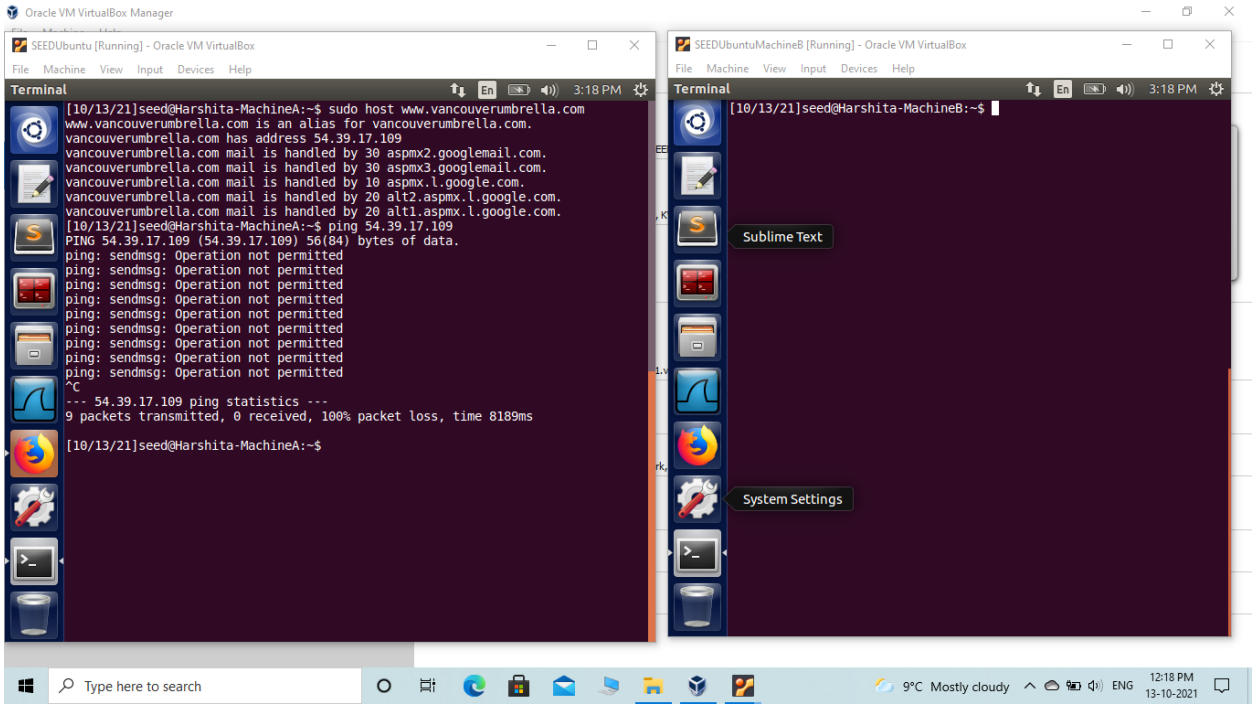
We now try to access “gmail.com” just to see that we can access any other website.



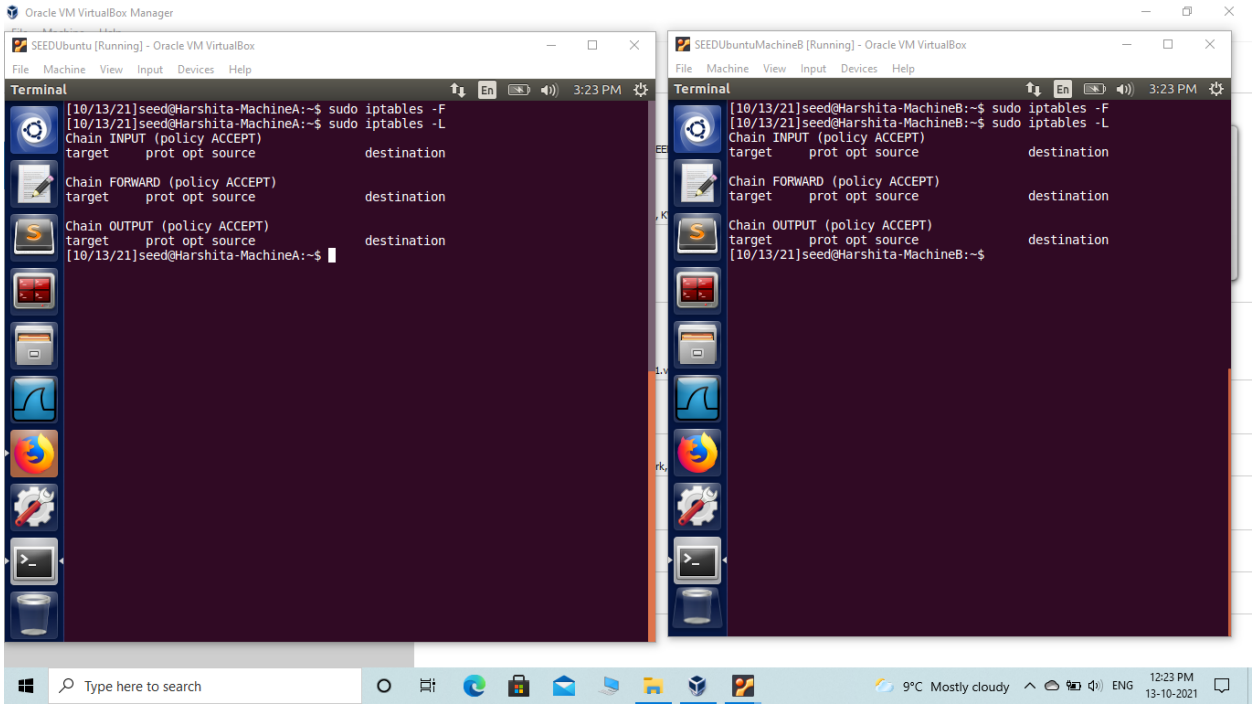
But we cannot access the external webpage that we blocked that is “www.vancouverumbrella.com”.



Similarly, we try to ping the external website "www.vancouverumbrella.com" which is not permitted as we saw in the previous screenshots.

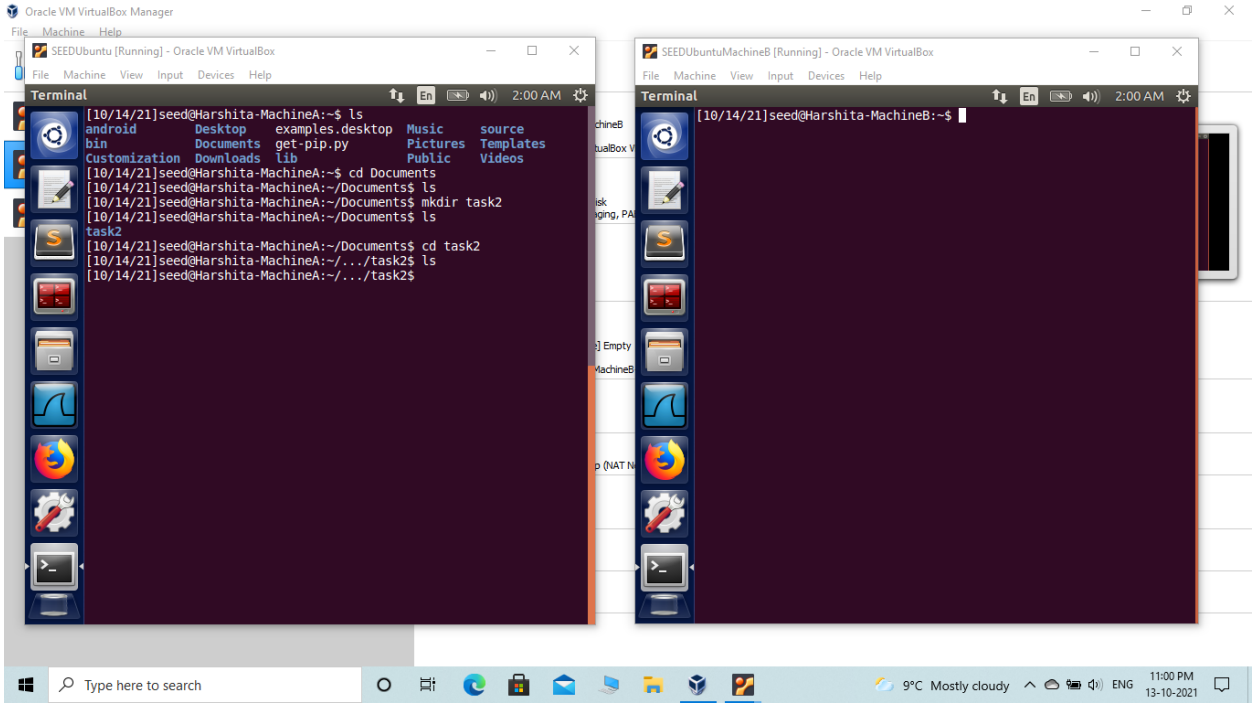


Now, we remove all the iptable rules using “iptables –F”.

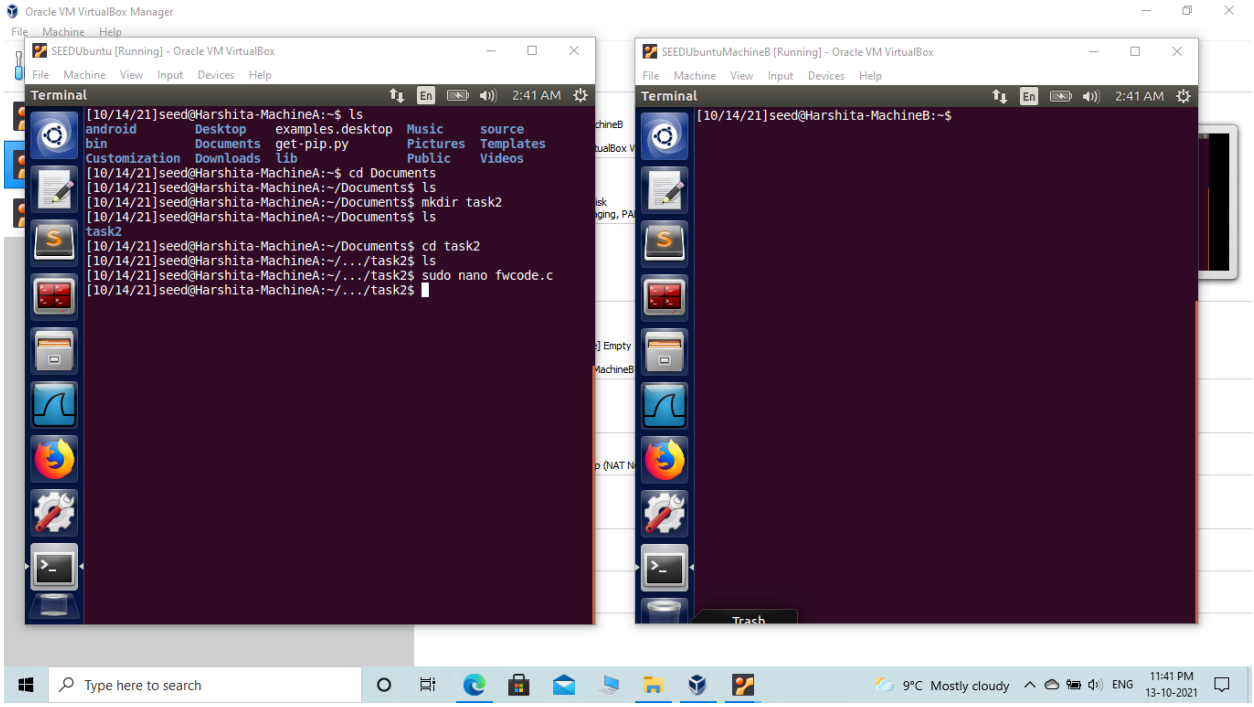


Task 2: Implementing a Simple Firewall

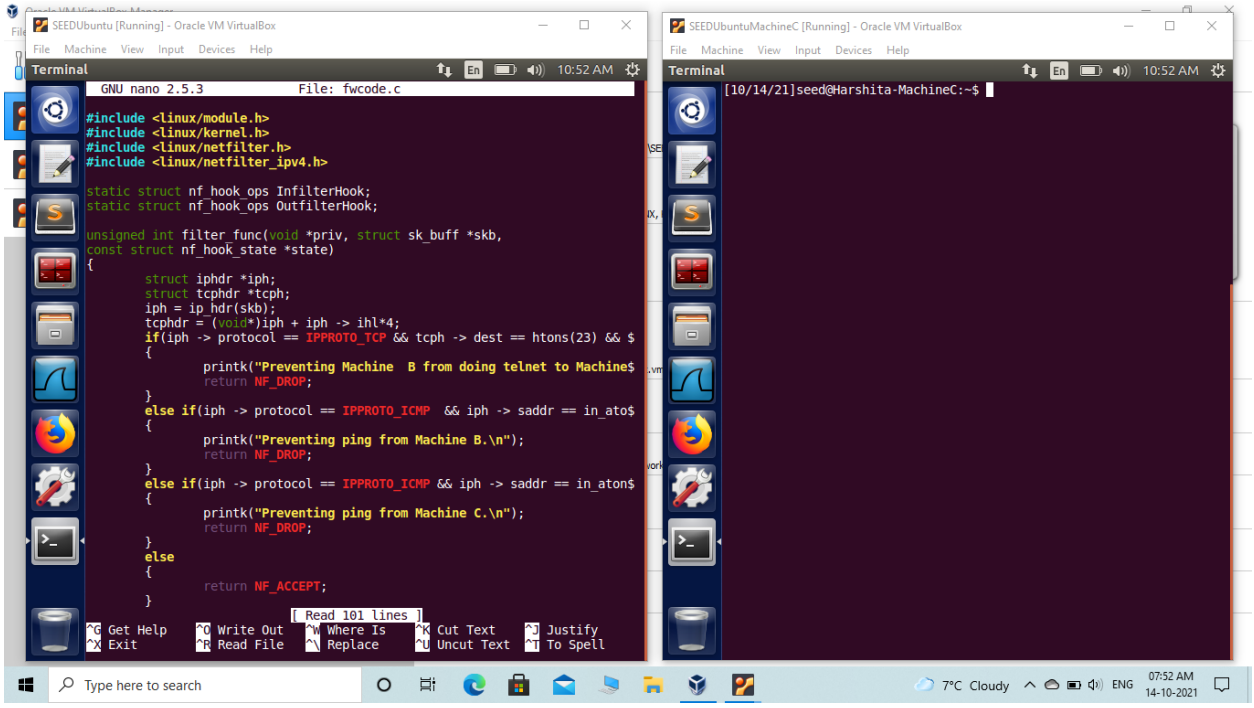
We firstly create a folder named “task2” in Documents folder.

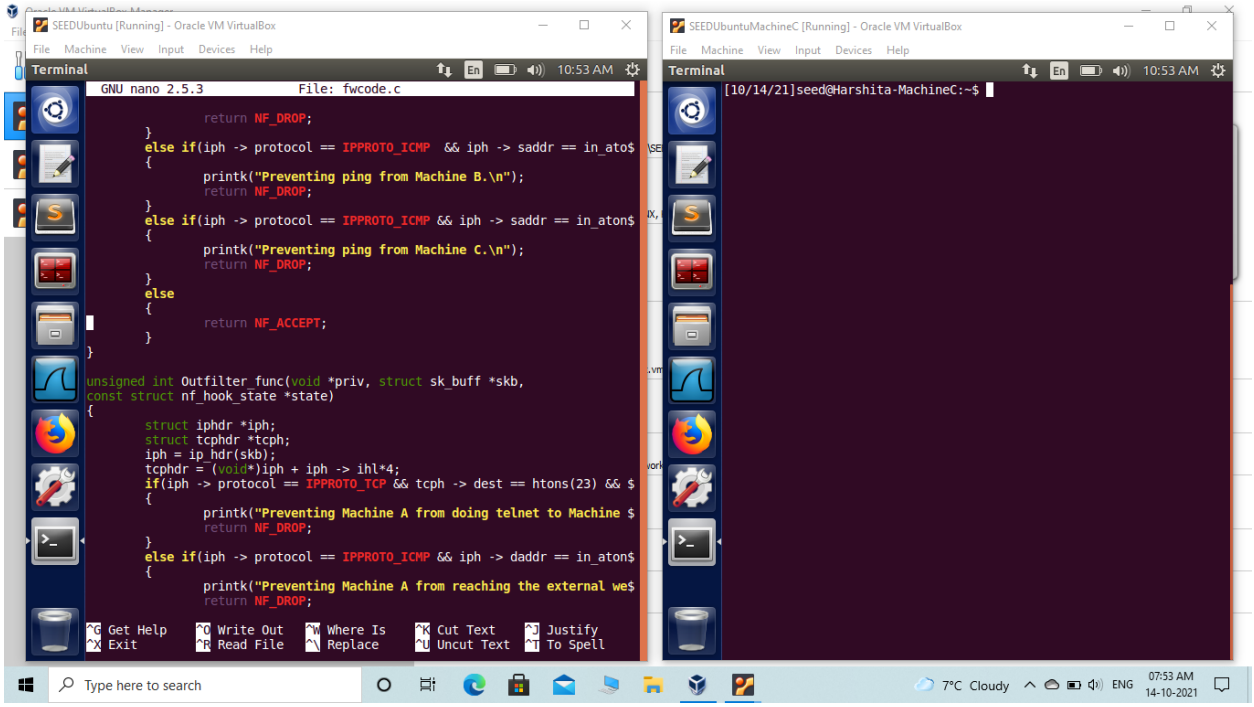


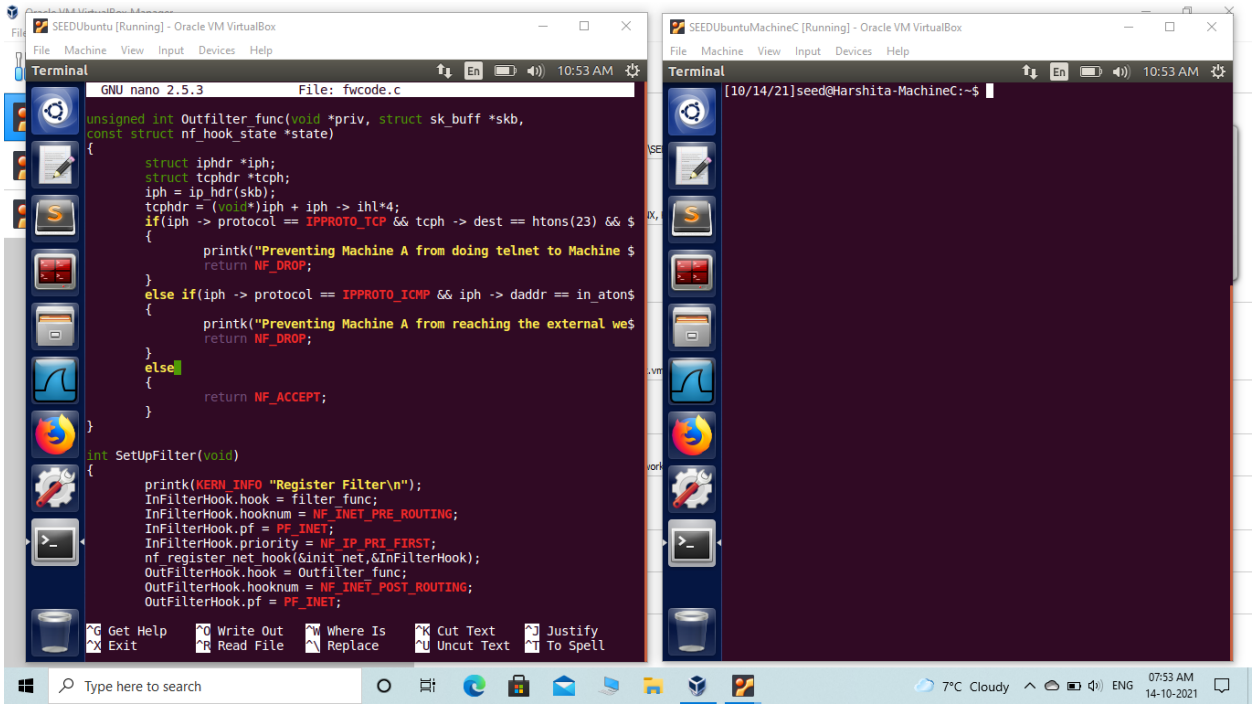
Now, we create a “fwcode.c” file for our netfilter firewall rules.

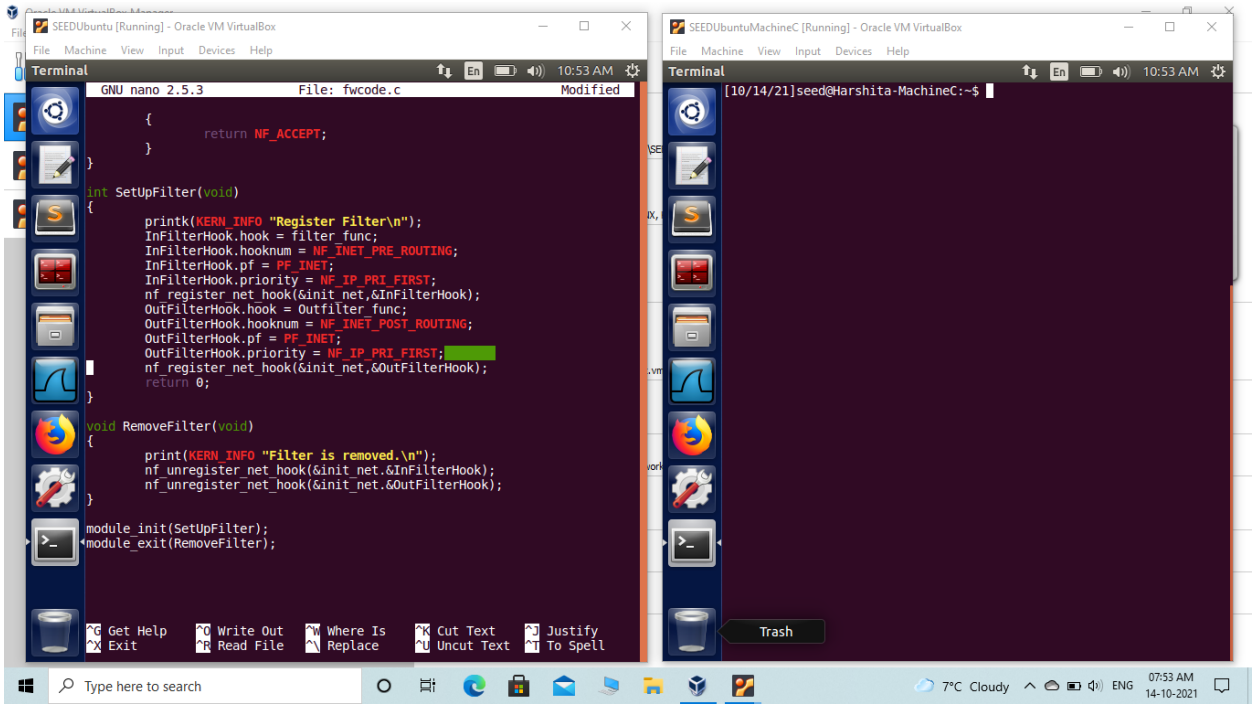


This is the code for our file “fwcode.c”.

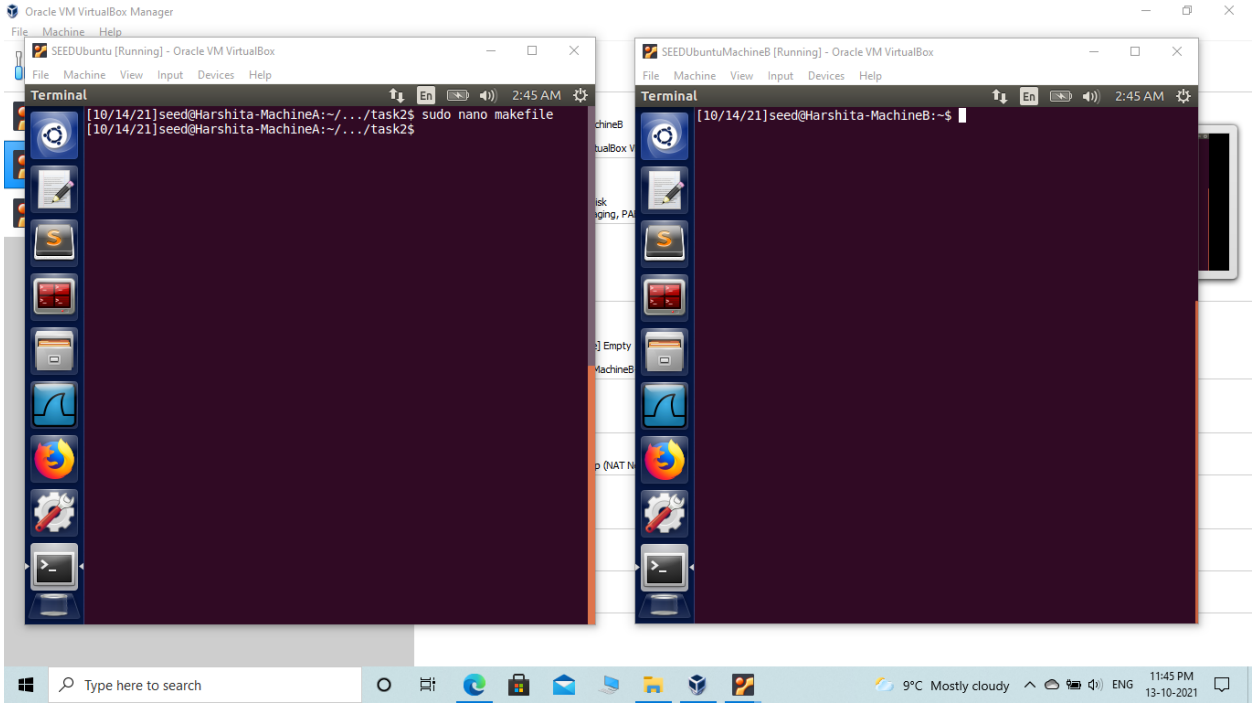



SEEDUbuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
10:53 AM
GNU nano 2.5.3 File: fwcode.c
return NF_DROP;
}
else if(iph -> protocol == IPPROTO_ICMP && iph -> saddr == in_ato$
{
printk("Preventing ping from Machine B.\n");
return NF_DROP;
}
else if(iph -> protocol == IPPROTO_ICMP && iph -> saddr == in_aton$
{
printk("Preventing ping from Machine C.\n");
return NF_DROP;
}
else
{
return NF_ACCEPT;
}
}
unsigned int Outfilter_func(void *priv, struct sk_buff *skb,
const struct nf_hook_state *state)
{
struct iphdr *iph;
struct tcphdr *tcph;
iph = ip_hdr(skb);
tcphdr = (void*)iph + iph -> ihl*4;
if(iph -> protocol == IPPROTO_TCP && tcph -> dest == htons(23) && $
{
printk("Preventing Machine A from doing telnet to Machine $
return NF_DROP;
}
else if(iph -> protocol == IPPROTO_ICMP && iph -> daddr == in_aton$
{
printk("Preventing Machine A from reaching the external we$
return NF_DROP;
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell
SEEDUbuntuMachineC [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
10:53 AM
[10/14/21]seed@Harshita-MachineC:~$
Type here to search
7°C Cloudy
ENG
07:53 AM
14-10-2021

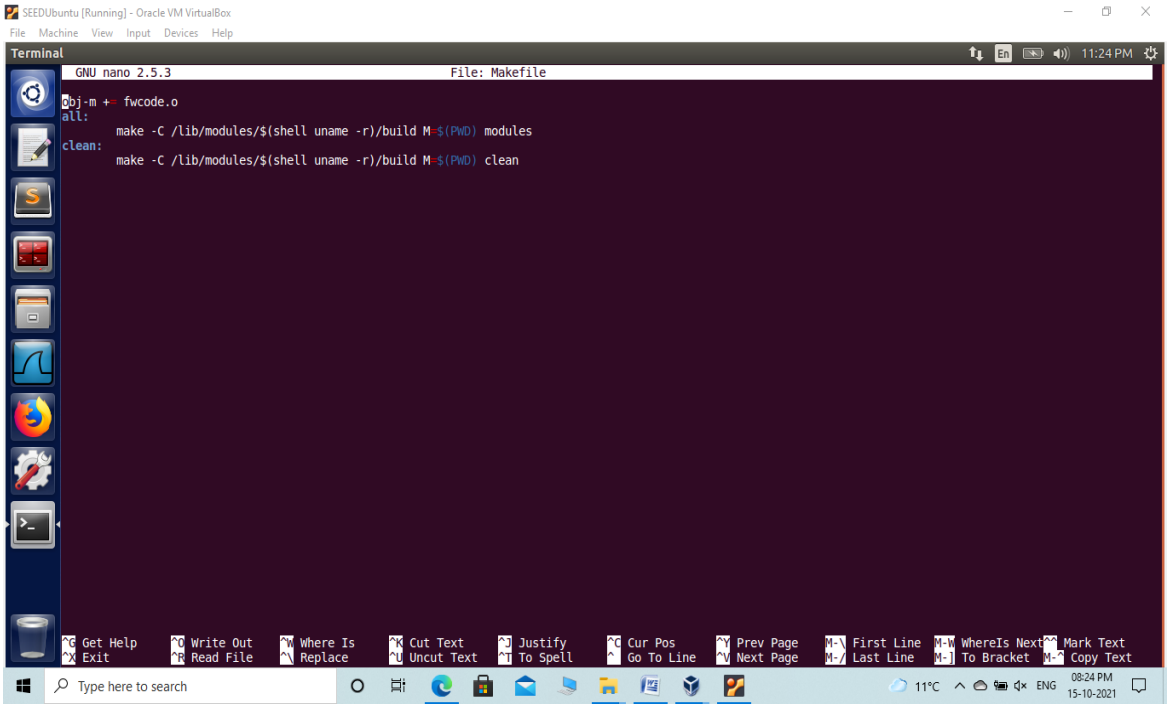

SEEDUbuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
10:53 AM
GNU nano 2.5.3 File: fwcode.c
unsigned int Outfilter_func(void *priv, struct sk_buff *skb,
const struct nf_hook_state *state)
{
struct iphdr *iph;
struct tcphdr *tcph;
iph = ip_hdr(skb);
tcphdr = (void*)iph + iph -> ihl*4;
if(iph -> protocol == IPPROTO_TCP && tcph -> dest == htons(23) && $
{
printk("Preventing Machine A from doing telnet to Machine $
return NF_DROP;
}
else if(iph -> protocol == IPPROTO_ICMP && iph -> daddr == in_aton$
{
printk("Preventing Machine A from reaching the external we$
return NF_DROP;
}
else
{
return NF_ACCEPT;
}
}
int SetUpFilter(void)
{
printk(KERN_INFO "Register Filter\n");
InFilterHook.hook = filter_func;
InFilterHook.hooknum = NF_INET_PRE_ROUTING;
InFilterHook.pf = PF_INET;
InFilterHook.priority = NF_IP_PRI_FIRST;
nf_register_net_hook(&init_net,&InFilterHook);
OutFilterHook.hook = Outfilter_func;
OutFilterHook.hooknum = NF_INET_POST_ROUTING;
OutFilterHook.pf = PF_INET;
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell
SEEDUbuntuMachineC [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
10:53 AM
[10/14/21]seed@Harshita-MachineC:~$
Type here to search
7°C Cloudy
ENG
07:53 AM
14-10-2021



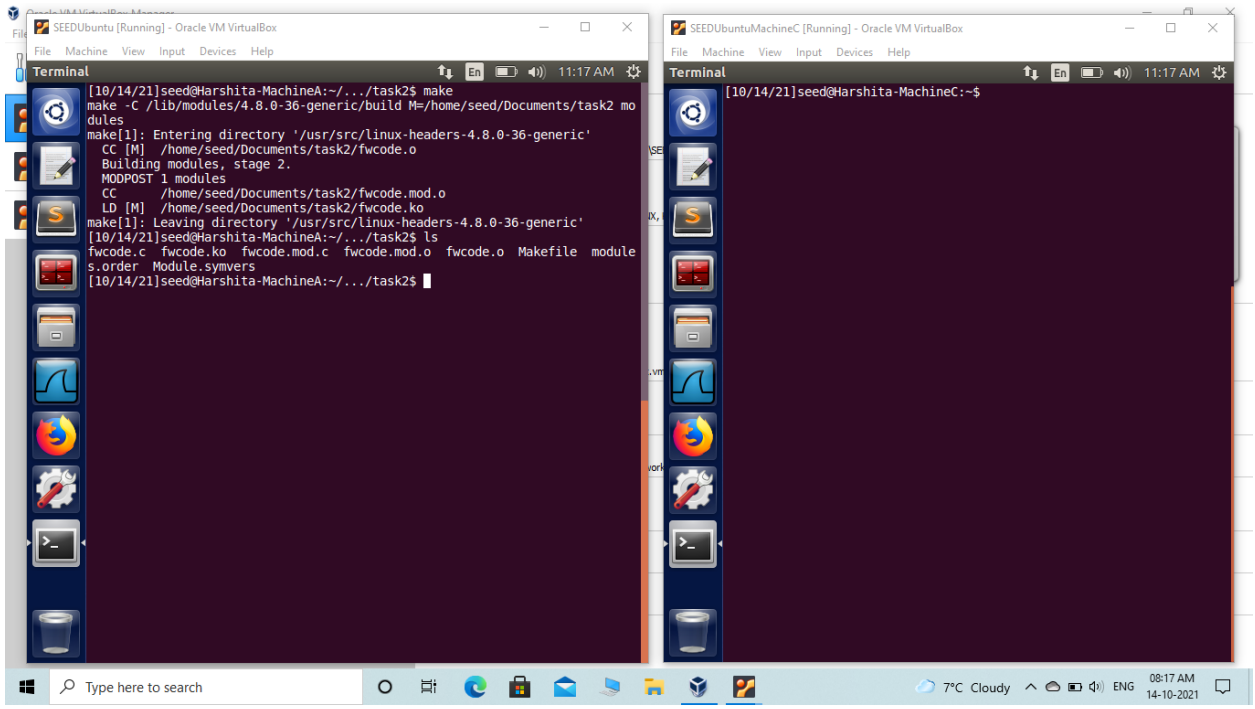
Now, we create a “Makefile”.



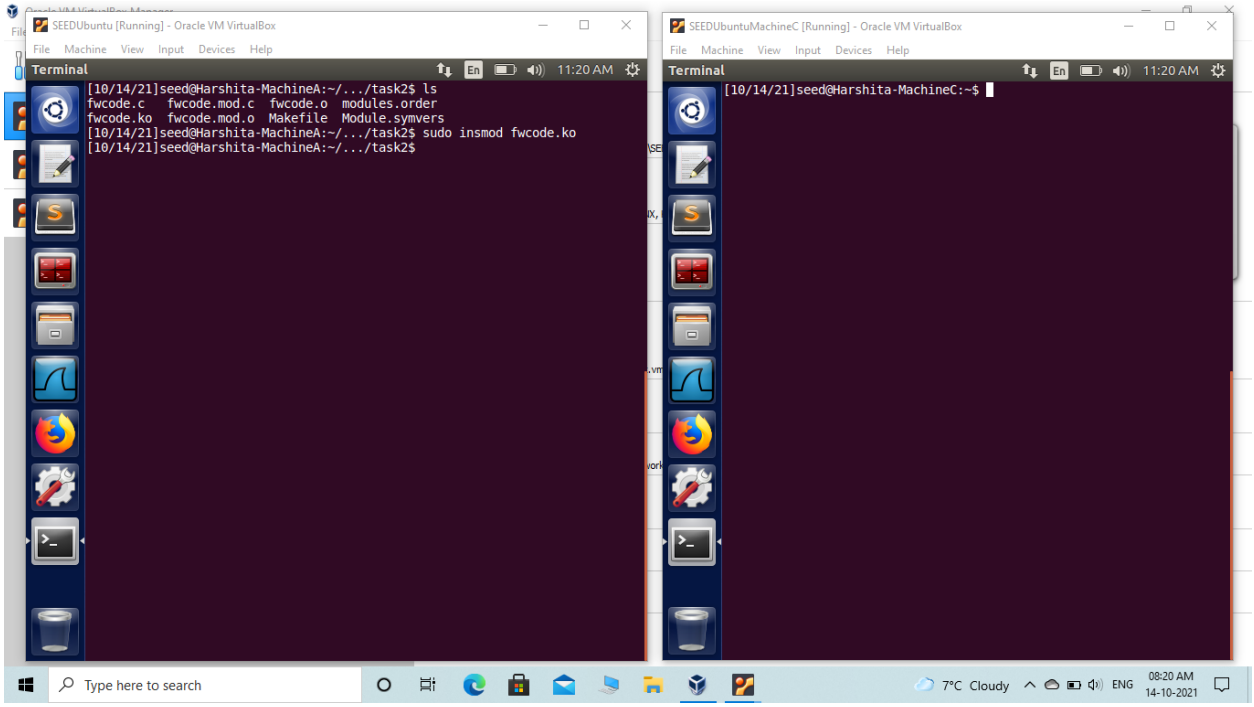
Here is the code for “Makefile”.



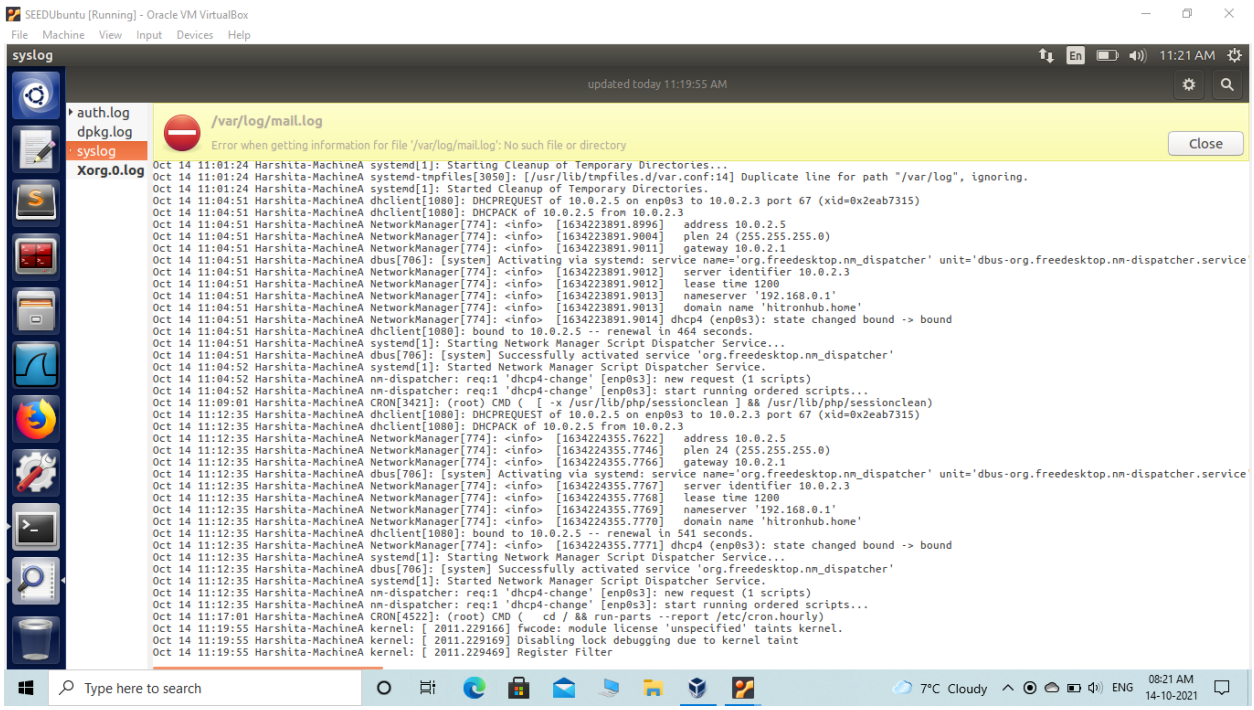
Now, we use “make” command.



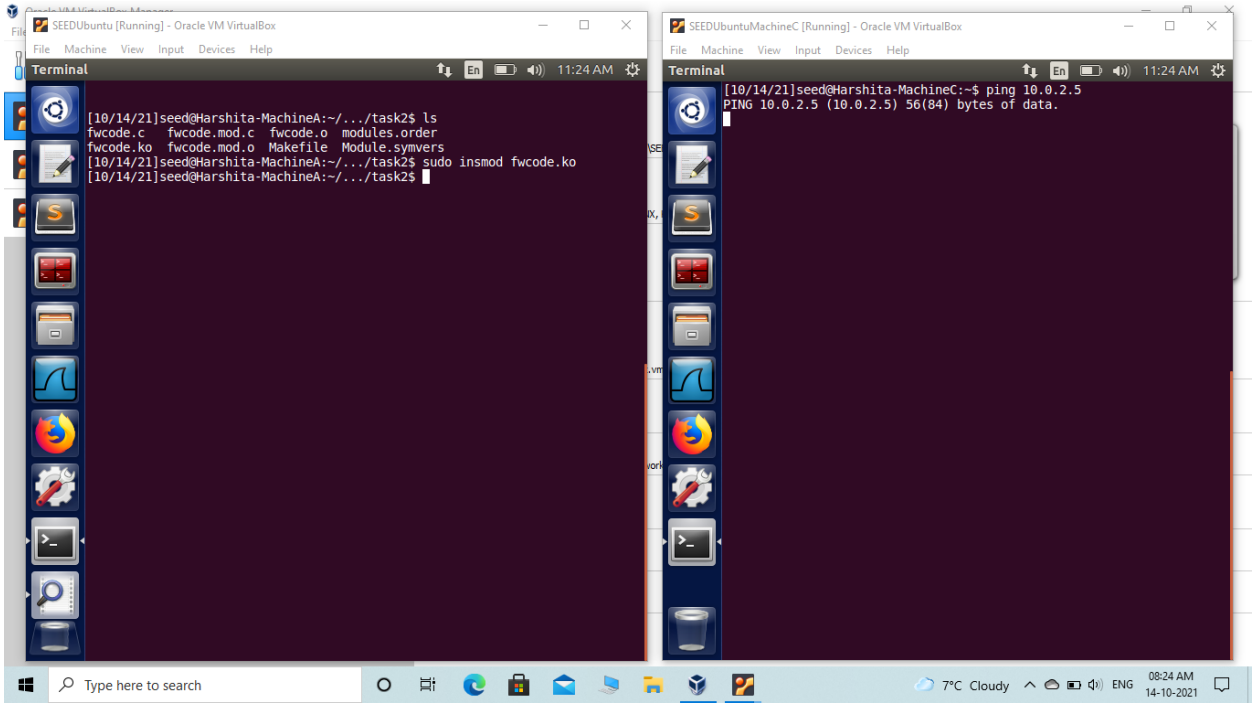
We use “sudo insmod fwcode.ko” to insert the module “fwcode.ko” into the kernel space.



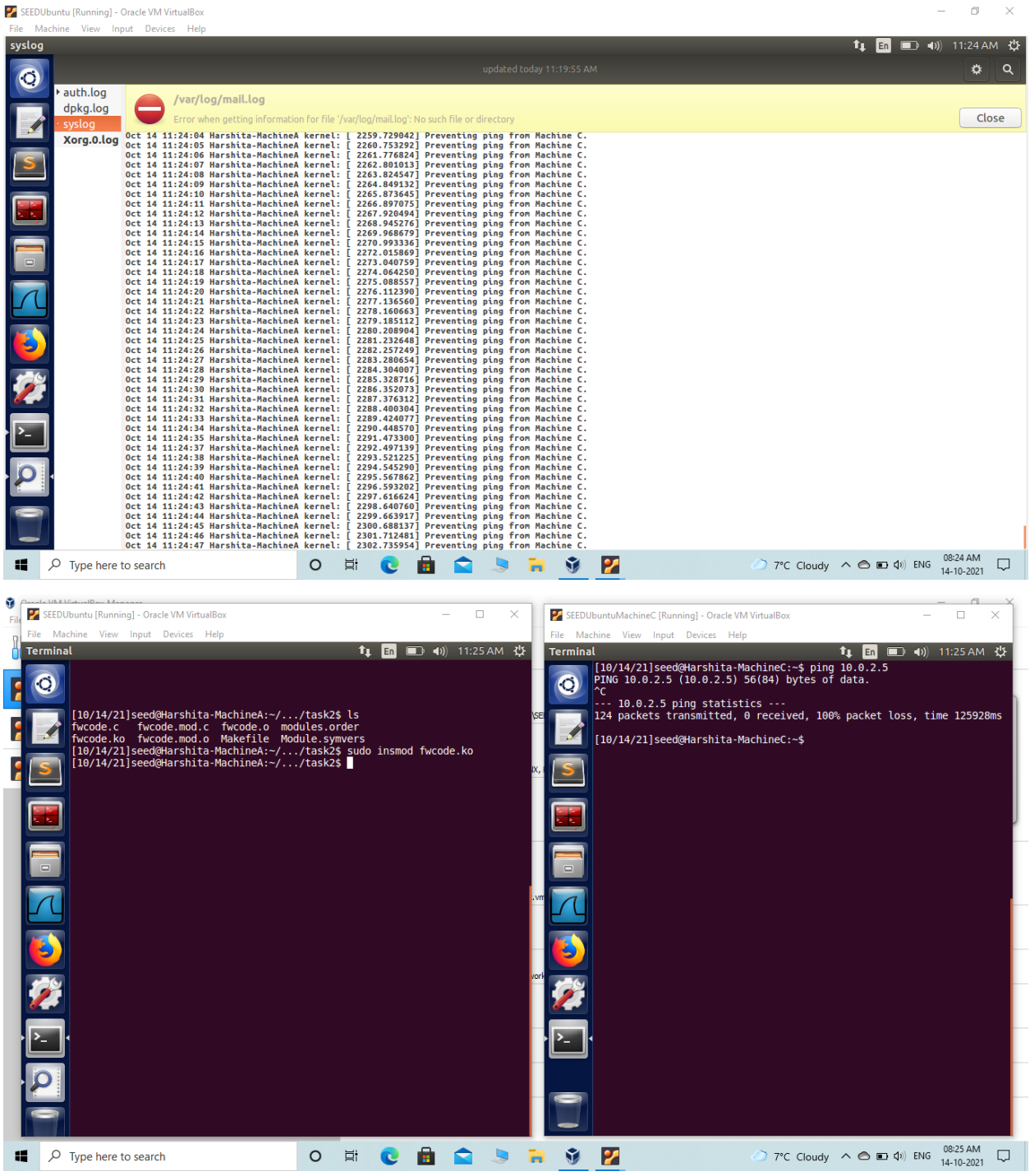
We see in the syslog that the filter is registered.



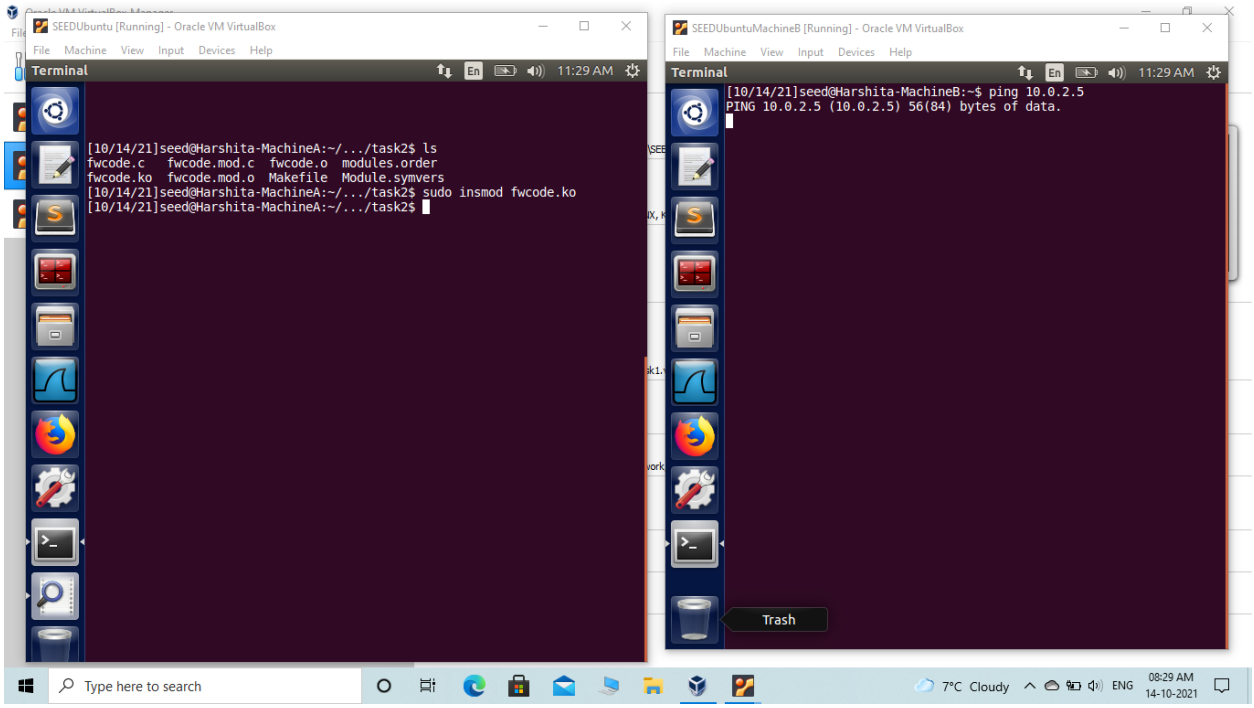
(a) Prevent Machine C to ping Machine A.



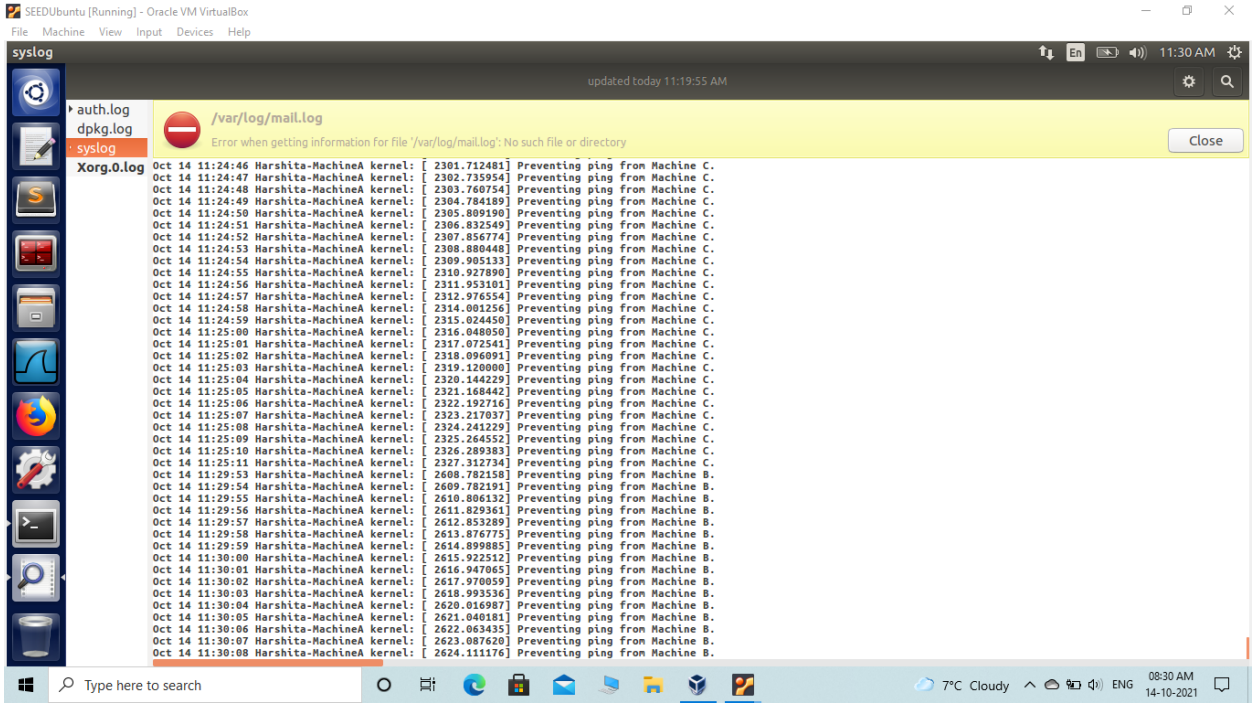
Messages in the syslog “Preventing ping from Machine C”.

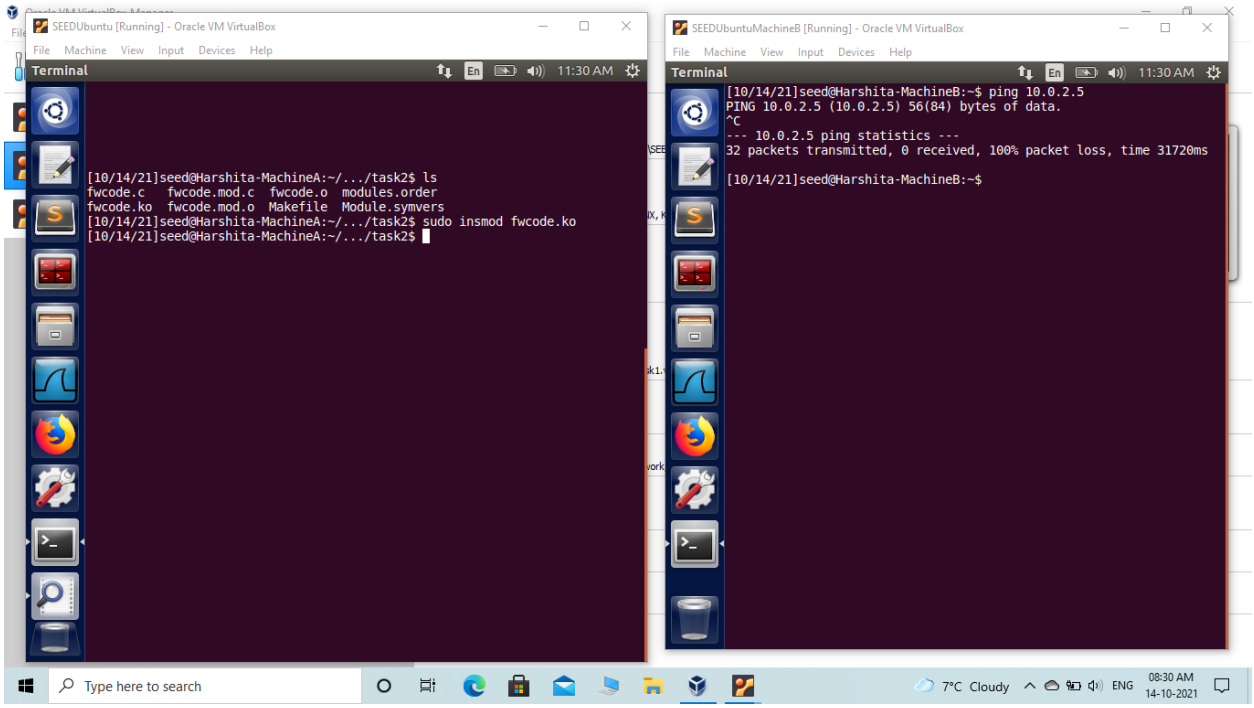


(b) Preventing Machine B from pinging Machine A.

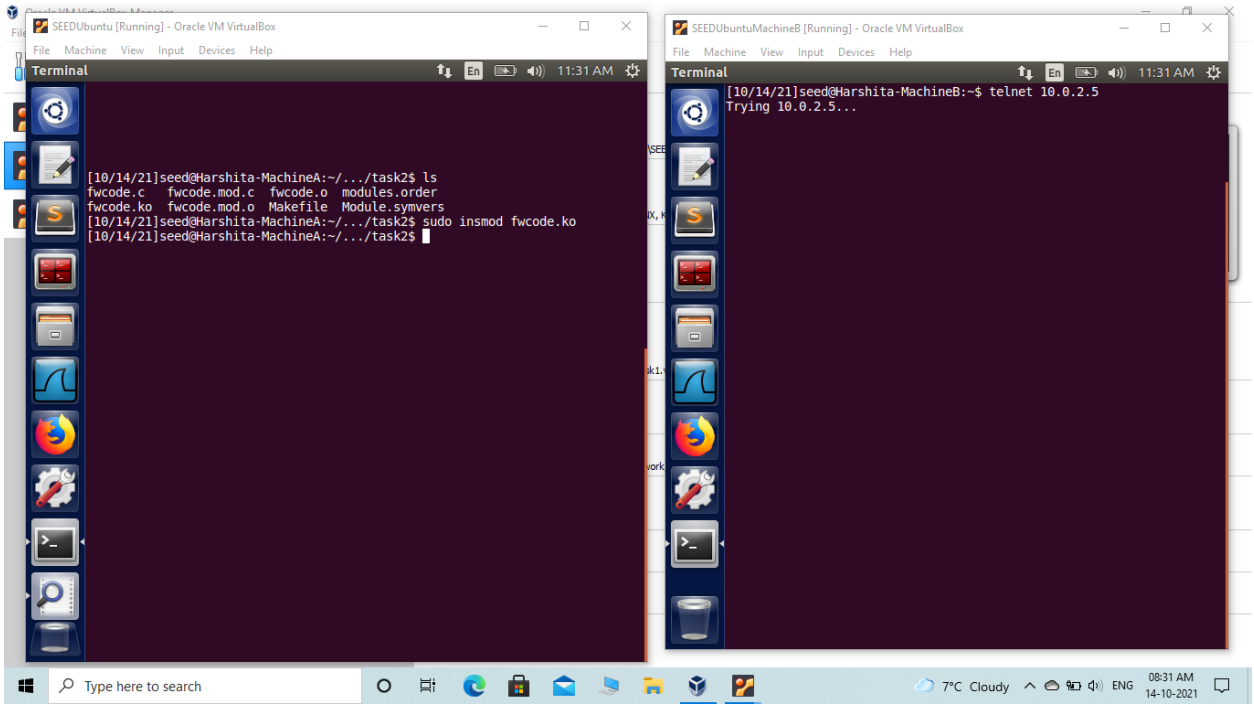


Messages in the syslog : “Preventing ping from Machine C”.

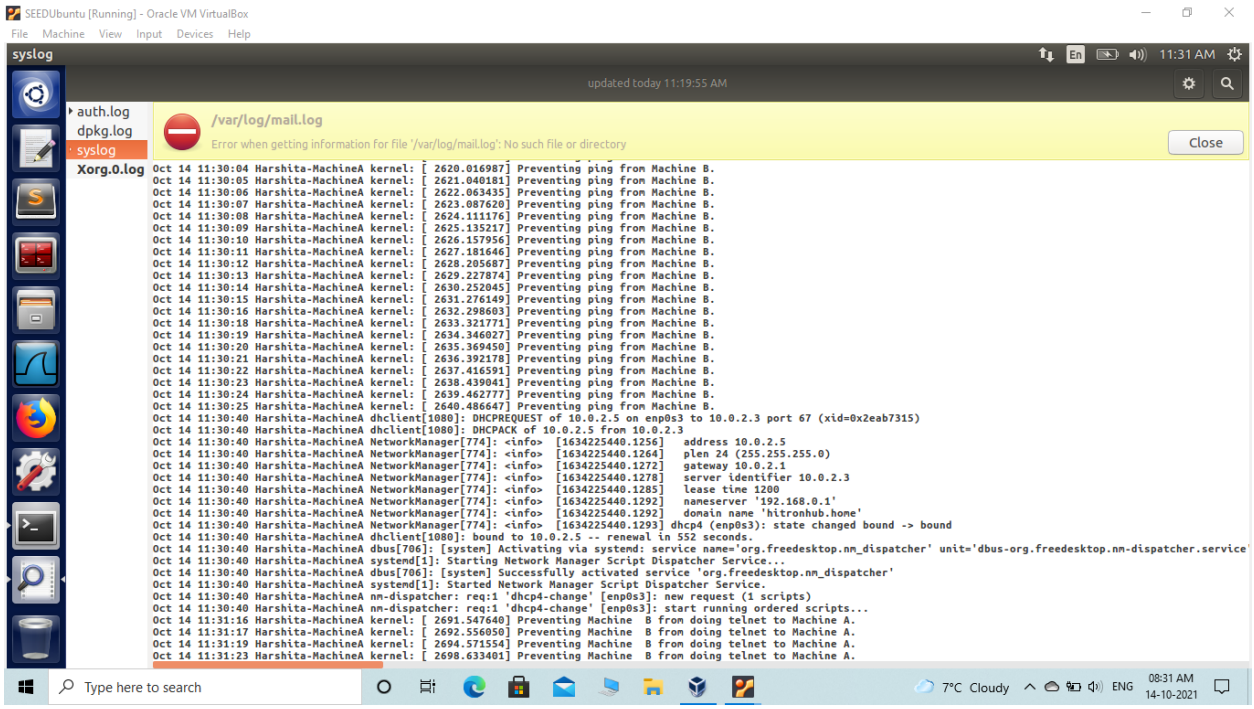


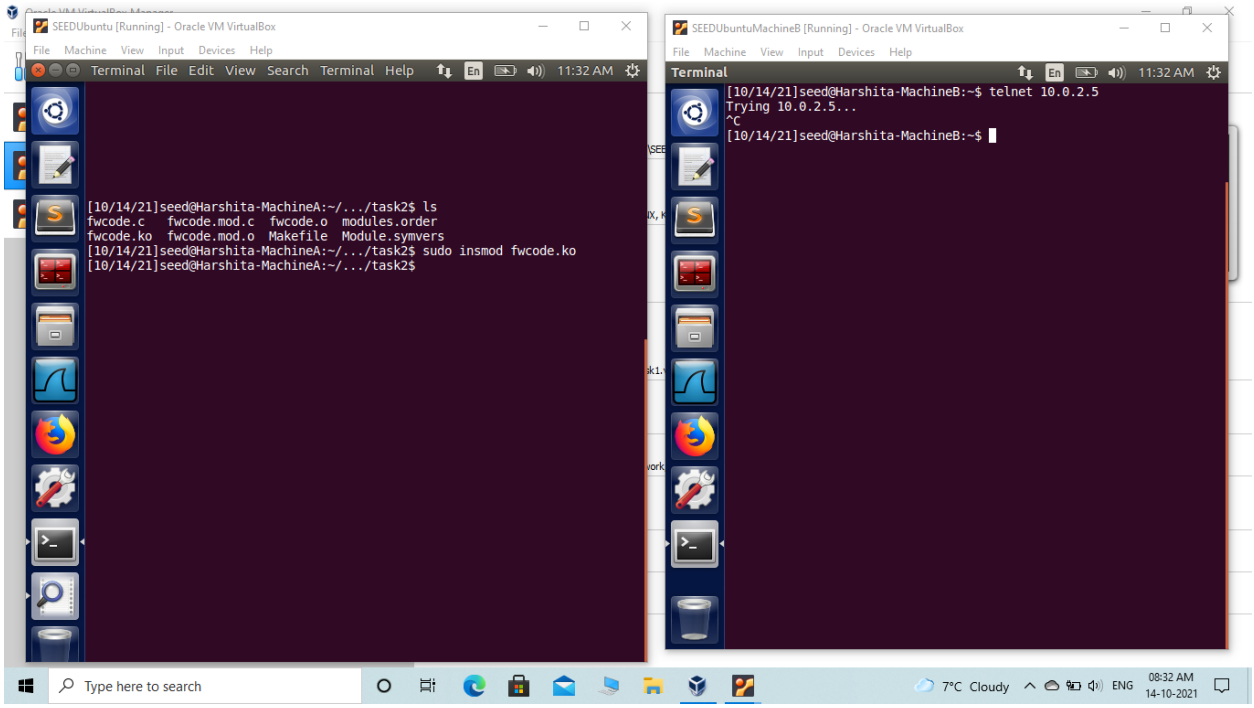


(c) Preventing telnet from Machine B to Machine A.

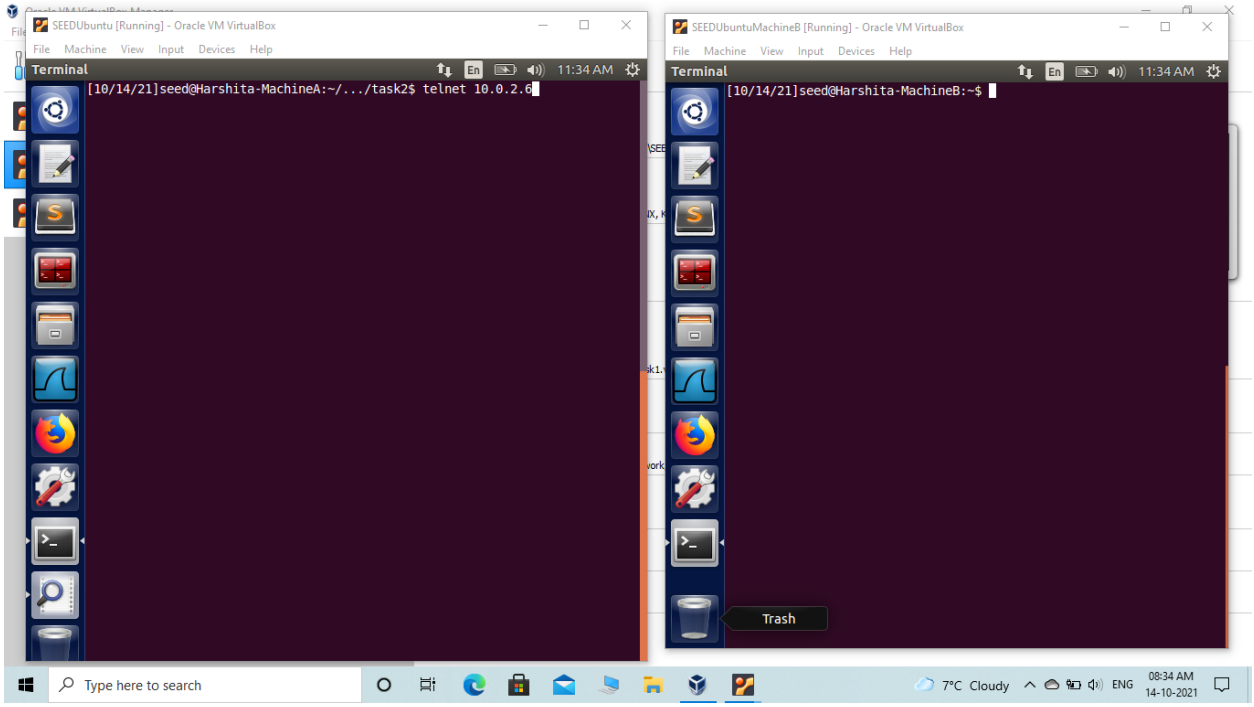


Messages in syslog: “Preventing Machine B from doing telnet to Machine A”.

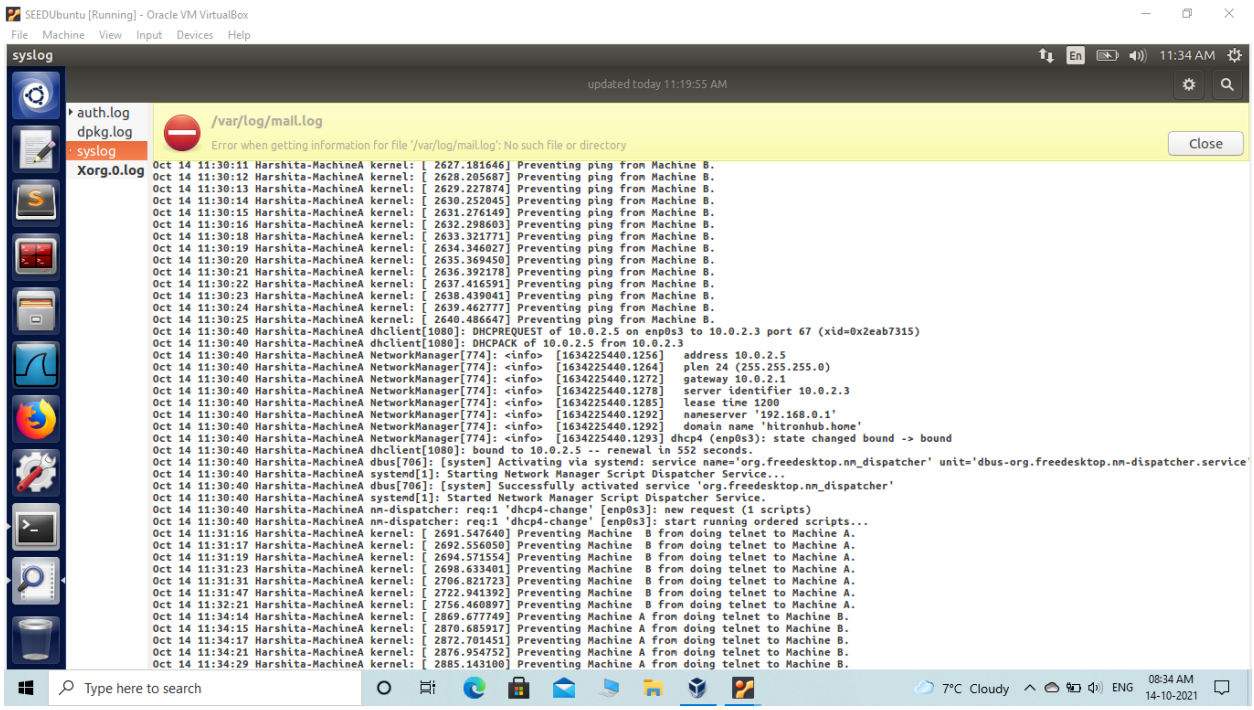


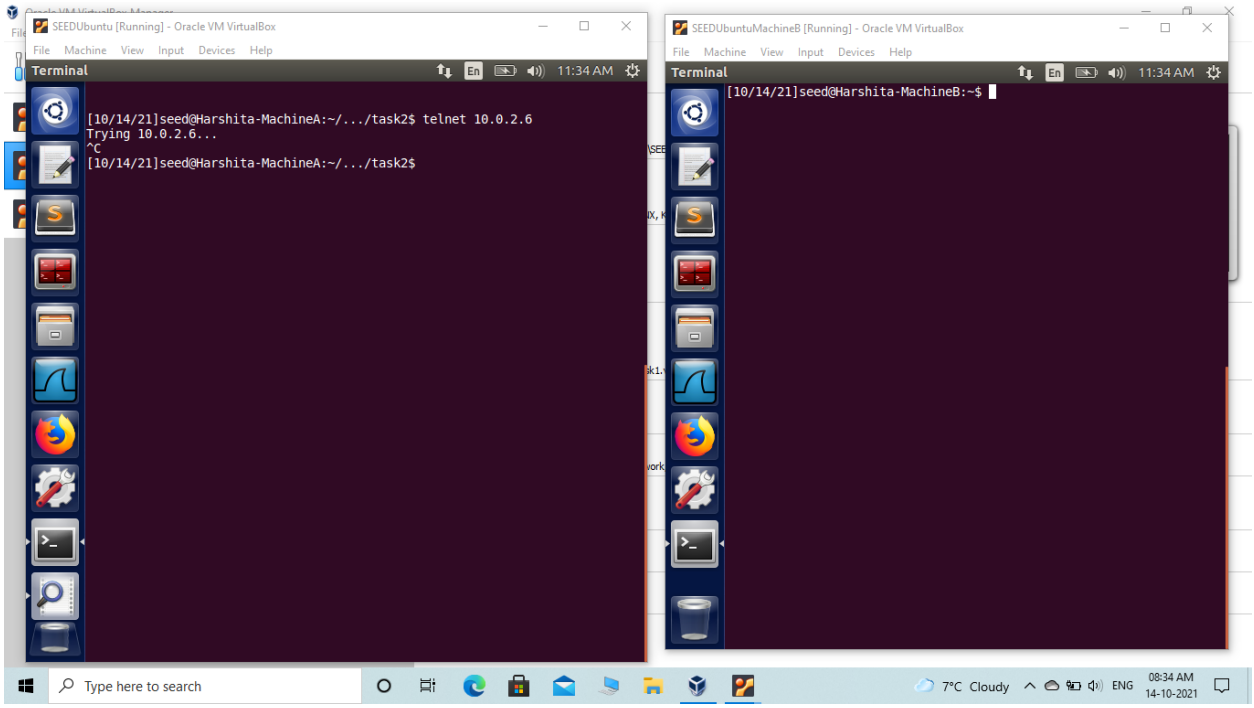


(d) Prevent machine A from doing telnet to Machine B

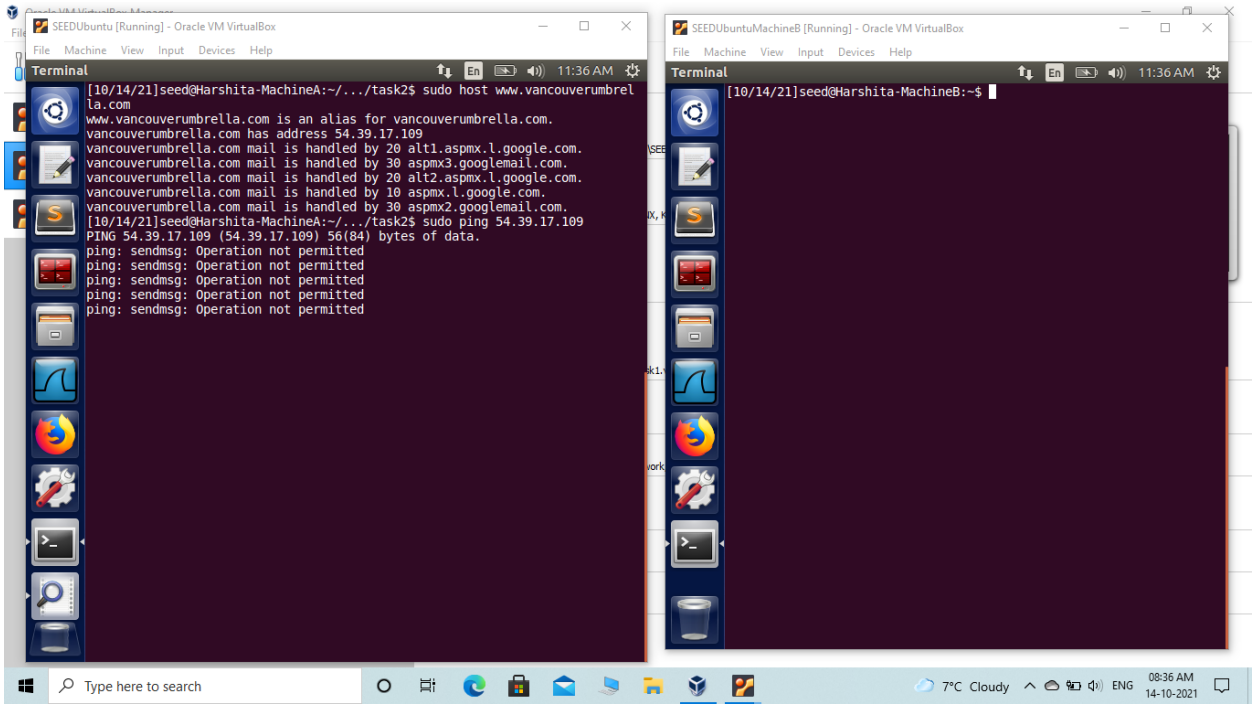


Messages in syslog: “Preventing machine A from doing telnet to Machine B”

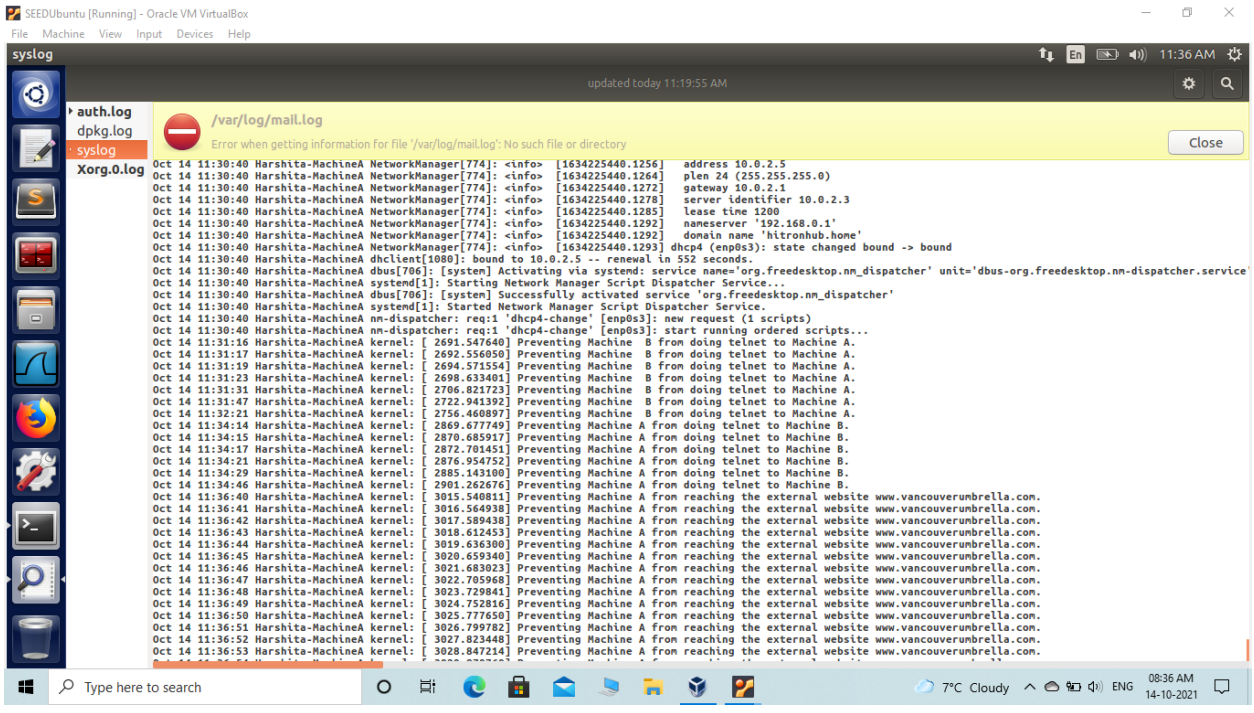


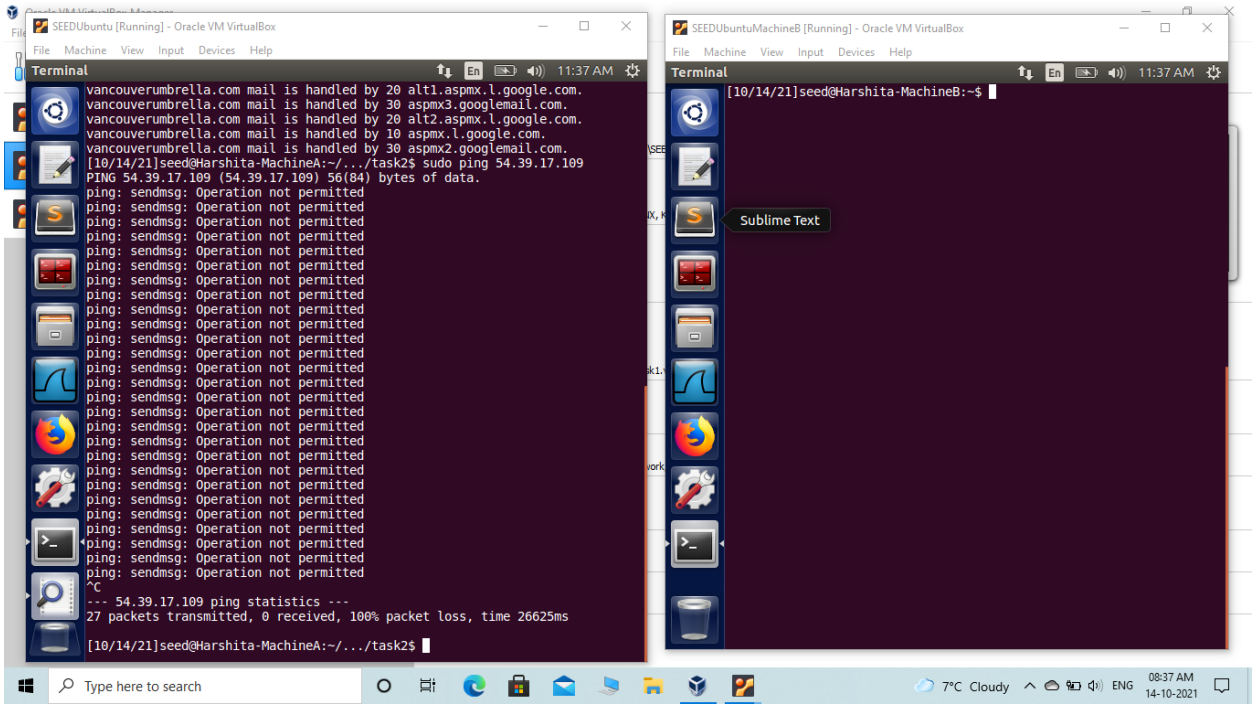


(e) Prevent Machine A from accessing an external website "www.vancouverumbrella.com"

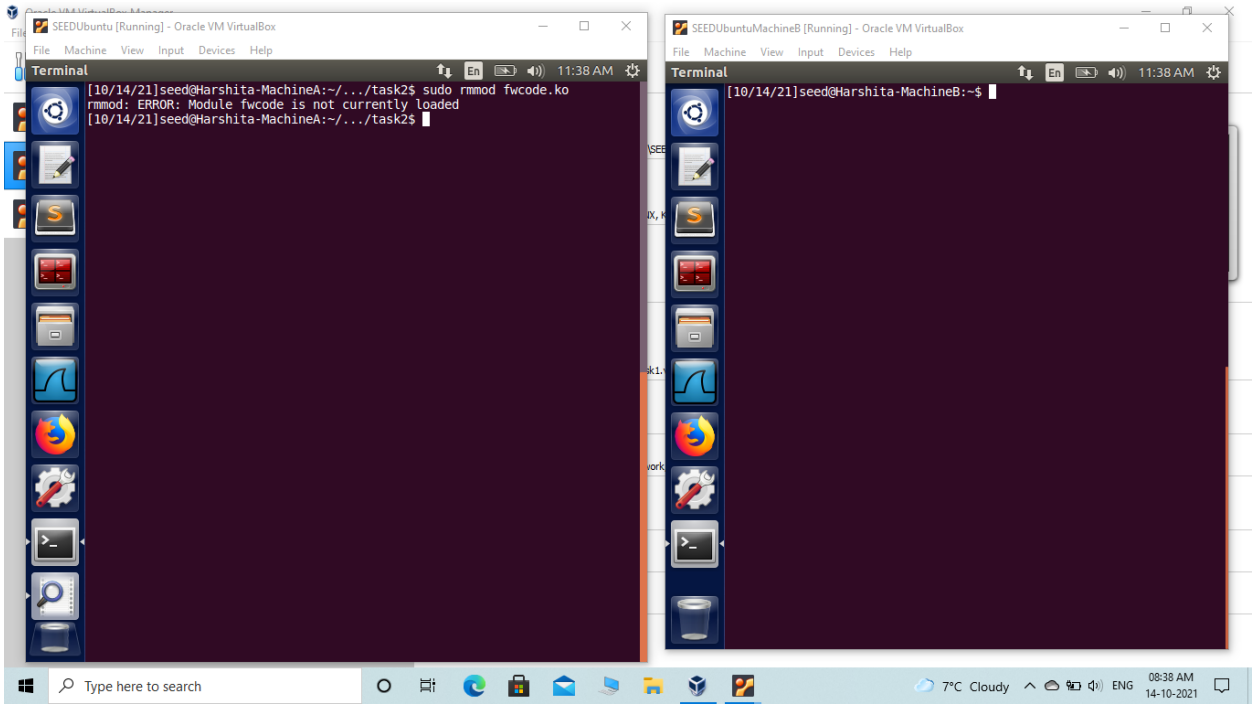


Messages in the syslog: “Preventing machine A from telneting external website”



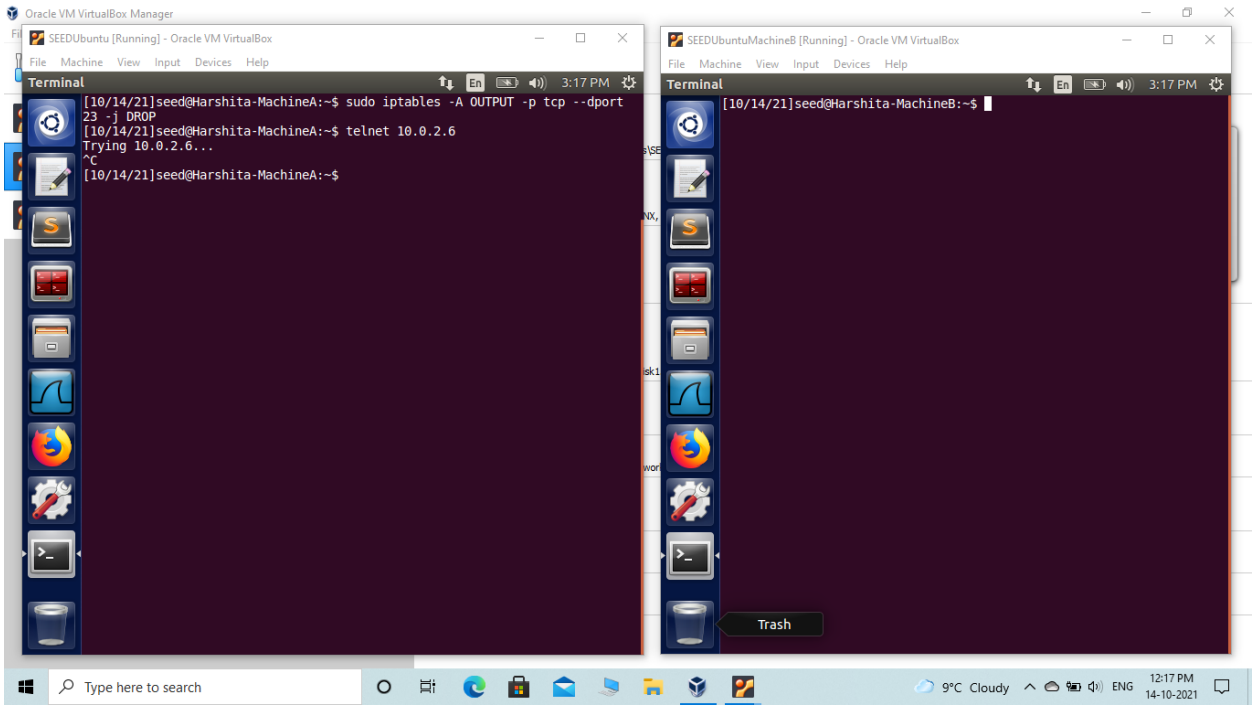


Now, removing the module using “rmmod” command.

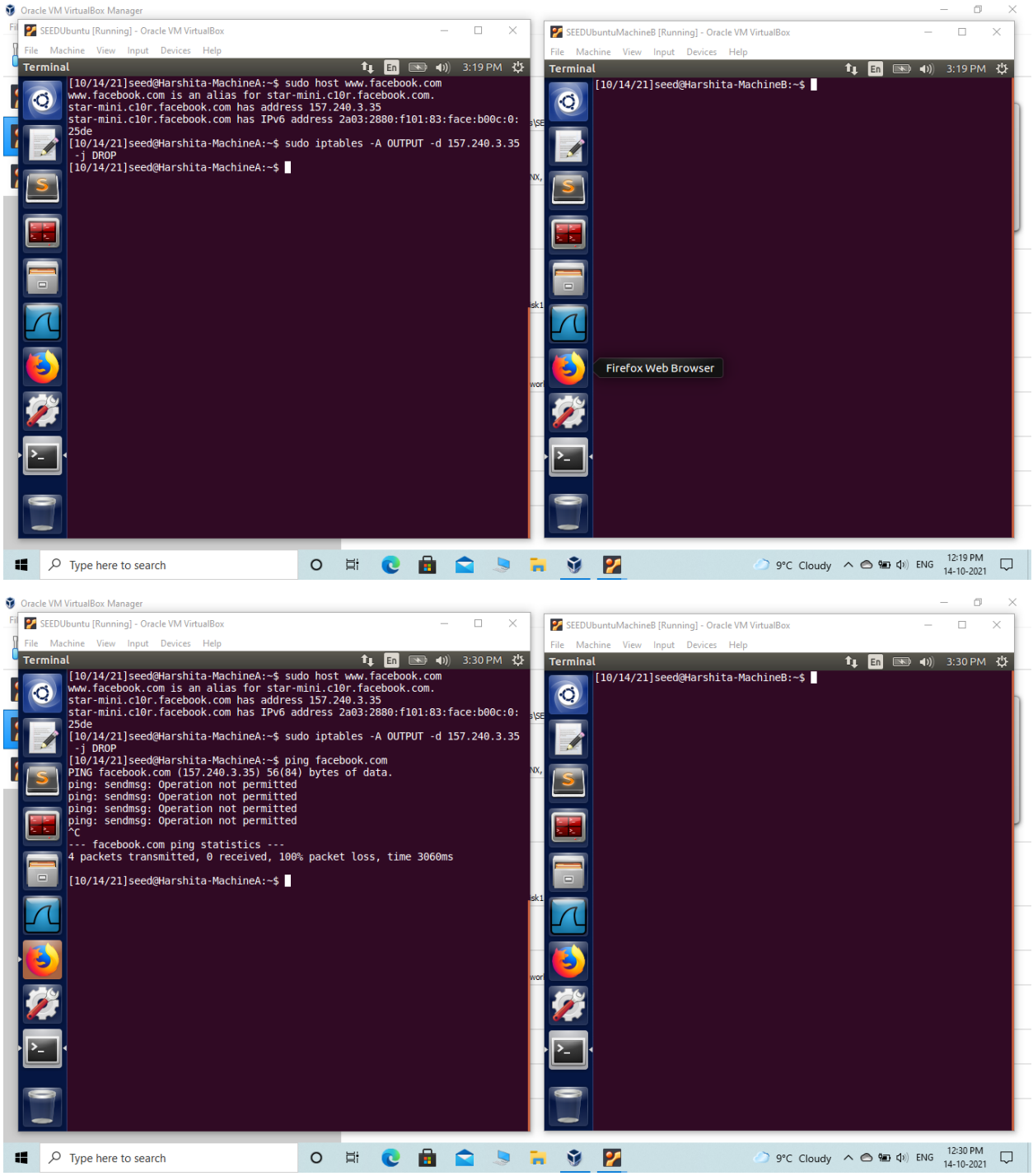


Task 3: Evading Egress Filtering

- **Block all the outgoing traffic to external telnet servers.**

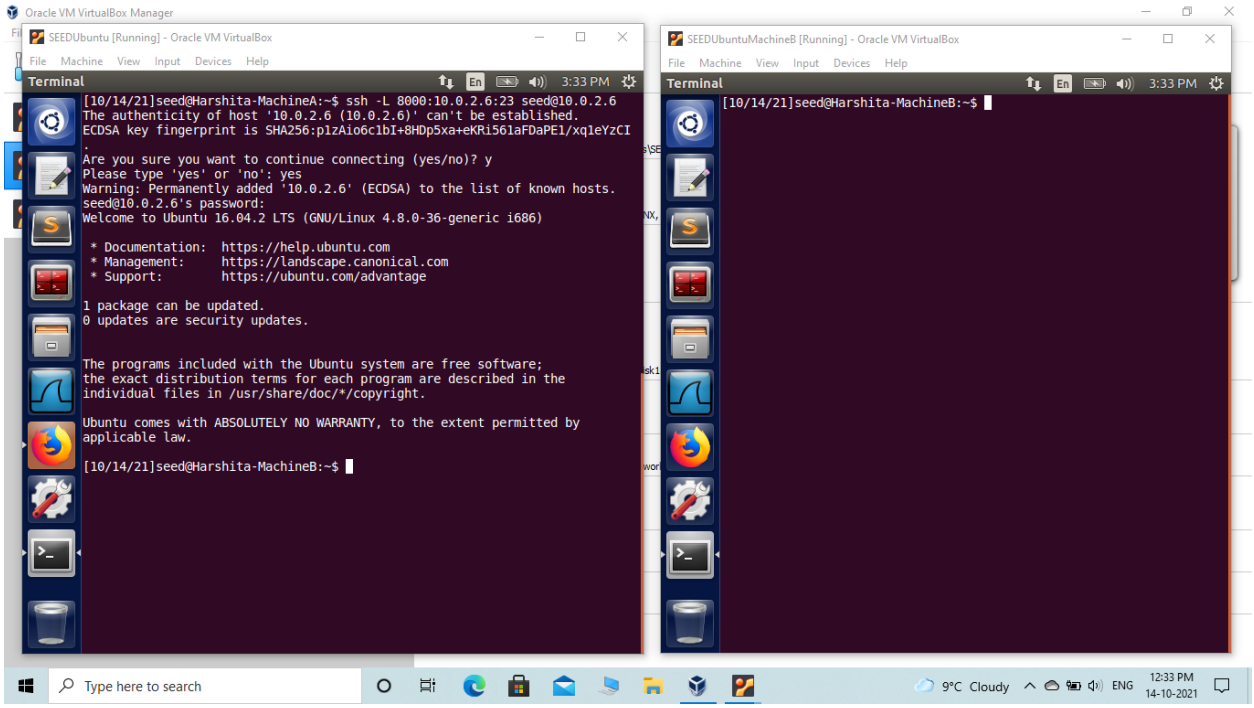


- **Block all the outgoing traffic to www.facebook.com**

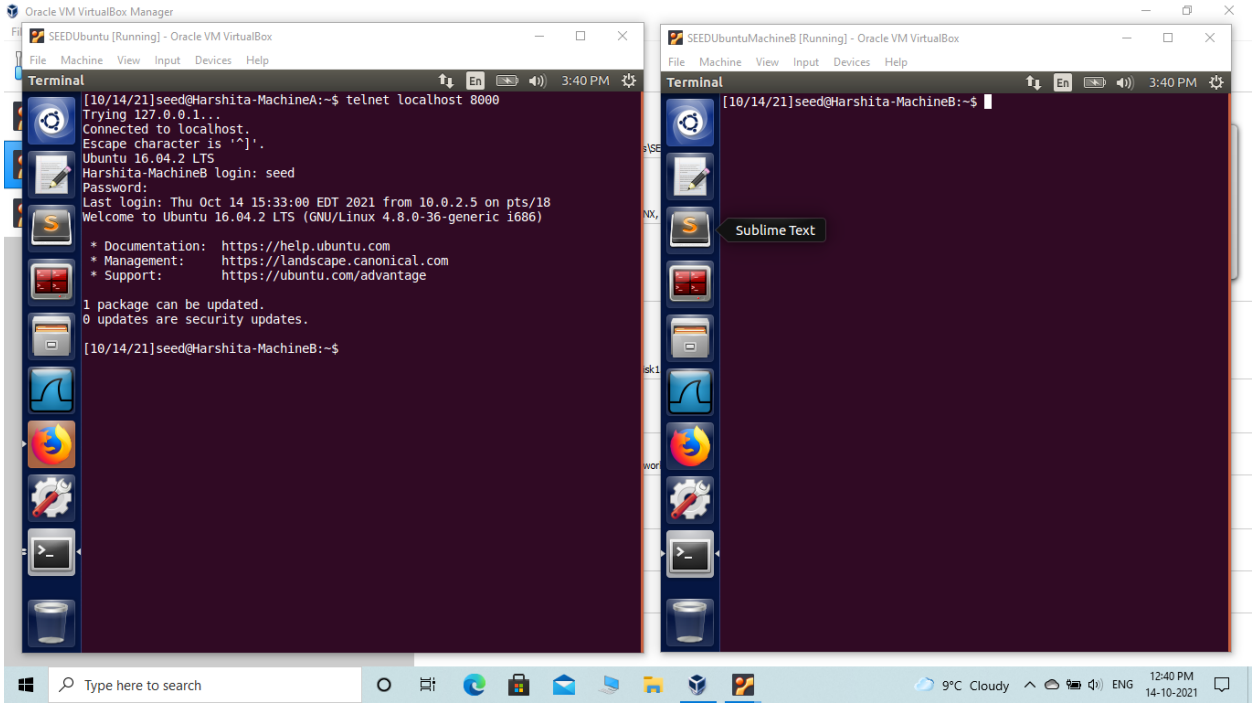


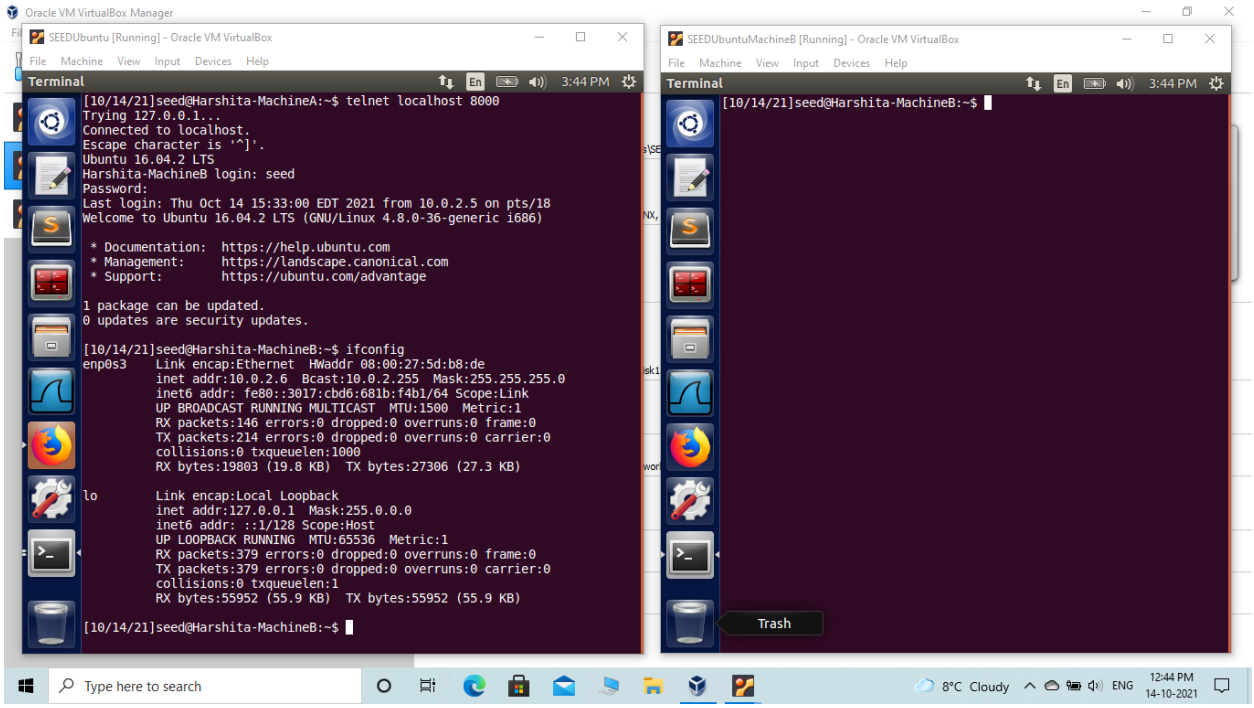
(a) Telnet to Machine B through the firewall

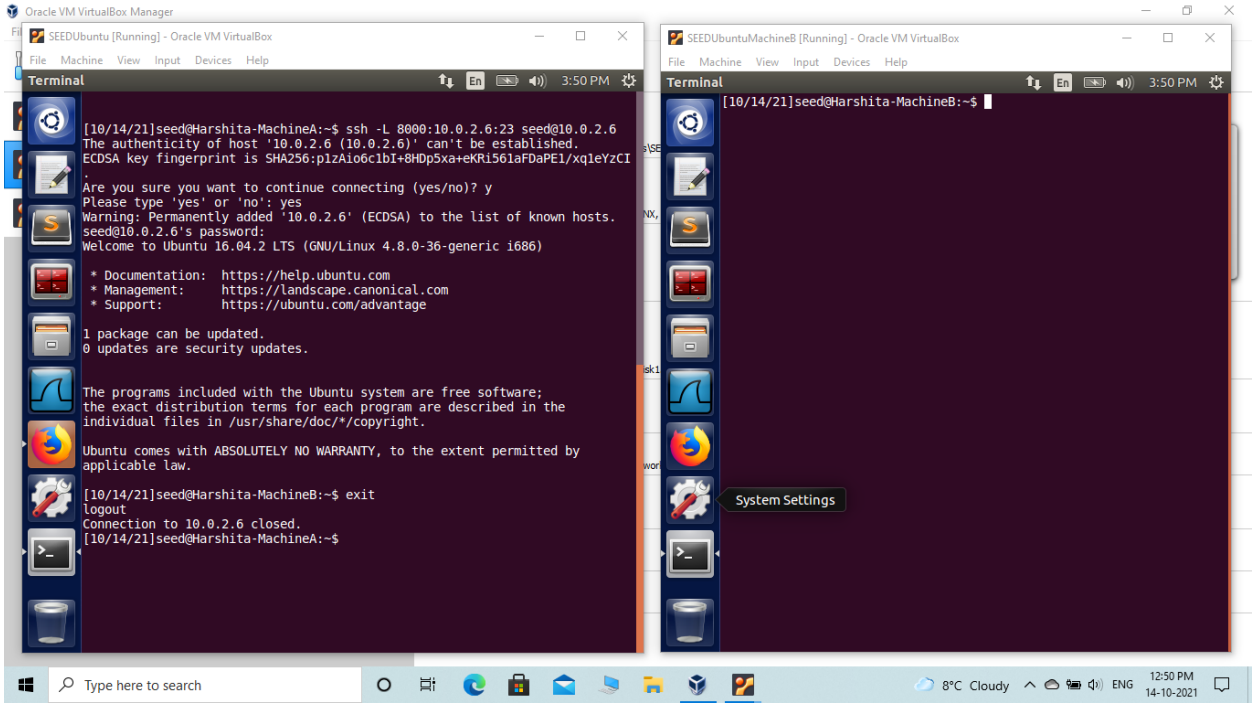
Now, we are connecting to machine B.



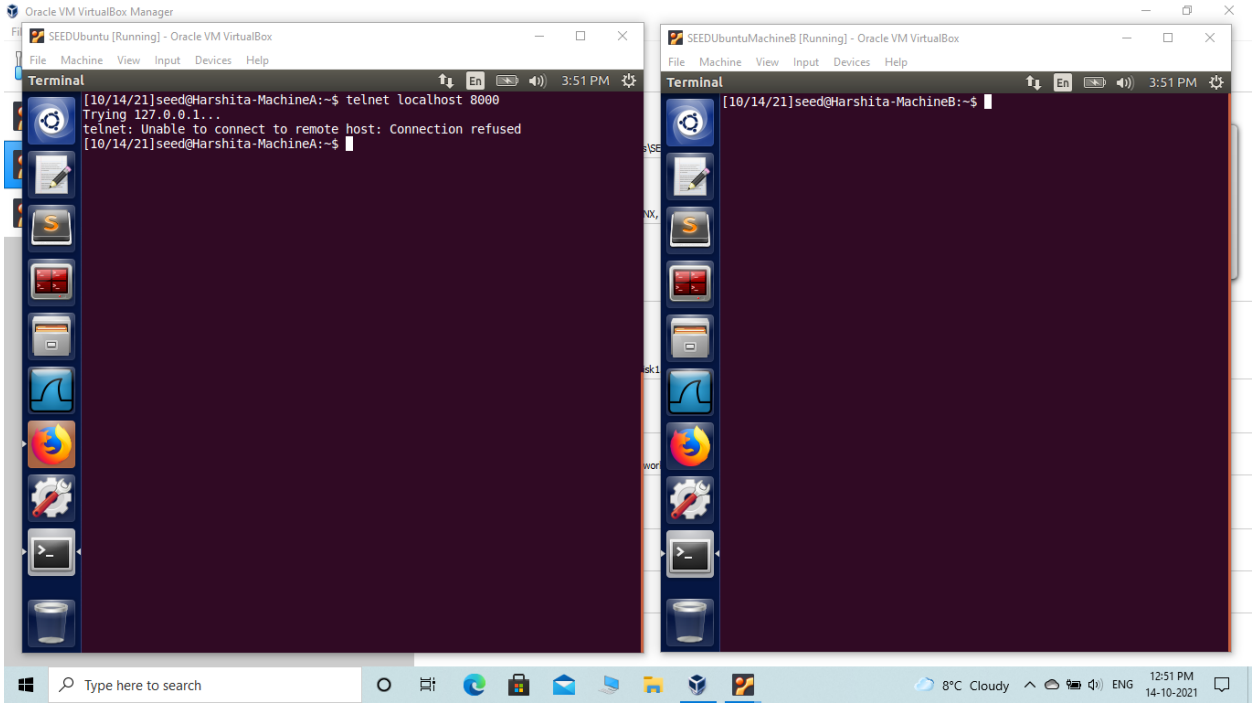
Now, we can telnet Machine B.





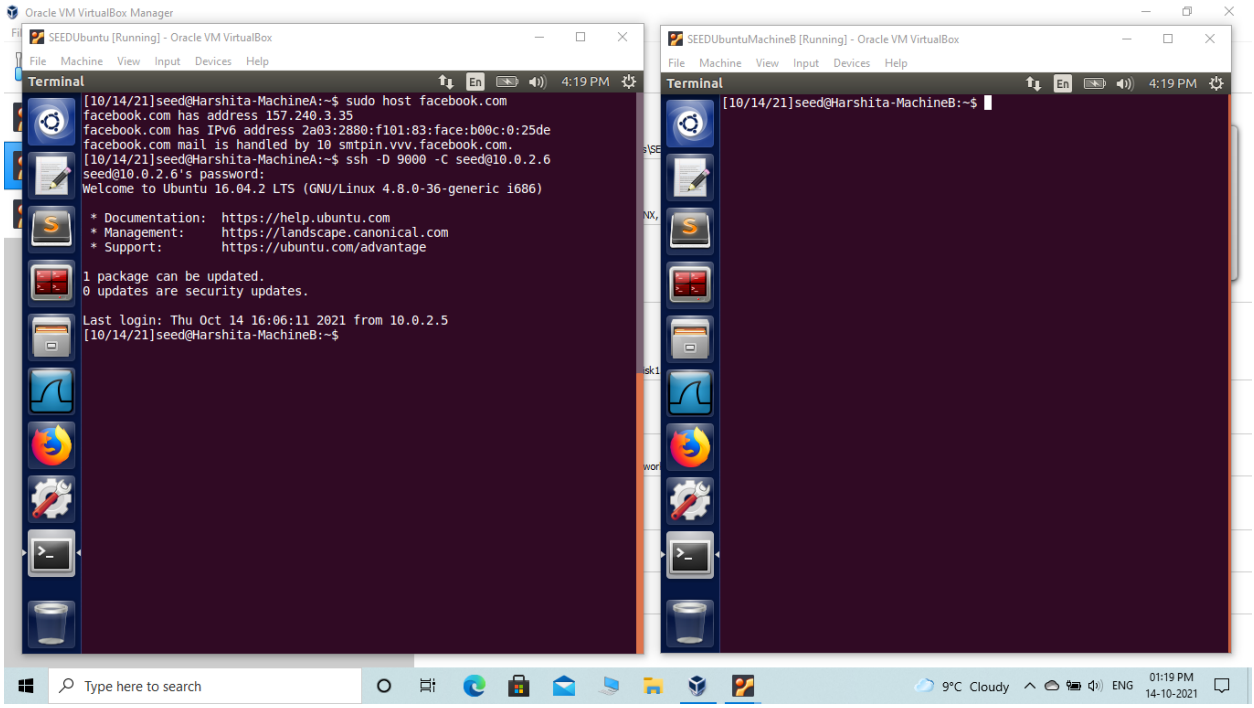


Now, We are again trying to connect to machine B.
But we are unable to connect because we have closed the connection.

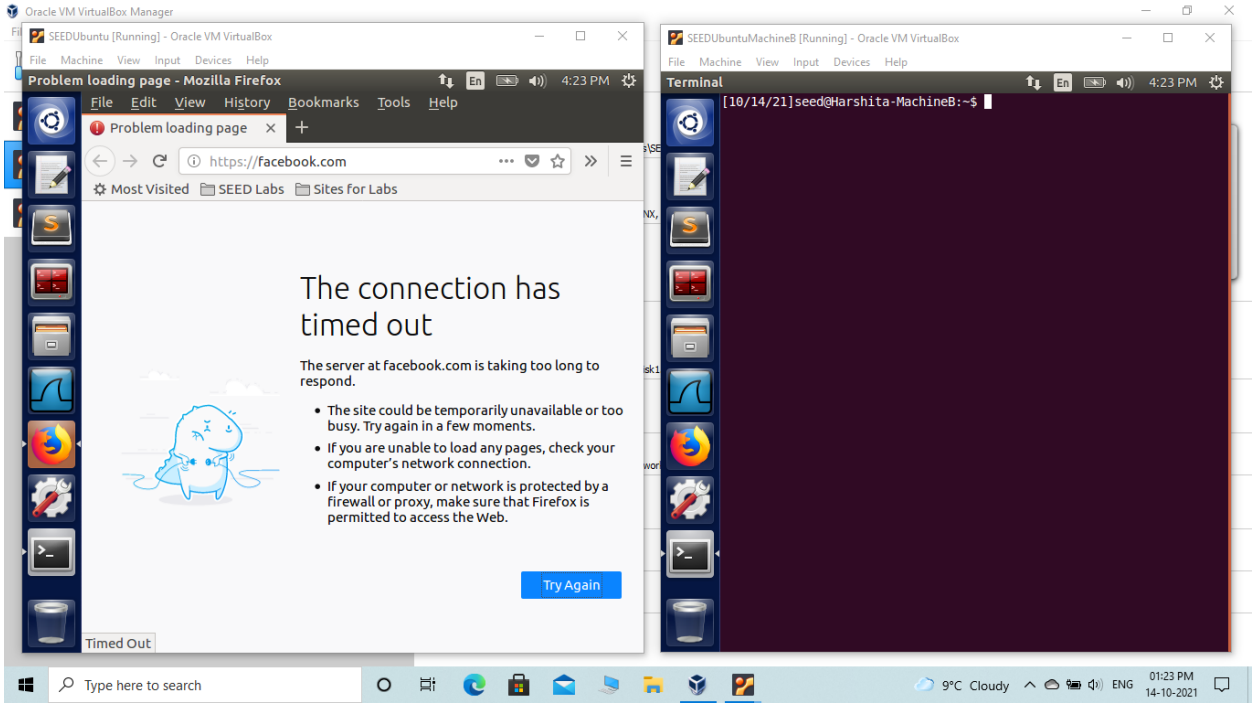


(b) Connect to Facebook using SSH Tunnel

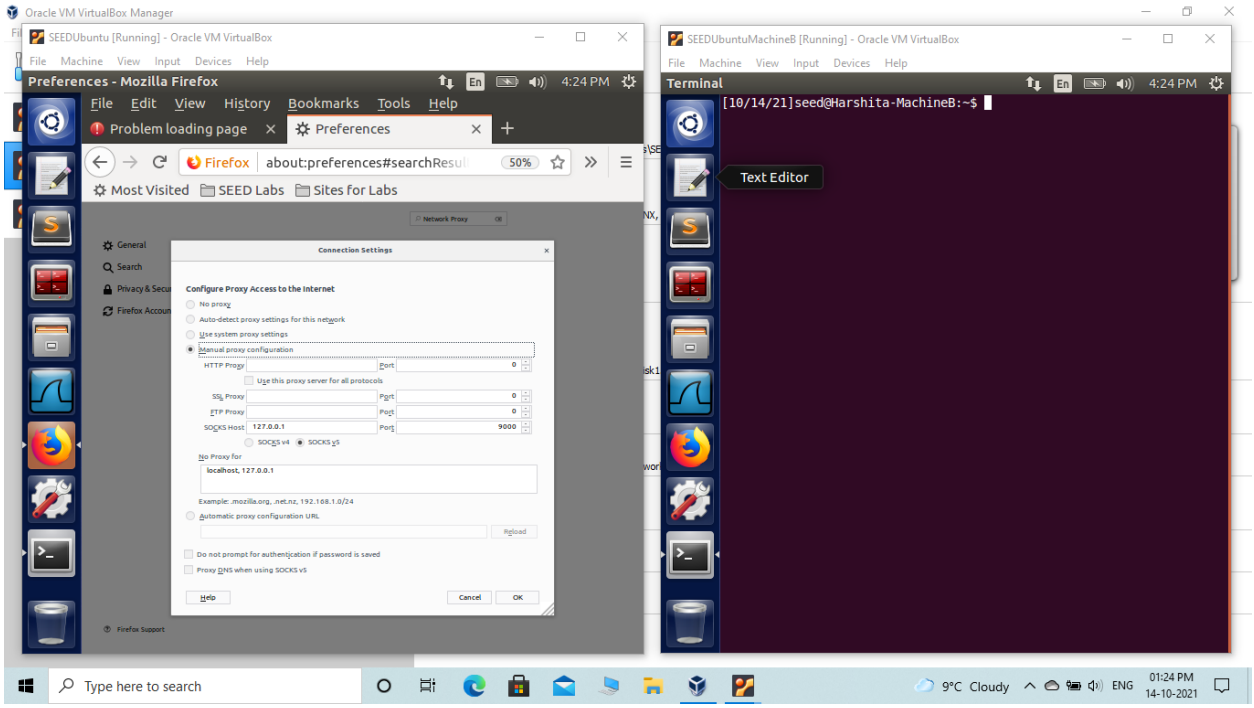
Now, we are connecting to Machine B.



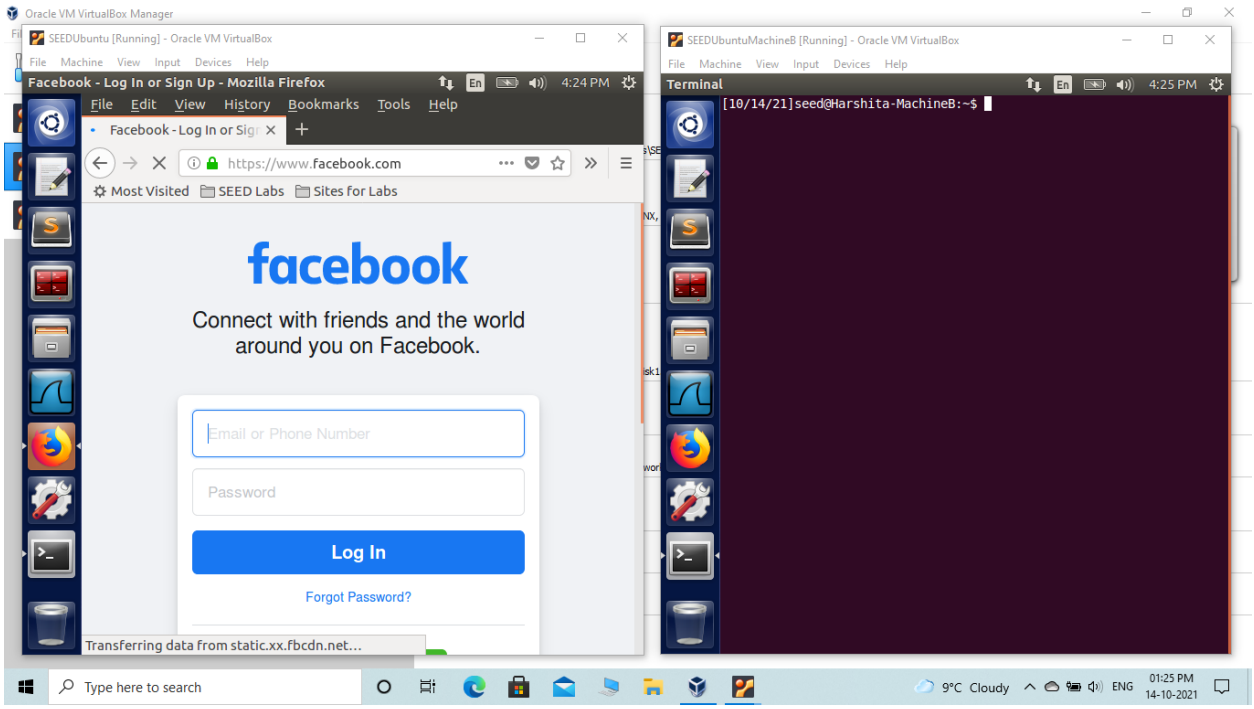
When we tried to access facebook on firefox, we are unable to access it.



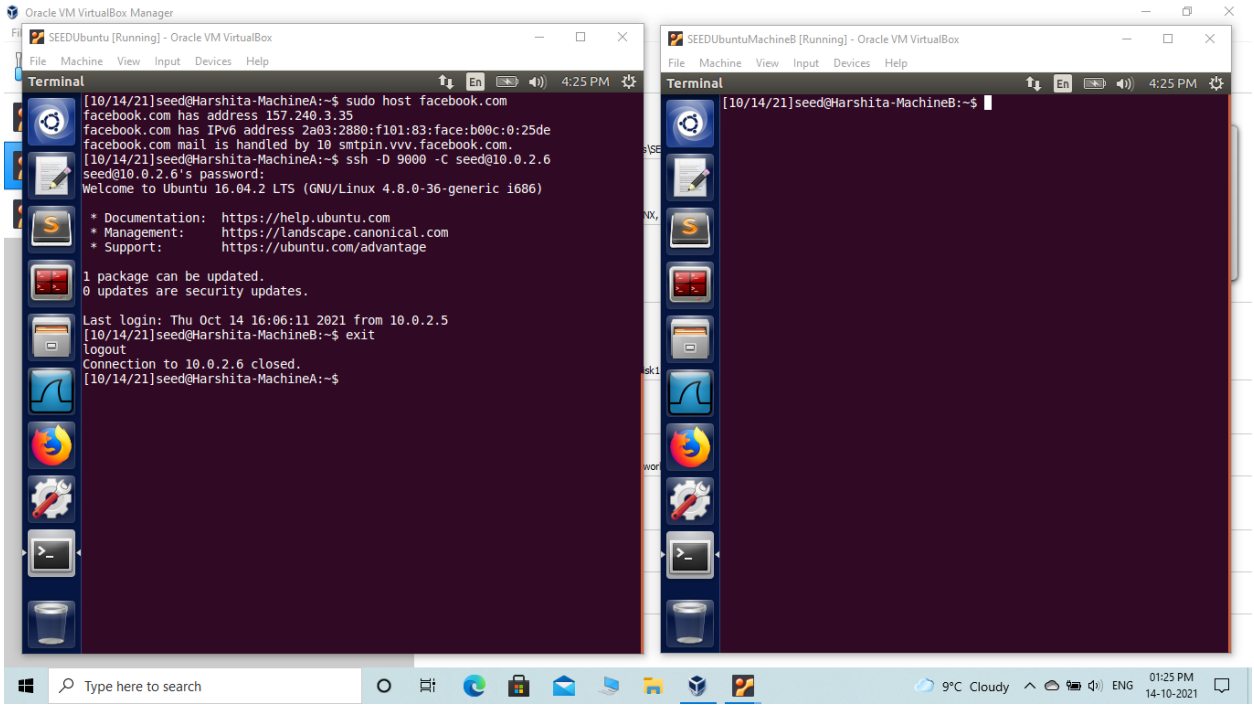
Now, we change the network proxy settings.



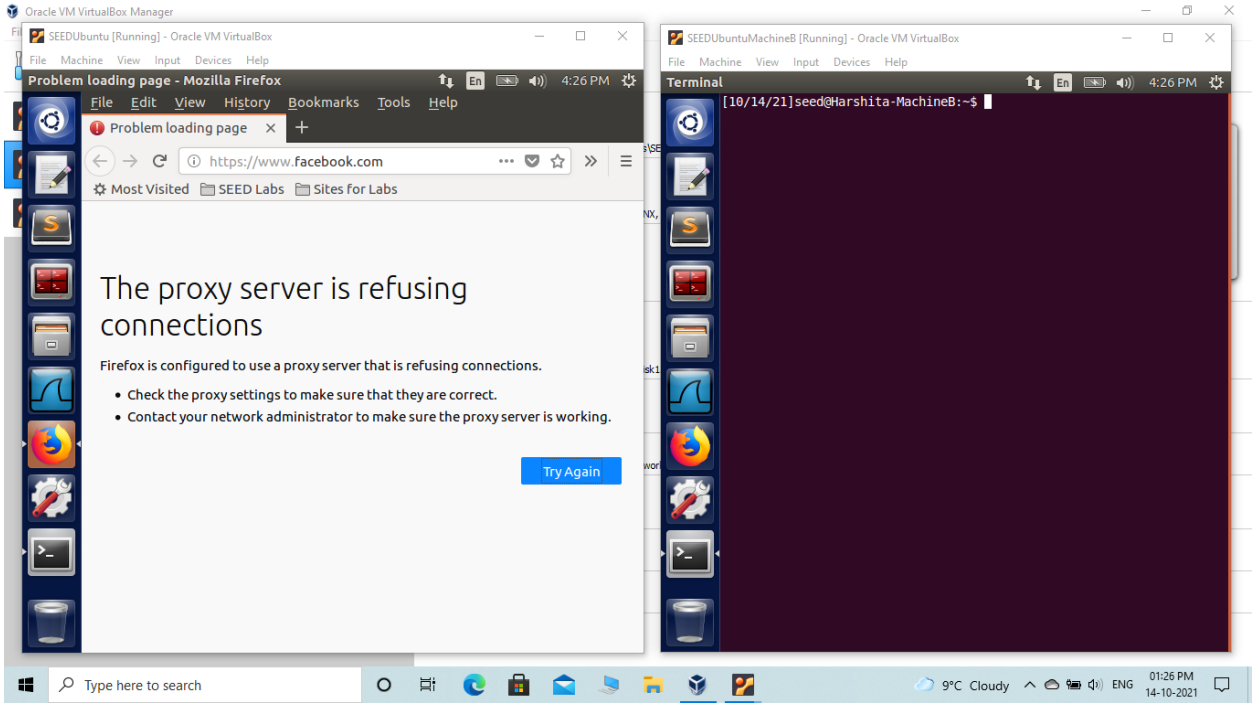
Now, we are able to access the facebook.



We exit and close the connection.

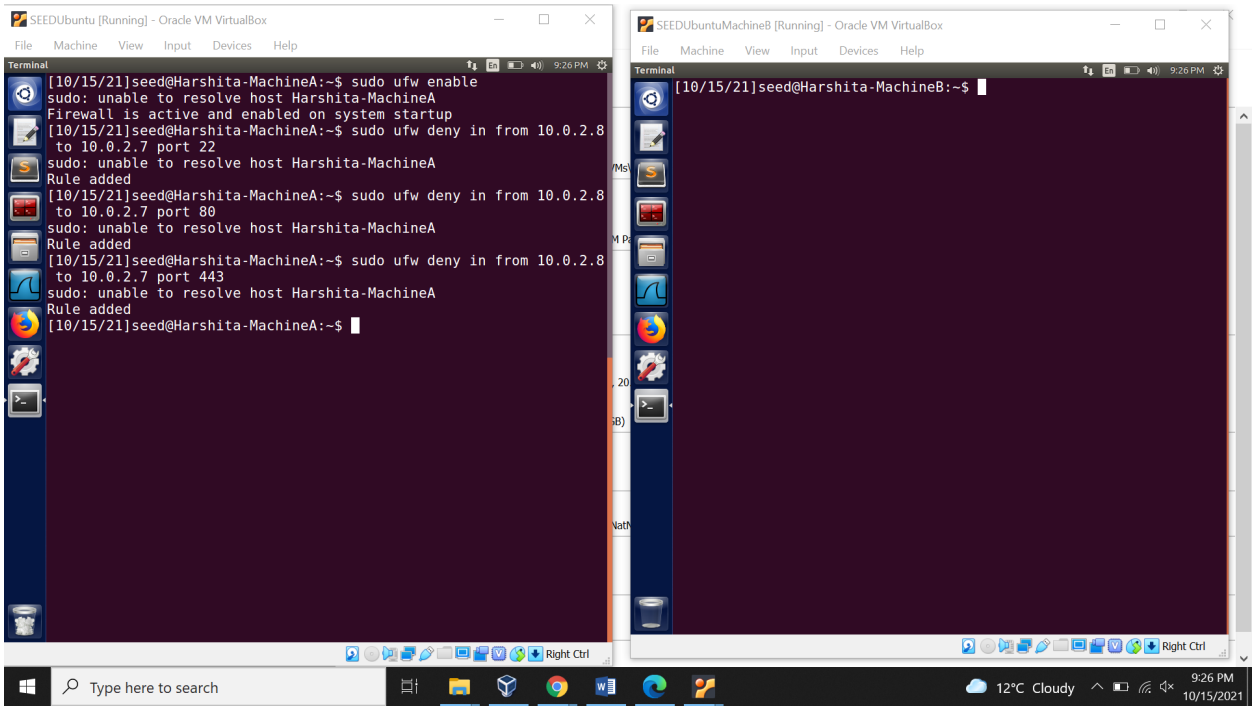


Again, we tried to access the facebook, but we are unable to access it again.

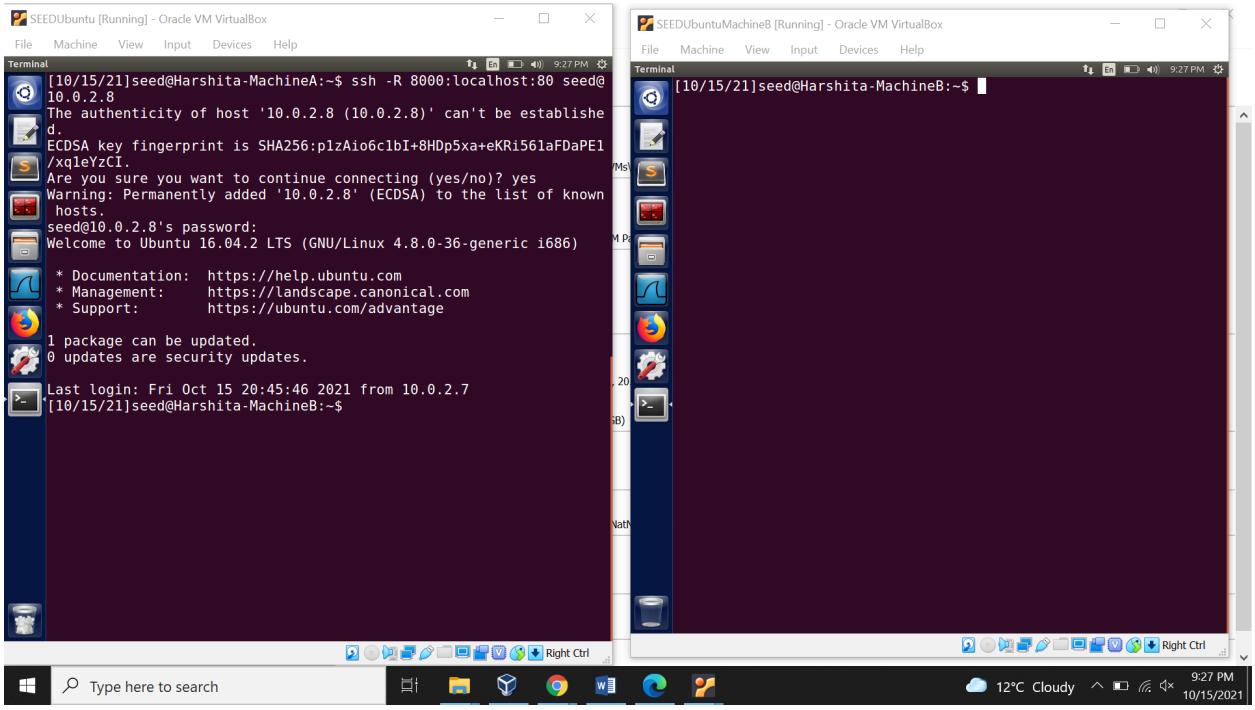


Task 4: Evading Ingress Filtering

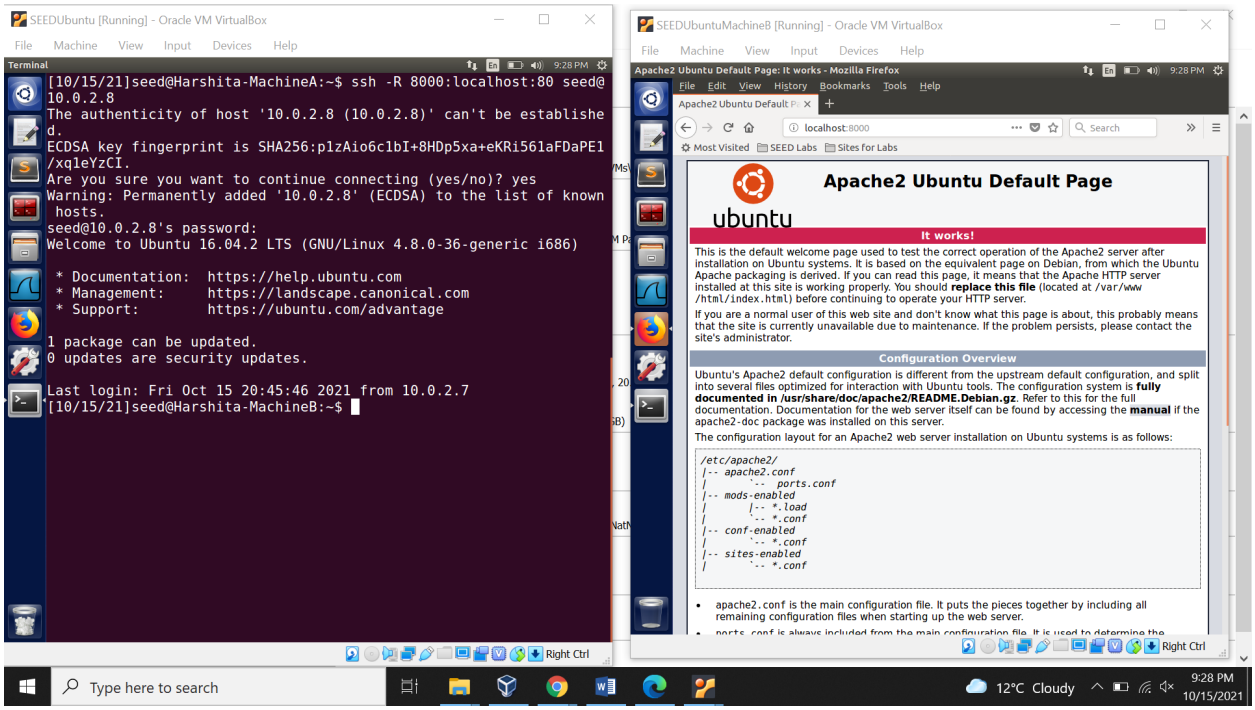
We block Machine B from accessing its port 80 (web server), 22 (SSH server) and port 443.



Set up a reverse SSH tunnel on Machine A.



Accessing the web server on Machine A from Machine B.



Appendix:

Code for the file fwcode.c

```
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/inet.h>
#include <linux/tcp.h>
#include <linux/ip.h>

static struct nf_hook_ops InfilterHook;
static struct nf_hook_ops OutfilterHook;

unsigned int filter_func(void *priv, struct sk_buff *skb,
const struct nf_hook_state *state)
{
        struct iphdr *iph;
        struct tcphdr *tcph;
        iph = ip_hdr(skb);
```

```
        tcph = (void*)iph + iph -> ihl*4;
        if(iph -> protocol == IPPROTO_TCP && tcph -> dest == htons(23) && iph -> saddr ==
in_aton("10.0.2.6"))
        {
                printk("Preventing Machine  B from doing telnet to Machine A.\n");
                return NF_DROP;
        }
        else if(iph -> protocol == IPPROTO_ICMP  && iph -> saddr == in_aton("10.0.2.6"))
        {
                printk("Preventing ping from Machine B.\n");
                return NF_DROP;
        }
        else if(iph -> protocol == IPPROTO_ICMP && iph -> saddr == in_aton("10.0.2.7"))
        {
                printk("Preventing ping from Machine C.\n");
                return NF_DROP;
        }
        else
        {
                return NF_ACCEPT;
        }
}

unsigned int Outfilter_func(void *priv, struct sk_buff *skb,
const struct nf_hook_state *state)
{
      struct iphdr *iph;
      struct tcphdr *tcph;
      iph = ip_hdr(skb);
      tcph = (void*)iph + iph -> ihl*4;
         if(iph -> protocol == IPPROTO_TCP && tcph -> dest == htons(23) && iph -> saddr ==
in_aton("10.0.2.5"))
      {
           printk("Preventing Machine A from doing telnet to Machine B.\n");
           return NF_DROP;
      }
      else if(iph -> protocol == IPPROTO_ICMP && iph -> daddr == in_aton("54.39.17.109"))
      {
           printk("Preventing Machine A from reaching the external website
www.vancouverumbrella.com.\n");
           return NF_DROP;
      }
         else
```

```
        {
                return NF_ACCEPT;
        }
}

int SetUpFilter(void)
{
        printk(KERN_INFO "Register Filter\n");
        InfilterHook.hook = filter_func;
        InfilterHook.hooknum = NF_INET_PRE_ROUTING;
        InfilterHook.pf = PF_INET;
        InfilterHook.priority = NF_IP_PRI_FIRST;
        nf_register_net_hook(&init_net,&InfilterHook);
        OutfilterHook.hook = Outfilter_func;
    OutfilterHook.hooknum = NF_INET_POST_ROUTING;
    OutfilterHook.pf = PF_INET;
    OutfilterHook.priority = NF_IP_PRI_FIRST;
        nf_register_net_hook(&init_net,&OutfilterHook);
        return 0;
}

void RemoveFilter(void)
{
        printk(KERN_INFO "Filter is removed.\n");
        nf_unregister_net_hook(&init_net,&InfilterHook);
        nf_unregister_net_hook(&init_net,&OutfilterHook);
}

module_init(SetUpFilter);
module_exit(RemoveFilter);
```

Code for Makefile:

```
obj-m += fwcode.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
```