Team 2

Adeel Qureshi
Patrick Poplawska
Edward Tischler
Christine King
Troy Gittelmacher

Team Member Contribution Breakdown
Github: https://github.com/ppoplawska/EnterpriseDefenseFirewall

Adeel Qureshi	Task 1, Task 2, Task 3
Patrick Poplawska	Task 1, Task 2, Task 3
Edward Tischler	Task 2, Task 4
Christine King	Task 1,Task 4a, 4b
Troy Gittelmacher	Task 1, Task 2

Citations:

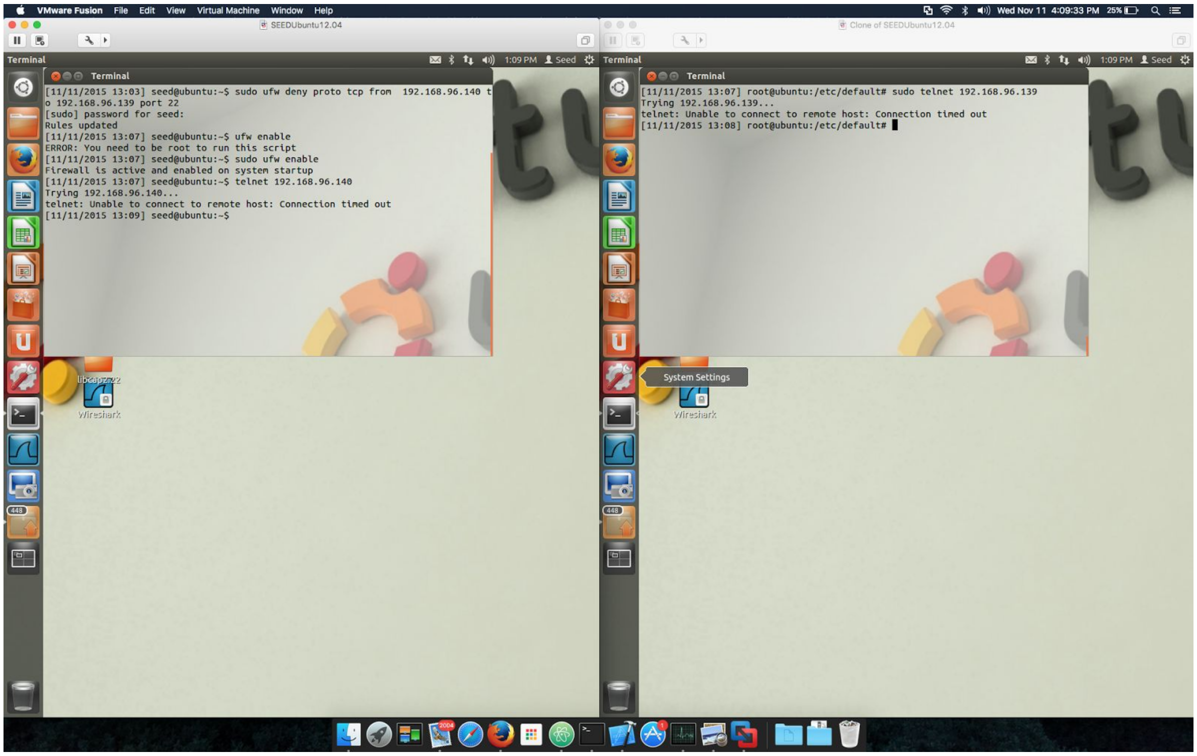
http://askubuntu.com/questions/532305/using-ufw-to-block-outgoing-traffic-to-website

https://workaround.org/squid-acls/ (task 4)

Task 1: Using the Linux Firewall (ufw / iptables)

The first task involves using the linux program ufw, which is a front facing application for iptables functionality in the linux kernel. This allows users to use simple rules to create fairly powerful restrictions.

The first rule we implemented (actually a set of two) was to prevent telnet connections between the two machines. The screenshot provided demonstrates this restriction in action. Machine A (left) cannot telnet into Machine B (right) and the converse is also true.

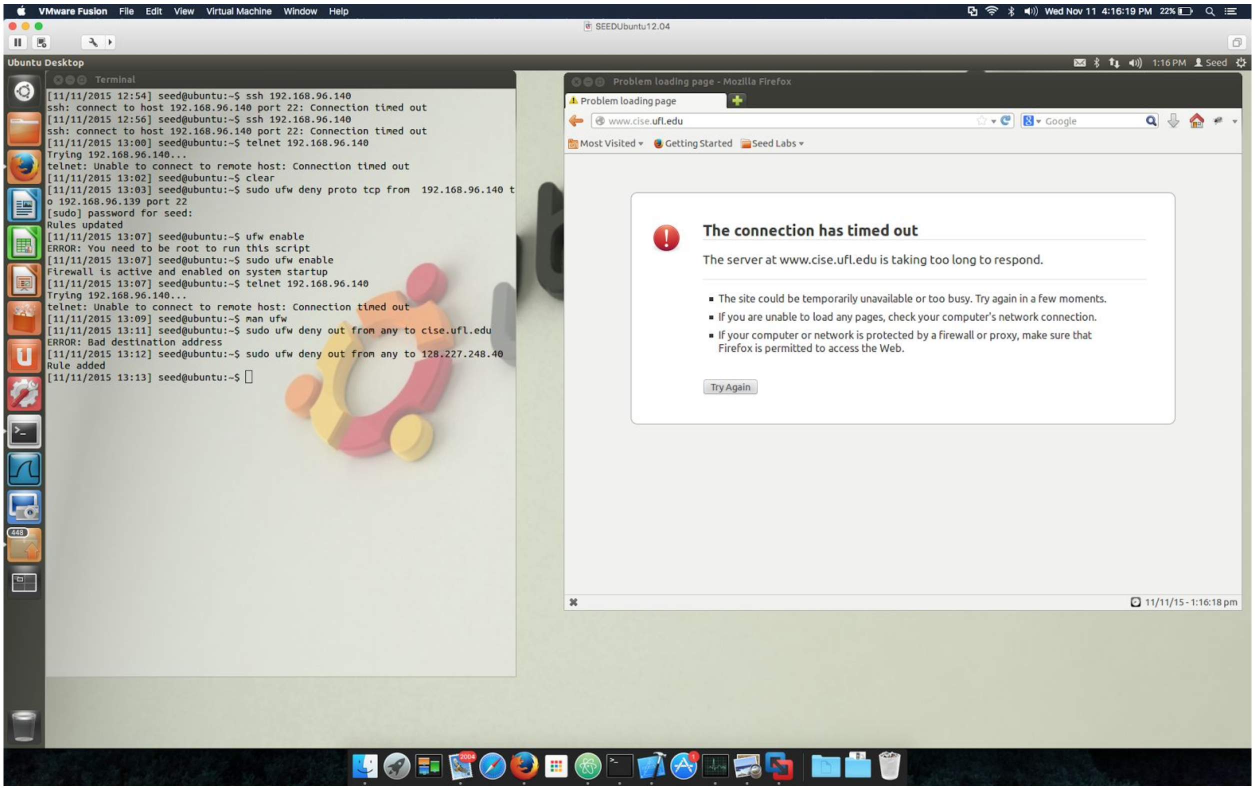


```
[11/11/2015 13:03] seed@ubuntu:~$ sudo ufw deny proto tcp from  192.168.96.140 t
o 192.168.96.139 port 22
[sudo] password for seed:
Rules updated
[11/11/2015 13:07] seed@ubuntu:~$ ufw enable
ERROR: You need to be root to run this script
[11/11/2015 13:07] seed@ubuntu:~$ sudo ufw enable
Firewall is active and enabled on system startup
[11/11/2015 13:07] seed@ubuntu:~$ telnet 192.168.96.140
Trying 192.168.96.140...
telnet: Unable to connect to remote host: Connection timed out
[11/11/2015 13:09] seed@ubuntu:~$
```

```
[11/11/2015 13:07] root@ubuntu:/etc/default# sudo telnet 192.168.96.139
Trying 192.168.96.139...
telnet: Unable to connect to remote host: Connection timed out
[11/11/2015 13:08] root@ubuntu:/etc/default# 
```

Task 1 continued:

The third restriction involved using ufw to block access to a ip address. We chose to block the ip address associated with cise.ufl.edu. The screenshot shows Machine A timing out when attempting to connect to cise.ufl.edu



```
[11/11/2015 12:54] seed@ubuntu:~$ ssh 192.168.96.140
ssh: connect to host 192.168.96.140 port 22: Connection timed out
[11/11/2015 12:56] seed@ubuntu:~$ ssh 192.168.96.140
ssh: connect to host 192.168.96.140 port 22: Connection timed out
[11/11/2015 13:00] seed@ubuntu:~$ telnet 192.168.96.140
Trying 192.168.96.140...
telnet: Unable to connect to remote host: Connection timed out
[11/11/2015 13:02] seed@ubuntu:~$ clear
[11/11/2015 13:03] seed@ubuntu:~$ sudo ufw deny proto tcp from  192.168.96.140 t
o 192.168.96.139 port 22
[sudo] password for seed:
Rules updated
[11/11/2015 13:07] seed@ubuntu:~$ ufw enable
ERROR: You need to be root to run this script
[11/11/2015 13:07] seed@ubuntu:~$ sudo ufw enable
Firewall is active and enabled on system startup
[11/11/2015 13:07] seed@ubuntu:~$ telnet 192.168.96.140
Trying 192.168.96.140...
telnet: Unable to connect to remote host: Connection timed out
[11/11/2015 13:09] seed@ubuntu:~$ man ufw
[11/11/2015 13:11] seed@ubuntu:~$ sudo ufw deny out from any to cise.ufl.edu
ERROR: Bad destination address
[11/11/2015 13:12] seed@ubuntu:~$ sudo ufw deny out from any to 128.227.248.40
Rule added
[11/11/2015 13:13] seed@ubuntu:~$ 
```

The connection has timed out

The server at www.cise.ufl.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

Task 2: How Firewall Works

We implemented the following (5) required rules.

1. Restricted / Disallowed access from Machine A to Machine B (via Telnet).
2. Restricted / Disallowed access from Machine B to Machine A (via Telnet)
3. Block Access to ufl.edu (ip: "128.227.9.48")
4. Block Access to cise.ufl.edu (ip:"128.227.248.40")
5. Block Access to yahoo.com (ip:”206.190.36.45”)
6.

The firewall code we are running is known as task2.c the source code is provided for your convenience.

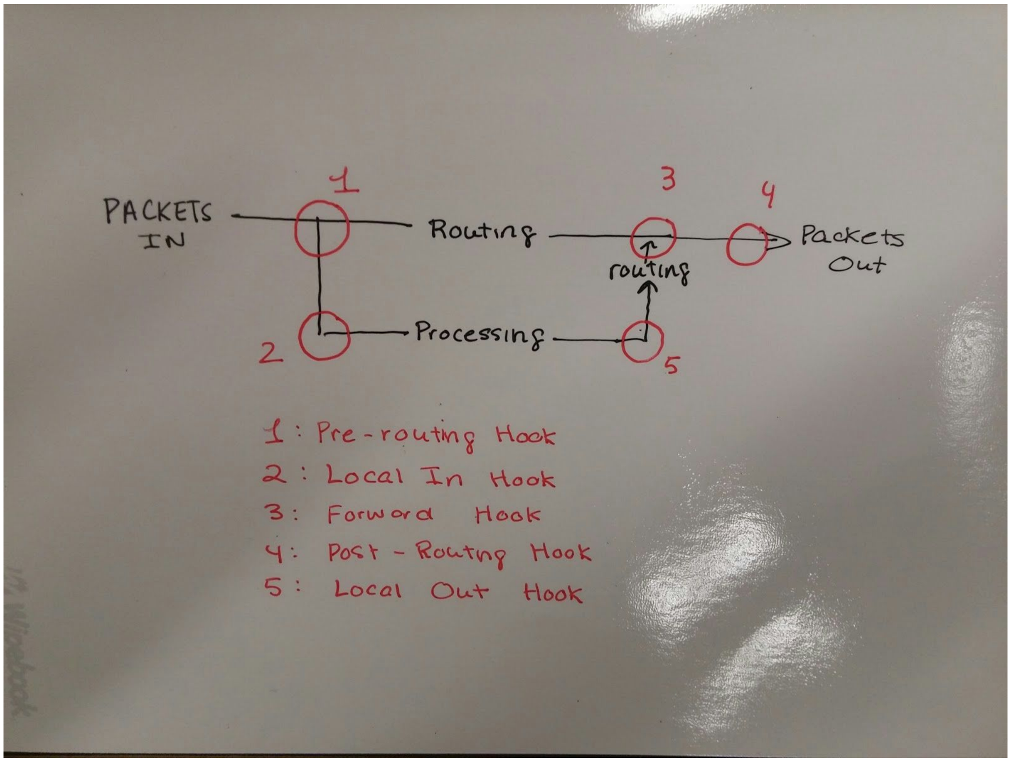
```
//task2.c
#define __KERNEL__
#define MODULE
#include <linux/ip.h>
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/netdevice.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/skbuff.h>
#include <linux/udp.h>
#include <linux/tcp.h>
#include <linux/ip.h>
//sd
static struct nf_hook_ops netfilter_ops;
//static unsigned char *ip_address = "\xC0\xA8\x00\x01";
static unsigned char *ip_address1 = "\x80\xE3\x09\x30";
//0x80E30930 ufl.edu
static unsigned char *ip_address2 = "\x62\x8A\xFD\x6D";
//0x628AFD6D belongs to yahoo.com
static char *interface = "lo";
unsigned char *port = "\x00\x17";

//FOR TCP STUFF
unsigned int src_port;
unsigned int dest_port;

struct sk_buff *sock_buff;
struct udphdr *udp_header;
struct tcphdr *tcp_header;

unsigned int main_hook(unsigned int hooknum,
               struct sk_buff *skb,
               const struct net_device *in,
               const struct net_device *out,
               int (*okfn)(struct sk_buff*))
{

  printk(KERN_INFO "main hook\n");

  sock_buff = skb;

  //get ip header info
  struct iphdr *ip_header = (struct iphdr
*)skb_network_header(sock_buff);

  char* UFL_EDU = "128.227.9.48";
  char*    YAHOO_COM = "206.190.36.45";
  char*    CISE_UF = "128.227.248.40";

    char* VM2 = "10.0.2.4";
    char* self = "10.0.2.15";
  //get source address

  if(!sock_buff){
     printk(KERN_INFO "socket buffer is empty\n");
     return NF_ACCEPT;
  }

  if(!ip_header){
    printk(KERN_INFO "no ip header in the socket buffer!\n");
    return NF_ACCEPT;
  }
  // get ip and analyze

  int ipSize = 15;
  char * str[ipSize];

  snprintf(str, ipSize, "%pI4", &ip_header->saddr);

  if(!strcmp(str, UFL_EDU) || !strcmp(str, YAHOO_COM) ||
!strcmp(str, CISE_UF) ) // if we wnated to make this dynamic,
register each of the banned IPS to an array

// figure out how to get user input for dynamic execution
(piping mostlikey)
  {
      printk(KERN_INFO "Found blacklisted IP [ %s ] Dropping
packet...\n", str);
      return NF_DROP; //drop it
  }

//block incoming telnet
    if(!strcmp(str, VM2))
    {
      if(ip_header->protocol == 6) //we got a TCP packet,
proceed to further anaylise
      {

           //we have the possible telnet connection b.w host and
VM2
           tcp_header= (struct tcphdr *)((__u32 *)ip_header+
ip_header->ihl); //this fixed the problem //figuire our the port
           int telnet = 23; //port to block (23 defaulted for
telent)

           unsigned int dport = htons((unsigned short int)
tcp_header->dest);
           //snprintf(port_string, pSize, "%s",
&tcp_header->dest);    //extracting readable port info
           printk(KERN_INFO "Found TCP packet from blacklisted
IP [ %s : %u ]..analyzing further \n" , str, dport);
```

```
           if(dport == telnet)
           {
                printk(KERN_INFO "Found telnet packet (port 23)
from blacklisted IP [ %s ]", str);
                //telnet default matches incoming destination
port, drop it
                return NF_DROP;
           }
      }
    }

//block outgoing telnet
  // get ip and anaylise

  snprintf(str, ipSize, "%pI4", &ip_header->daddr);

    if(!strcmp(str, self))
    {
      if(ip_header->protocol == 6) //we got a TCP packet,
proceed to further anaylise
      {

           //we have the possible telnet connection b.w host and
VM2
           tcp_header= (struct tcphdr *)((__u32 *)ip_header+
ip_header->ihl); //this fixed the problem //figuire our the port
           int telnet = 23; //port to block (23 defaulted for
telent)

           unsigned int sport = htons((unsigned short int)
tcp_header->source);
           //snprintf(port_string, pSize, "%s",
&tcp_header->dest);    //extracting readable port info
           printk(KERN_INFO "Found TCP packet transmission to
blacklisted IP [ %s : %u ]..analyzing further \n" , str, sport);
```

```
           if(sport == telnet)
           {
                printk(KERN_INFO "Found attempted telnet packet
(port 23) transmission to blacklisted IP [ %s ]", str);
                //telnet default matches incoming destination
port, drop it
                return NF_DROP;
           }
      }
    }


printk(KERN_INFO "Did not hit any firewall rules, packet
recieved...\n");
return NF_ACCEPT;
}
int init_module()
{
     netfilter_ops.hook            =    main_hook;
     netfilter_ops.pf              =    PF_INET;
     netfilter_ops.hooknum         =    NF_INET_PRE_ROUTING;
     netfilter_ops.priority        =    NF_IP_PRI_FIRST;
     nf_register_hook(&netfilter_ops);

return 0;
}
void cleanup_module() { nf_unregister_hook(&netfilter_ops); }
```

Question 1: What types of hooks does Netfilter support, and what can you do with these hooks? Please draw a diagram to show how packets flow through these hooks.

The netfilter framework supports 5 main hooks. Two hooks occur at the input of a packet. Pre-routing occurs before the packet has entered the network (this is good for blocking packets based on some header information). Local In, which will allow packets that have entered the system to be modified. Then we reach the forward hook, which is called whenever a packet is destined to be forwarded. Lastly we have the two hooks designed to top egress: post-routing, and local out. Local out can be used to block all network traffic, whereas post-routing can be used once information like destination have been attached to the packet.



Question 2: Where should you place a hook for ingress filtering, and where should you place a hook for egress filtering?

- In the diagram above from Question 1, ingress filtering should use Hook 1 or 2, the local in and pre-routing hooks. This way, we can inspect all aspects of the packet when it enter the machine and make a decision on how to deal with it accordingly. Egress filtering should use Hook 4 or 5, for local out or post routing. Similarly, we can inspect all parts of the packet (including all network layer headers) and decide if it is a packet that should be sent out, after routing has been completed so we can see where it's going.

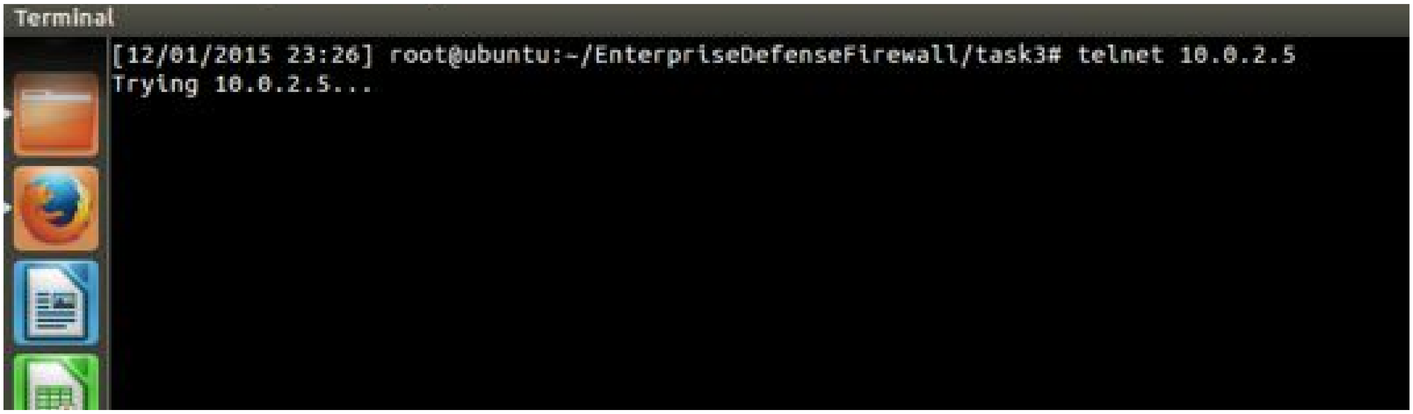
Question 3: Can you modify packets using Netfilter?

- Yes. You are capturing them and you have to access to the data fields such as port number, source and destination IP addresses, et cetera in a C program environment, without any kind of protection on these values to ensure their integrity (unless the intended host has a security check such as a hash or checksum), so you are quite free to modify packets at will using netfilter.

Task 3: Evading Egress Filtering

For this task we developed a special firewall that blocks all telnet communication, and blocks ufl.edu. This firewall is located in the task3.c firewall.

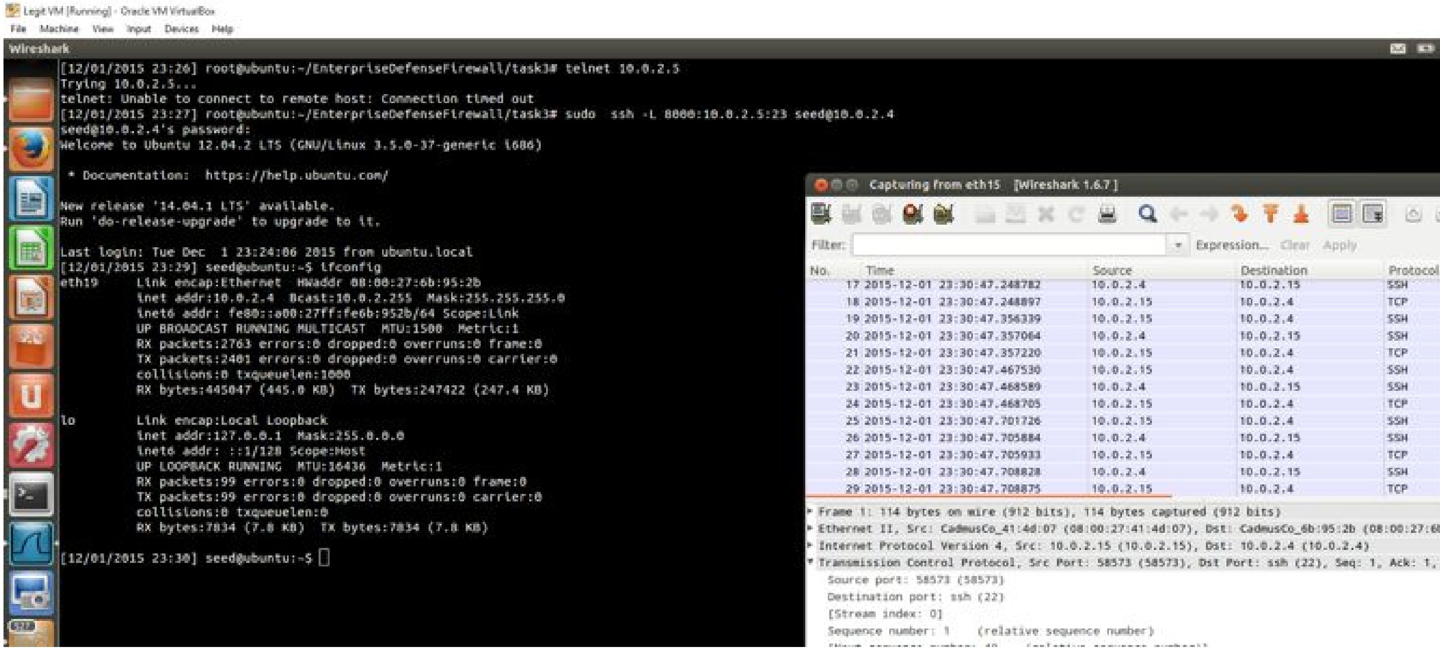
Task 3.a: Telnet to Machine B through the firewall



```
[12/01/2015 23:26] root@ubuntu:~/EnterpriseDefenseFirewall/task3# telnet 10.0.2.5
Trying 10.0.2.5...
```

As instructed we will use an SSH tunnel to bypass the firewalls restriction on telnet communication.

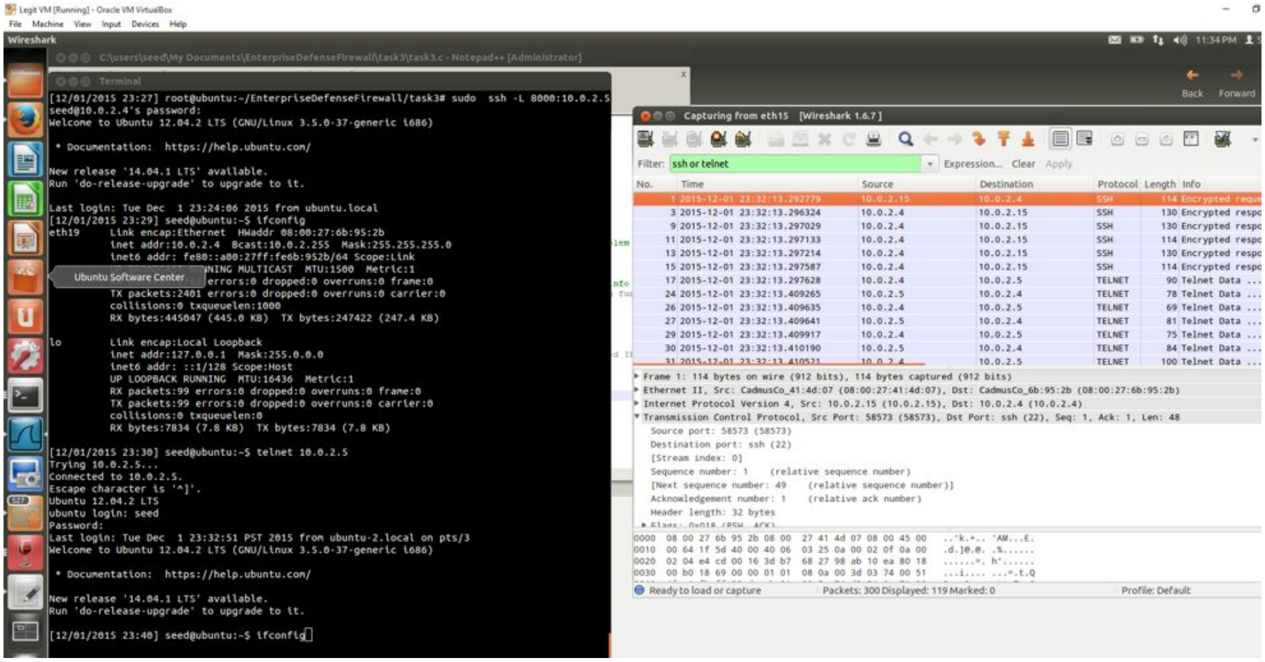
Establishing a SSH Tunnel from VM A to VM B


Legit VM [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Wireshark
Capturing from eth15 [Wireshark 1.6.7]
Filter: Expression... Clear Apply

```
[12/01/2015 23:26] root@ubuntu:~/EnterpriseDefenseFirewall/task3# telnet 10.0.2.5
Trying 10.0.2.5...
telnet: Unable to connect to remote host: Connection timed out
[12/01/2015 23:27] root@ubuntu:~/EnterpriseDefenseFirewall/task3# sudo  ssh -L 8000:10.0.2.5:23 seed@10.0.2.4
seed@10.0.2.4'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: Tue Dec  1 23:24:06 2015 from ubuntu.local
[12/01/2015 23:29] seed@ubuntu:~$ ifconfig
eth19     Link encap:Ethernet  HWaddr 08:00:27:6b:95:2b
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe6b:952b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2763 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2401 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:445047 (445.0 KB)  TX bytes:247422 (247.4 KB)

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:99 errors:0 dropped:0 overruns:0 frame:0
          TX packets:99 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:7834 (7.8 KB)  TX bytes:7834 (7.8 KB)

[12/01/2015 23:30] seed@ubuntu:~$ 
```

```
No.   Time                          Source       Destination  Protocol
   17 2015-12-01 23:30:47.248782    10.0.2.4     10.0.2.15    SSH
   18 2015-12-01 23:30:47.248897    10.0.2.15    10.0.2.4     TCP
   19 2015-12-01 23:30:47.356339    10.0.2.15    10.0.2.4     SSH
   20 2015-12-01 23:30:47.357064    10.0.2.4     10.0.2.15    SSH
   21 2015-12-01 23:30:47.357220    10.0.2.15    10.0.2.4     TCP
   22 2015-12-01 23:30:47.467530    10.0.2.15    10.0.2.4     SSH
   23 2015-12-01 23:30:47.468589    10.0.2.4     10.0.2.15    SSH
   24 2015-12-01 23:30:47.468705    10.0.2.15    10.0.2.4     TCP
   25 2015-12-01 23:30:47.701726    10.0.2.15    10.0.2.4     SSH
   26 2015-12-01 23:30:47.705884    10.0.2.4     10.0.2.15    SSH
   27 2015-12-01 23:30:47.705933    10.0.2.15    10.0.2.4     TCP
   28 2015-12-01 23:30:47.708828    10.0.2.4     10.0.2.15    SSH
   29 2015-12-01 23:30:47.708875    10.0.2.15    10.0.2.4     TCP
▸ Frame 1: 114 bytes on wire (912 bits), 114 bytes captured (912 bits)
▸ Ethernet II, Src: CadmusCo_41:4d:07 (08:00:27:41:4d:07), Dst: CadmusCo_6b:95:2b (08:00:27:6
▸ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 10.0.2.4 (10.0.2.4)
▾ Transmission Control Protocol, Src Port: 58573 (58573), Dst Port: ssh (22), Seq: 1, Ack: 1,
    Source port: 58573 (58573)
    Destination port: ssh (22)
    [Stream index: 0]
    Sequence number: 1    (relative sequence number)
```

WIthin the SSH Tunnel, establish a telnet connection from B to C (which is then forwarded to A). We observe through wireshark that telnet traffic is being forwarded from B to A through the tunnel.

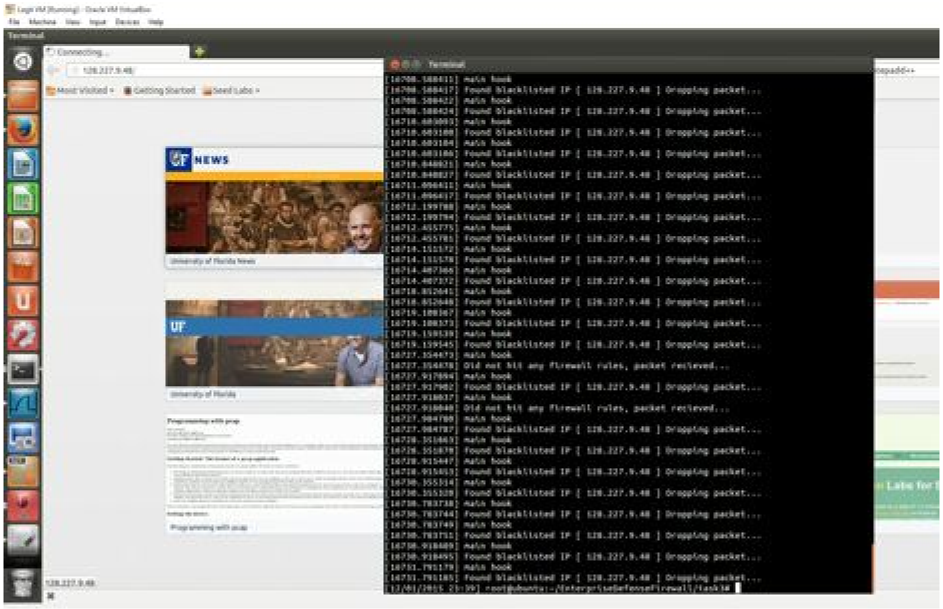

Legit VM [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Wireshark
C:\users\seed\My Documents\EnterpriseDefenseFirewall\task3\task3.c - Notepad++ [Administrator]
Terminal
Ubuntu Software Center
11:34 PM
Back
Forward
Capturing from eth15 [Wireshark 1.6.7]
Filter: ssh or telnet
Expression... Clear Apply

```
[12/01/2015 23:27] root@ubuntu:~/EnterpriseDefenseFirewall/task3# sudo  ssh -L 8000:10.0.2.5
seed@10.0.2.4'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: Tue Dec  1 23:24:06 2015 from ubuntu.local
[12/01/2015 23:29] seed@ubuntu:~$ ifconfig
eth19     Link encap:Ethernet  HWaddr 08:00:27:6b:95:2b
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe6b:952b/64 Scope:Link
                    NING MULTICAST  MTU:1500  Metric:1
                    errors:0 dropped:0 overruns:0 frame:0
          TX packets:2401 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:445047 (445.0 KB)  TX bytes:247422 (247.4 KB)

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:99 errors:0 dropped:0 overruns:0 frame:0
          TX packets:99 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:7834 (7.8 KB)  TX bytes:7834 (7.8 KB)

[12/01/2015 23:30] seed@ubuntu:~$ telnet 10.0.2.5
Trying 10.0.2.5...
Connected to 10.0.2.5.
Escape character is '^]'.
Ubuntu 12.04.2 LTS
ubuntu login: seed
Password:
Last login: Tue Dec  1 23:32:51 PST 2015 from ubuntu-2.local on pts/3
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.

[12/01/2015 23:40] seed@ubuntu:~$ ifconfig
```

```
No.   Time                          Source       Destination  Protocol Length Info
    1 2015-12-01 23:32:13.292779    10.0.2.15    10.0.2.4     SSH         114 Encrypted reque
    3 2015-12-01 23:32:13.296324    10.0.2.4     10.0.2.15    SSH         130 Encrypted respo
    9 2015-12-01 23:32:13.297029    10.0.2.4     10.0.2.15    SSH         130 Encrypted respo
   11 2015-12-01 23:32:13.297133    10.0.2.4     10.0.2.15    SSH         114 Encrypted respo
   13 2015-12-01 23:32:13.297214    10.0.2.4     10.0.2.15    SSH         130 Encrypted respo
   15 2015-12-01 23:32:13.297587    10.0.2.4     10.0.2.15    SSH         114 Encrypted respo
   17 2015-12-01 23:32:13.297628    10.0.2.4     10.0.2.5     TELNET       90 Telnet Data ...
   24 2015-12-01 23:32:13.409265    10.0.2.5     10.0.2.4     TELNET       78 Telnet Data ...
   26 2015-12-01 23:32:13.409635    10.0.2.4     10.0.2.5     TELNET       69 Telnet Data ...
   27 2015-12-01 23:32:13.409641    10.0.2.5     10.0.2.4     TELNET       81 Telnet Data ...
   29 2015-12-01 23:32:13.409917    10.0.2.4     10.0.2.5     TELNET       75 Telnet Data ...
   30 2015-12-01 23:32:13.410190    10.0.2.5     10.0.2.4     TELNET       84 Telnet Data ...
   31 2015-12-01 23:32:13.410521    10.0.2.4     10.0.2.5     TELNET      100 Telnet Data ...
▸ Frame 1: 114 bytes on wire (912 bits), 114 bytes captured (912 bits)
▸ Ethernet II, Src: CadmusCo_41:4d:07 (08:00:27:41:4d:07), Dst: CadmusCo_6b:95:2b (08:00:27:6b:95:2b)
▸ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 10.0.2.4 (10.0.2.4)
▾ Transmission Control Protocol, Src Port: 58573 (58573), Dst Port: ssh (22), Seq: 1, Ack: 1, Len: 48
    Source port: 58573 (58573)
    Destination port: ssh (22)
    [Stream index: 0]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 49    (relative sequence number)]
    Acknowledgement number: 1    (relative ack number)
    Header length: 32 bytes
  ▸ Flags: 0x018 (PSH, ACK)
0000  08 00 27 6b 95 2b 08 00  27 41 4d 07 08 00 45 00   ..'k.+.. 'AM...E.
0010  00 64 1f 5d 40 00 40 06  03 25 0a 00 02 0f 0a 00   .d.]@.@. .%......
0020  02 04 e4 cd 00 16 3d b7  68 27 98 ab 10 ea 80 18   ......=. h'......
0030  00 b0 18 69 00 00 01 01  08 0a 00 3d 03 74 00 51   ...i.... ...=.t.Q
Ready to load or capture     Packets: 300 Displayed: 119 Marked: 0     Profile: Default
```

Task 3.b: Connecting to “Facebook” aka Ufl.edu using SSH Tunnel.

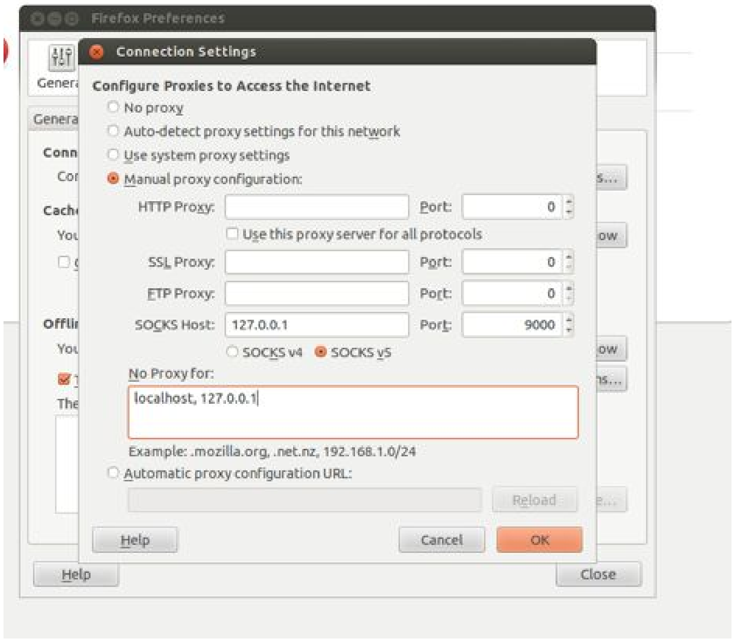
The screenshot demonstrates ufl.edu being blocked using our task 3.c firewall.



```
[16708.588411] main hook
[16708.588417] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16708.588422] main hook
[16708.588424] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16710.603093] main hook
[16710.603100] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16710.603104] main hook
[16710.603106] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16710.840821] main hook
[16710.840827] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16711.096411] main hook
[16711.096417] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16712.199788] main hook
[16712.199794] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16712.455775] main hook
[16712.455781] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16714.151572] main hook
[16714.151578] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16714.407366] main hook
[16714.407372] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16718.852641] main hook
[16718.852648] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16719.108367] main hook
[16719.108373] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16719.159539] main hook
[16719.159545] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16727.354473] main hook
[16727.354478] Did not hit any firewall rules, packet recieved...
[16727.917894] main hook
[16727.917902] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16727.918037] main hook
[16727.918040] Did not hit any firewall rules, packet recieved...
[16727.984780] main hook
[16727.984787] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16728.351863] main hook
[16728.351870] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16728.915447] main hook
[16728.915453] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16730.355314] main hook
[16730.355320] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16730.783738] main hook
[16730.783744] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16730.783749] main hook
[16730.783751] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16730.918489] main hook
[16730.918495] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[16731.791179] main hook
[16731.791185] Found blacklisted IP [ 128.227.9.48 ] Dropping packet...
[12/01/2015 21:39] root@ubuntu:~/EnterpriseDefenseFirewall/task3#
```

You can see that when we refresh the page our kernel log is being populated with ip address rejections with are coming from our netfilter based firewall. The page will not refresh and therefore the ip-address is blocked.

We then proceed to modify firefox settings in order to change the default port it is listening on (its new target will be the SSH port).



Firefox Preferences

Connection Settings

Configure Proxies to Access the Internet

○ No proxy
○ Auto-detect proxy settings for this network
○ Use system proxy settings
● Manual proxy configuration:

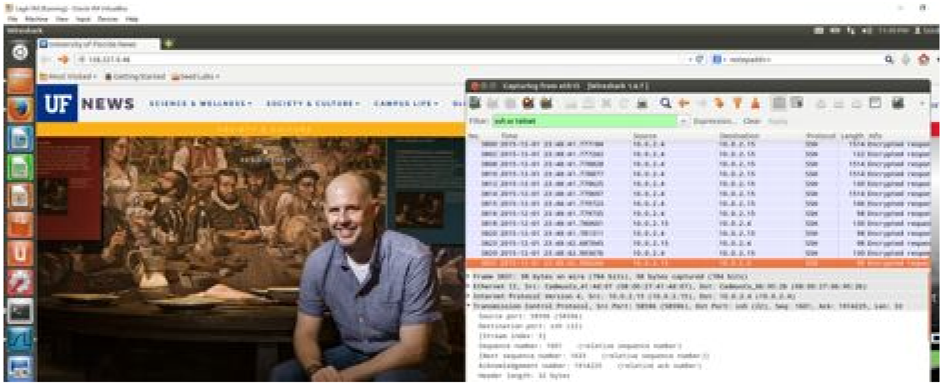
HTTP Proxy: Port: 0
□ Use this proxy server for all protocols
SSL Proxy: Port: 0
FTP Proxy: Port: 0
SOCKS Host: 127.0.0.1 Port: 9000
○ SOCKS v4 ● SOCKS v5

No Proxy for:
localhost, 127.0.0.1

Example: .mozilla.org, .net.nz, 192.168.1.0/24

○ Automatic proxy configuration URL:

We can now finally bypass the task 3.c firewall. Wireshark shows the packets running from C to B and then through our SSH tunnel back to A



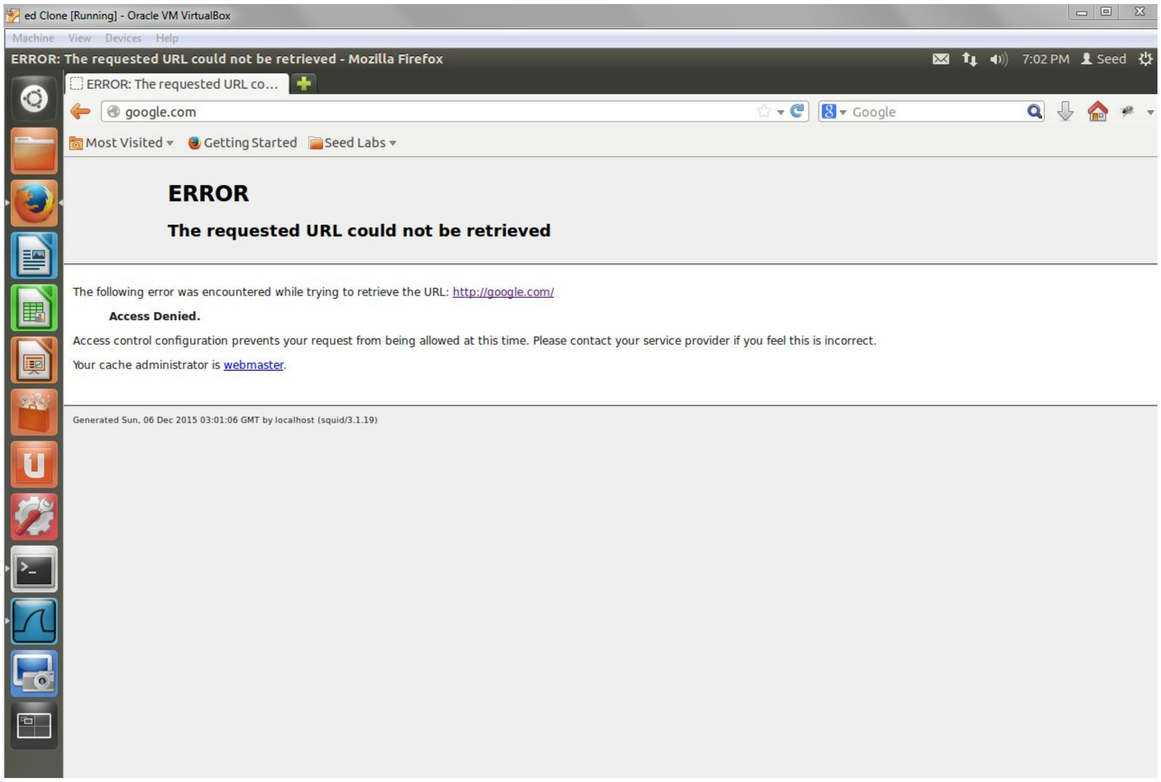
Question 4: If ufw blocks the TCP port 22, which is the port used by SSH, can you still set up an SSH tunnel to evade egress filtering?

- Yes. All you have to do is modify the SSH configuration to use another port other than Port 22. These port values chosen as the accepted port for certain traffic can be thought more of as guidelines than rules- you could just as easily send this traffic over a different port to the same effect, so long as you have the means to configure which port SSH connects to on your machine. In Linux machines, the file is typically located in /etc/ssh/sshd_conf. If you use a text editor on this file and change the port from 22 to another value and then save and restart to apply the changes.

Task 4: Web Proxy (Application Firewall)

Task 4.a: Setup

After the initial Squid setup was complete, I attempted to visit websites through VM A's browser. All websites were blocked. The two following screenshots show the proxy blocking google.com and the Wireshark packets.


ed Clone [Running] - Oracle VM VirtualBox
Machine View Devices Help
ERROR: The requested URL could not be retrieved - Mozilla Firefox
7:02 PM
Seed
ERROR: The requested URL co...
google.com
Google
Most Visited
Getting Started
Seed Labs

ERROR

The requested URL could not be retrieved

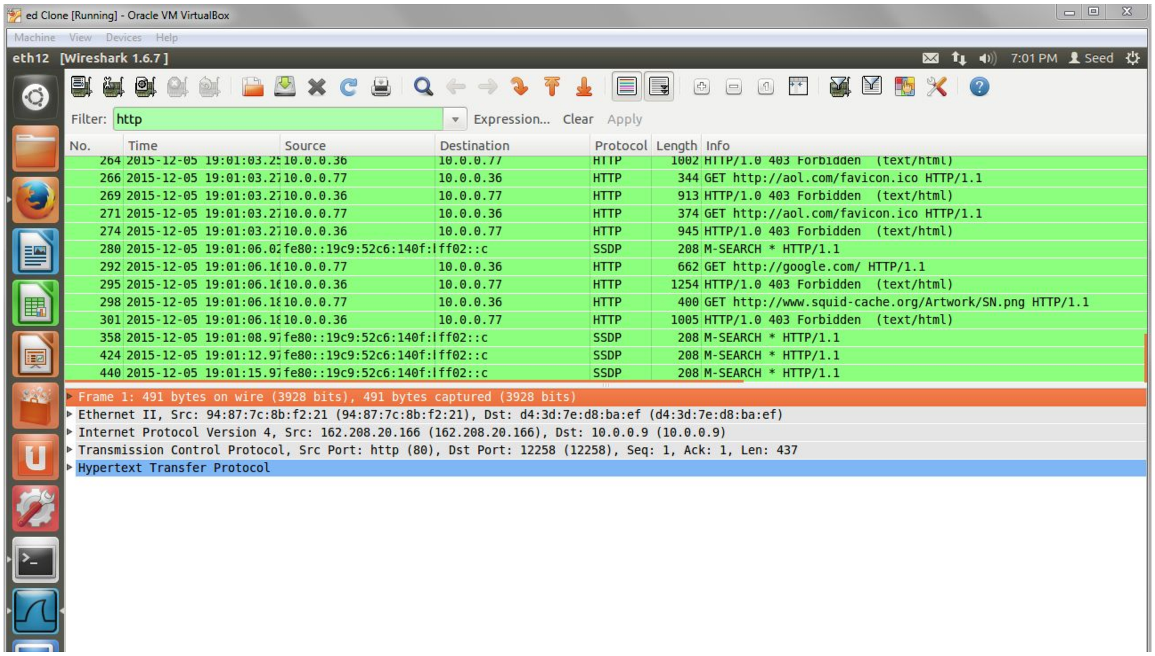
The following error was encountered while trying to retrieve the URL: http://google.com/

Access Denied.

Access control configuration prevents your request from being allowed at this time. Please contact your service provider if you feel this is incorrect.

Your cache administrator is webmaster.

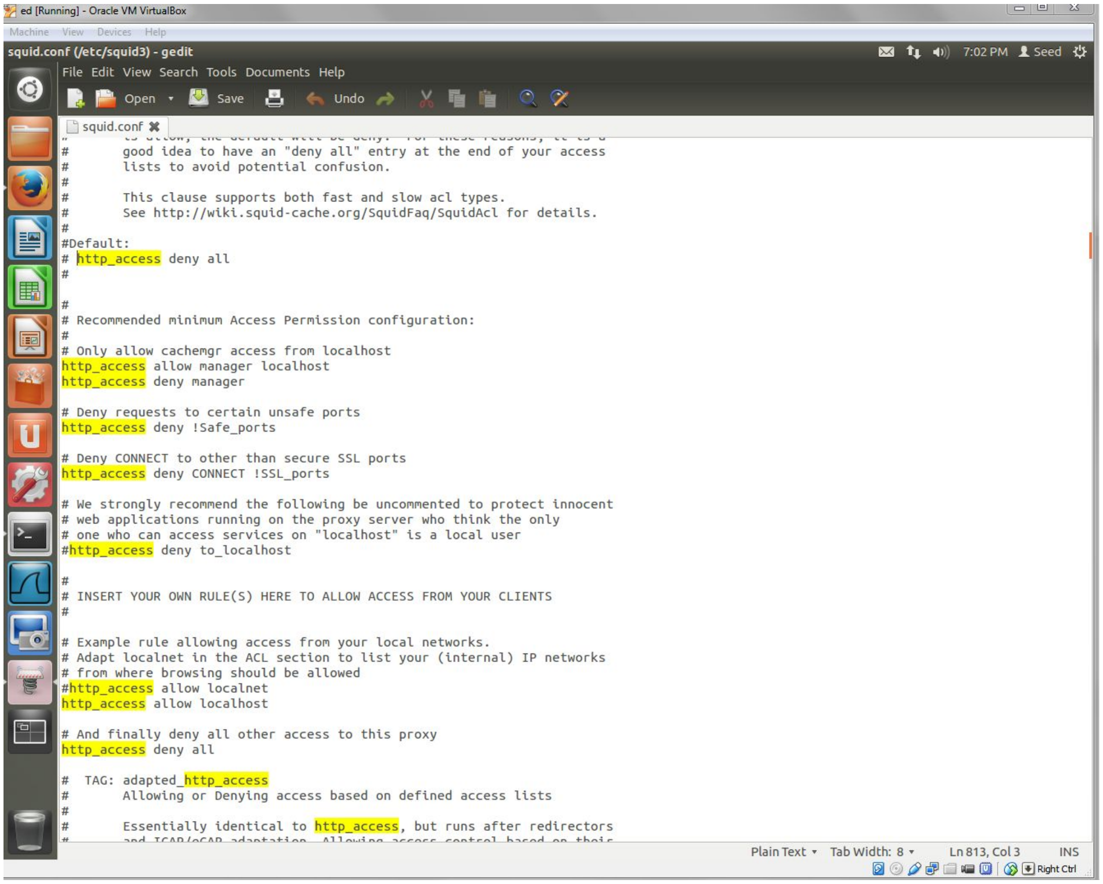
Generated Sun, 06 Dec 2015 03:01:06 GMT by localhost (squid/3.1.19)


ed Clone [Running] - Oracle VM VirtualBox
Machine View Devices Help
eth12 [Wireshark 1.6.7]
7:01 PM
Seed
Expression... Clear Apply

Filter: http

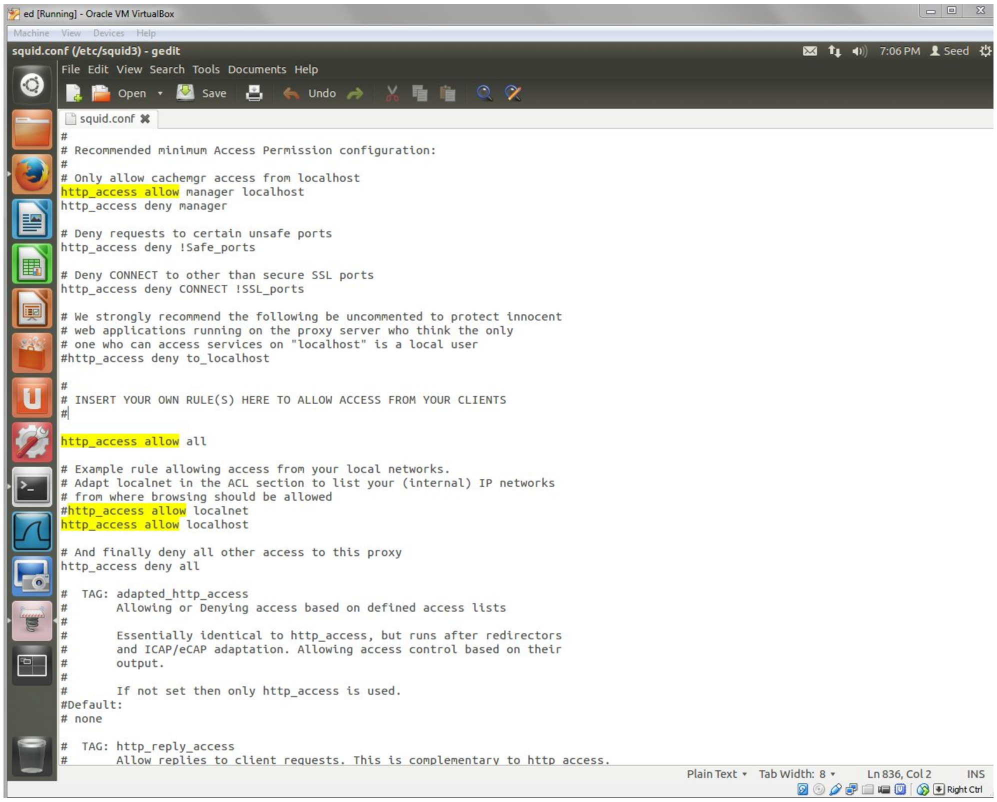
```
No.   Time                  Source                 Destination  Protocol Length Info
  264 2015-12-05 19:01:03.2510.0.0.36              10.0.0.77    HTTP     1002 HTTP/1.0 403 Forbidden  (text/html)
  266 2015-12-05 19:01:03.2710.0.0.77              10.0.0.36    HTTP      344 GET http://aol.com/favicon.ico HTTP/1.1
  269 2015-12-05 19:01:03.2710.0.0.36              10.0.0.77    HTTP      913 HTTP/1.0 403 Forbidden  (text/html)
  271 2015-12-05 19:01:03.2710.0.0.77              10.0.0.36    HTTP      374 GET http://aol.com/favicon.ico HTTP/1.1
  274 2015-12-05 19:01:03.2710.0.0.36              10.0.0.77    HTTP      945 HTTP/1.0 403 Forbidden  (text/html)
  280 2015-12-05 19:01:06.02fe80::19c9:52c6:140f:lff02::c      SSDP      208 M-SEARCH * HTTP/1.1
  292 2015-12-05 19:01:06.1610.0.0.77              10.0.0.36    HTTP      662 GET http://google.com/ HTTP/1.1
  295 2015-12-05 19:01:06.1610.0.0.36              10.0.0.77    HTTP     1254 HTTP/1.0 403 Forbidden  (text/html)
  298 2015-12-05 19:01:06.1810.0.0.77              10.0.0.36    HTTP      400 GET http://www.squid-cache.org/Artwork/SN.png HTTP/1.1
  301 2015-12-05 19:01:06.1810.0.0.36              10.0.0.77    HTTP     1005 HTTP/1.0 403 Forbidden  (text/html)
  358 2015-12-05 19:01:08.97fe80::19c9:52c6:140f:lff02::c      SSDP      208 M-SEARCH * HTTP/1.1
  424 2015-12-05 19:01:12.97fe80::19c9:52c6:140f:lff02::c      SSDP      208 M-SEARCH * HTTP/1.1
  440 2015-12-05 19:01:15.97fe80::19c9:52c6:140f:lff02::c      SSDP      208 M-SEARCH * HTTP/1.1

▸ Frame 1: 491 bytes on wire (3928 bits), 491 bytes captured (3928 bits)
▸ Ethernet II, Src: 94:87:7c:8b:f2:21 (94:87:7c:8b:f2:21), Dst: d4:3d:7e:d8:ba:ef (d4:3d:7e:d8:ba:ef)
▸ Internet Protocol Version 4, Src: 162.208.20.166 (162.208.20.166), Dst: 10.0.0.9 (10.0.0.9)
▸ Transmission Control Protocol, Src Port: http (80), Dst Port: 12258 (12258), Seq: 1, Ack: 1, Len: 437
▸ Hypertext Transfer Protocol
```

The following screenshot of squid.conf shows the rules that have caused all external websites to be blocked. The default rule is “http_access deny all”.

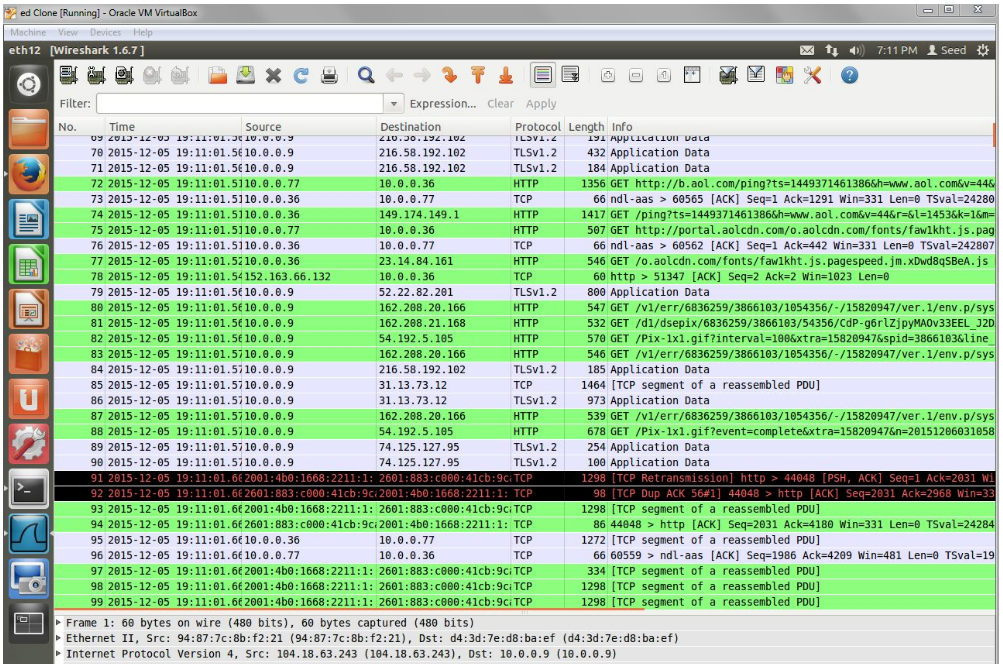

ed [Running] - Oracle VM VirtualBox
Machine View Devices Help
squid.conf (/etc/squid3) - gedit
7:02 PM
Seed
File Edit View Search Tools Documents Help
Open
Save
Undo
squid.conf
Plain Text
Tab Width: 8
Ln 813, Col 3
INS
Right Ctrl

```
#       is allow, the default will be deny.  For these reasons, it is a
#       good idea to have an "deny all" entry at the end of your access
#       lists to avoid potential confusion.
#
#       This clause supports both fast and slow acl types.
#       See http://wiki.squid-cache.org/SquidFaq/SquidAcl for details.
#
#Default:
# http_access deny all
#


#
# Recommended minimum Access Permission configuration:
#
# Only allow cachemgr access from localhost
http_access allow manager localhost
http_access deny manager

# Deny requests to certain unsafe ports
http_access deny !Safe_ports

# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports

# We strongly recommend the following be uncommented to protect innocent
# web applications running on the proxy server who think the only
# one who can access services on "localhost" is a local user
#http_access deny to_localhost


#
# 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
```

By adding “http_access allow all” under the rules in squid.conf, all websites will be allowed. The edited squid.conf file and the Wireshark packets connecting to aol.com and google.com are shown in the two following screenshots.


ed [Running] - Oracle VM VirtualBox
Machine View Devices Help
squid.conf (/etc/squid3) - gedit
7:06 PM Seed
File Edit View Search Tools Documents Help
Open Save Undo
squid.conf
Plain Text Tab Width: 8 Ln 836, Col 2 INS
Right Ctrl

```
#
# Recommended minimum Access Permission configuration:
#
# Only allow cachemgr access from localhost
http_access allow manager localhost
http_access deny manager

# Deny requests to certain unsafe ports
http_access deny !Safe_ports

# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports

# We strongly recommend the following be uncommented to protect innocent
# web applications running on the proxy server who think the only
# one who can access services on "localhost" is a local user
#http_access deny to_localhost

#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#

http_access allow all

# 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.
```

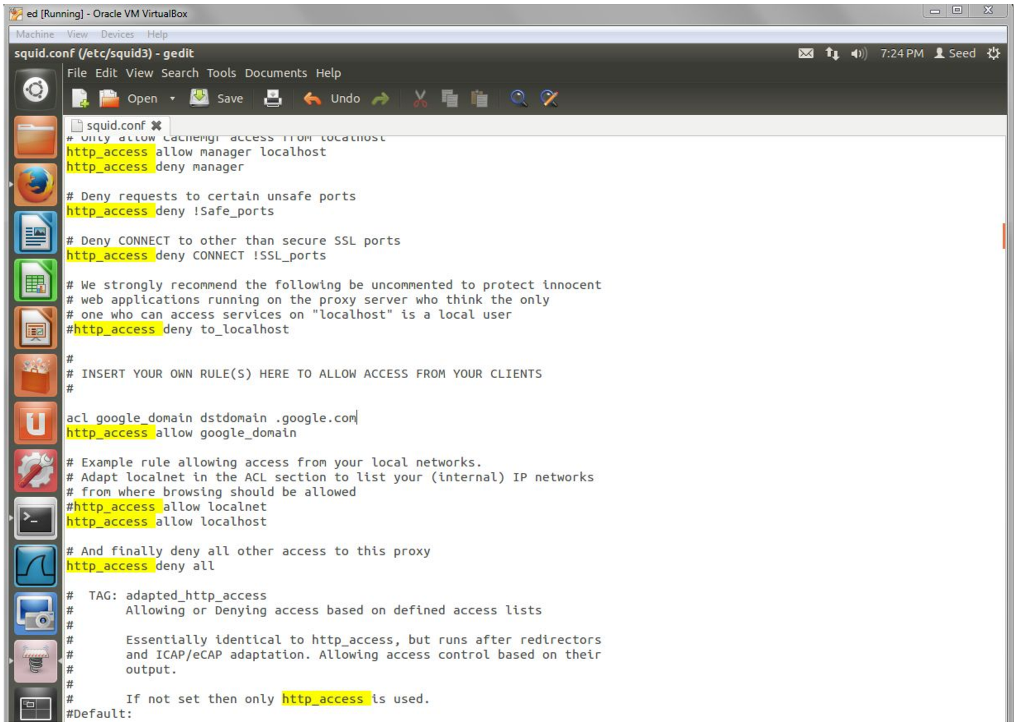

ed Clone [Running] - Oracle VM VirtualBox
Machine View Devices Help
eth12 [Wireshark 1.6.7]
7:11 PM Seed
Filter:
Expression... Clear Apply

```
No.  Time                   Source                  Destination             Protocol Length Info
  69 2015-12-05 19:11:01.5610.0.0.9                216.58.192.102          TLSv1.2   191 Application Data
  70 2015-12-05 19:11:01.5610.0.0.9                216.58.192.102          TLSv1.2   432 Application Data
  71 2015-12-05 19:11:01.5610.0.0.9                216.58.192.102          TLSv1.2   184 Application Data
  72 2015-12-05 19:11:01.5110.0.0.77               10.0.0.36               HTTP     1356 GET http://b.aol.com/ping?ts=1449371461386&h=www.aol.com&v=44&
  73 2015-12-05 19:11:01.5110.0.0.36               10.0.0.77               TCP        66 ndl-aas > 60565 [ACK] Seq=1 Ack=1291 Win=331 Len=0 TSval=24280
  74 2015-12-05 19:11:01.5110.0.0.36               149.174.149.1           HTTP     1417 GET /ping?ts=1449371461386&h=www.aol.com&v=44&r=&l=1453&k=1&m=
  75 2015-12-05 19:11:01.5110.0.0.77               10.0.0.36               HTTP      507 GET http://portal.aolcdn.com/o.aolcdn.com/fonts/faw1kht.js.pag
  76 2015-12-05 19:11:01.5110.0.0.36               10.0.0.77               TCP        66 ndl-aas > 60562 [ACK] Seq=1 Ack=442 Win=331 Len=0 TSval=242807
  77 2015-12-05 19:11:01.5210.0.0.36               23.14.84.161            HTTP      546 GET /o.aolcdn.com/fonts/faw1kht.js.pagespeed.jm.xDwd8qSBeA.js
  78 2015-12-05 19:11:01.54152.163.66.132          10.0.0.36               TCP        60 http > 51347 [ACK] Seq=2 Ack=2 Win=1023 Len=0
  79 2015-12-05 19:11:01.5610.0.0.9                52.22.82.201            TLSv1.2   800 Application Data
  80 2015-12-05 19:11:01.5610.0.0.9                162.208.20.166          HTTP      547 GET /v1/err/6836259/3866103/1054356/-/15820947/ver.1/env.p/sys
  81 2015-12-05 19:11:01.5610.0.0.9                162.208.21.168          HTTP      532 GET /d1/dsepix/6836259/3866103/54356/CdP-g6rlZjpyMAOv33EEL_J2D
  82 2015-12-05 19:11:01.5610.0.0.9                54.192.5.105            HTTP      570 GET /Pix-1x1.gif?interval=100&xtra=15820947&spid=3866103&line_
  83 2015-12-05 19:11:01.5710.0.0.9                162.208.20.166          HTTP      546 GET /v1/err/6836259/3866103/1054356/-/15820947/ver.1/env.p/sys
  84 2015-12-05 19:11:01.5710.0.0.9                216.58.192.102          TLSv1.2   185 Application Data
  85 2015-12-05 19:11:01.5710.0.0.9                31.13.73.12             TCP      1464 [TCP segment of a reassembled PDU]
  86 2015-12-05 19:11:01.5710.0.0.9                31.13.73.12             TLSv1.2   973 Application Data
  87 2015-12-05 19:11:01.5710.0.0.9                162.208.20.166          HTTP      539 GET /v1/err/6836259/3866103/1054356/-/15820947/ver.1/env.p/sys
  88 2015-12-05 19:11:01.5710.0.0.9                54.192.5.105            HTTP      678 GET /Pix-1x1.gif?event=complete&xtra=15820947&n=20151206031058
  89 2015-12-05 19:11:01.5710.0.0.9                74.125.127.95           TLSv1.2   254 Application Data
  90 2015-12-05 19:11:01.5710.0.0.9                74.125.127.95           TLSv1.2   100 Application Data
  91 2015-12-05 19:11:01.662001:4b0:1668:2211:1:   2601:883:c000:41cb:9caTCP      1298 [TCP Retransmission] http > 44048 [PSH, ACK] Seq=1 Ack=2031 Wi
  92 2015-12-05 19:11:01.662601:883:c000:41cb:9ca2001:4b0:1668:2211:1:   TCP        98 [TCP Dup ACK 56#1] 44048 > http [ACK] Seq=2031 Ack=2968 Win=33
  93 2015-12-05 19:11:01.662001:4b0:1668:2211:1:   2601:883:c000:41cb:9caTCP      1298 [TCP segment of a reassembled PDU]
  94 2015-12-05 19:11:01.662601:883:c000:41cb:9ca2001:4b0:1668:2211:1:   TCP        86 44048 > http [ACK] Seq=2031 Ack=4180 Win=331 Len=0 TSval=24284
  95 2015-12-05 19:11:01.6610.0.0.36               10.0.0.77               TCP      1272 [TCP segment of a reassembled PDU]
  96 2015-12-05 19:11:01.6610.0.0.77               10.0.0.36               TCP        66 60559 > ndl-aas [ACK] Seq=1986 Ack=4209 Win=481 Len=0 TSval=19
  97 2015-12-05 19:11:01.662001:4b0:1668:2211:1:   2601:883:c000:41cb:9caTCP       334 [TCP segment of a reassembled PDU]
  98 2015-12-05 19:11:01.662001:4b0:1668:2211:1:   2601:883:c000:41cb:9caTCP      1298 [TCP segment of a reassembled PDU]
  99 2015-12-05 19:11:01.662001:4b0:1668:2211:1:   2601:883:c000:41cb:9caTCP      1298 [TCP segment of a reassembled PDU]

▸ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
▸ Ethernet II, Src: 94:87:7c:8b:f2:21 (94:87:7c:8b:f2:21), Dst: d4:3d:7e:d8:ba:ef (d4:3d:7e:d8:ba:ef)
▸ Internet Protocol Version 4, Src: 104.18.63.243 (104.18.63.243), Dst: 10.0.0.9 (10.0.0.9)
```

To only allow access to google.com and no other websites, the following lines were added to squid.conf:

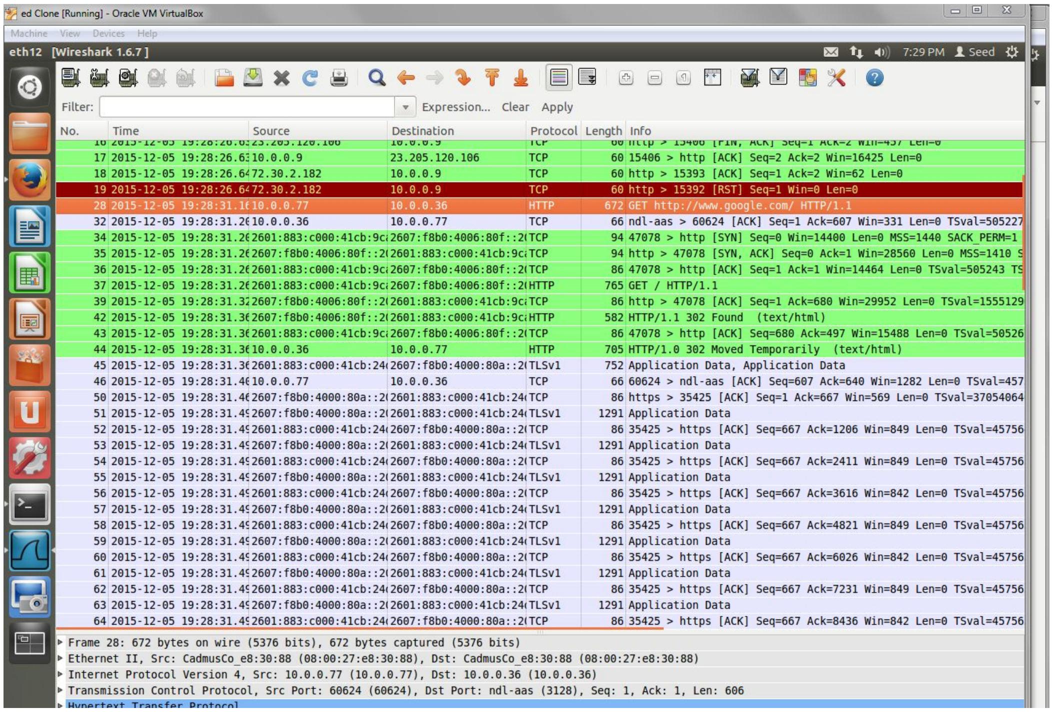
acl google_domain dstdomain .google.com

http_access allow google_domain


ed [Running] - Oracle VM VirtualBox
Machine View Devices Help
squid.conf (/etc/squid3) - gedit
7:24 PM Seed
File Edit View Search Tools Documents Help
Open Save Undo
squid.conf

```
# Only allow cachemgr access from localhost
http_access allow manager localhost
http_access deny manager

# Deny requests to certain unsafe ports
http_access deny !Safe_ports

# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports

# We strongly recommend the following be uncommented to protect innocent
# web applications running on the proxy server who think the only
# one who can access services on "localhost" is a local user
#http_access deny to_localhost

#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#

acl google_domain dstdomain .google.com
http_access allow google_domain

# 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:
```

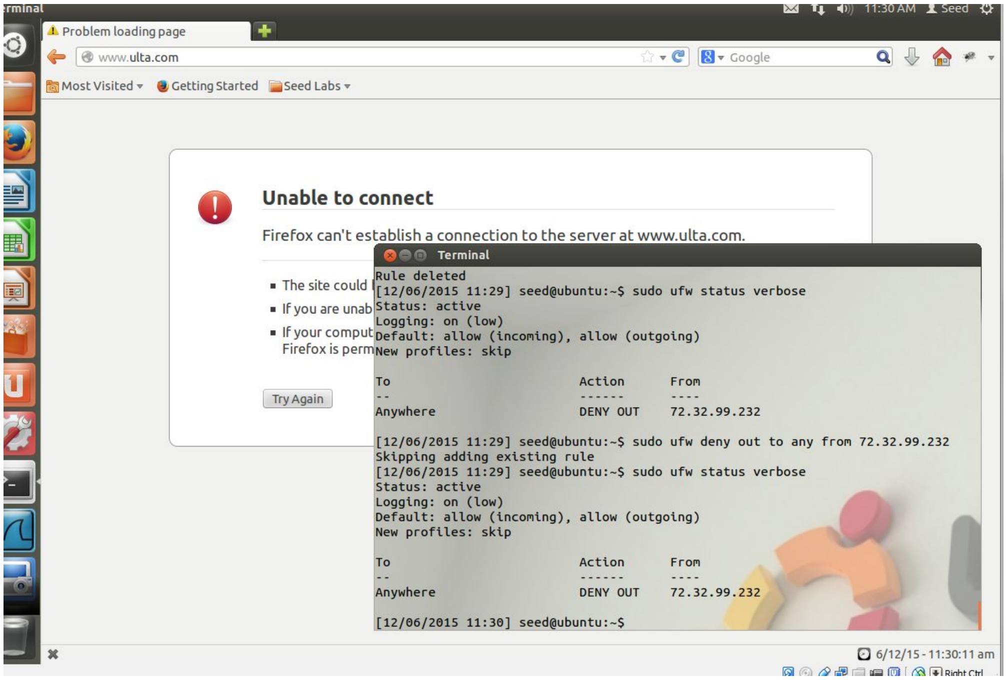
With the above rule, all websites except google.com are blocked. The Wireshark screenshot below shows access to aol.com being blocked by Squid.


ed Clone [Running] - Oracle VM VirtualBox
Machine View Devices Help
eth12 [Wireshark 1.6.7]
7:29 PM
Seed
Filter:
Expression...
Clear
Apply

```
No.  Time                Source                 Destination            Protocol Length Info
  16 2015-12-05 19:28:26.6 23.205.120.106       10.0.0.9               TCP        60 http > 15406 [FIN, ACK] Seq=1 Ack=2 Win=457 Len=0
  17 2015-12-05 19:28:26.6 10.0.0.9             23.205.120.106         TCP        60 15406 > http [ACK] Seq=2 Ack=2 Win=16425 Len=0
  18 2015-12-05 19:28:26.6 72.30.2.182          10.0.0.9               TCP        60 http > 15393 [ACK] Seq=1 Ack=2 Win=62 Len=0
  19 2015-12-05 19:28:26.6 72.30.2.182          10.0.0.9               TCP        60 http > 15392 [RST] Seq=1 Win=0 Len=0
  28 2015-12-05 19:28:31.1 10.0.0.77            10.0.0.36              HTTP      672 GET http://www.google.com/ HTTP/1.1
  32 2015-12-05 19:28:31.2 10.0.0.36            10.0.0.77              TCP        66 ndl-aas > 60624 [ACK] Seq=1 Ack=607 Win=331 Len=0 TSval=505227
  34 2015-12-05 19:28:31.2 2601:883:c000:41cb:9c 2607:f8b0:4006:80f::2 TCP        94 47078 > http [SYN] Seq=0 Win=14400 Len=0 MSS=1440 SACK_PERM=1
  35 2015-12-05 19:28:31.2 2607:f8b0:4006:80f::2 2601:883:c000:41cb:9c TCP        94 http > 47078 [SYN, ACK] Seq=0 Ack=1 Win=28560 Len=0 MSS=1410 S
  36 2015-12-05 19:28:31.2 2601:883:c000:41cb:9c 2607:f8b0:4006:80f::2 TCP        86 47078 > http [ACK] Seq=1 Ack=1 Win=14464 Len=0 TSval=505243 TS
  37 2015-12-05 19:28:31.2 2601:883:c000:41cb:9c 2607:f8b0:4006:80f::2 HTTP      765 GET / HTTP/1.1
  39 2015-12-05 19:28:31.3 2607:f8b0:4006:80f::2 2601:883:c000:41cb:9c TCP        86 http > 47078 [ACK] Seq=1 Ack=680 Win=29952 Len=0 TSval=1555129
  42 2015-12-05 19:28:31.3 2607:f8b0:4006:80f::2 2601:883:c000:41cb:9c HTTP      582 HTTP/1.1 302 Found  (text/html)
  43 2015-12-05 19:28:31.3 2601:883:c000:41cb:9c 2607:f8b0:4006:80f::2 TCP        86 47078 > http [ACK] Seq=680 Ack=497 Win=15488 Len=0 TSval=50526
  44 2015-12-05 19:28:31.3 10.0.0.36            10.0.0.77              HTTP      705 HTTP/1.0 302 Moved Temporarily  (text/html)
  45 2015-12-05 19:28:31.3 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TLSv1     752 Application Data, Application Data
  46 2015-12-05 19:28:31.4 10.0.0.77            10.0.0.36              TCP        66 60624 > ndl-aas [ACK] Seq=607 Ack=640 Win=1282 Len=0 TSval=457
  50 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TCP        86 https > 35425 [ACK] Seq=1 Ack=667 Win=569 Len=0 TSval=37054064
  51 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TLSv1    1291 Application Data
  52 2015-12-05 19:28:31.4 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TCP        86 35425 > https [ACK] Seq=667 Ack=1206 Win=849 Len=0 TSval=45756
  53 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TLSv1    1291 Application Data
  54 2015-12-05 19:28:31.4 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TCP        86 35425 > https [ACK] Seq=667 Ack=2411 Win=849 Len=0 TSval=45756
  55 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TLSv1    1291 Application Data
  56 2015-12-05 19:28:31.4 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TCP        86 35425 > https [ACK] Seq=667 Ack=3616 Win=842 Len=0 TSval=45756
  57 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TLSv1    1291 Application Data
  58 2015-12-05 19:28:31.4 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TCP        86 35425 > https [ACK] Seq=667 Ack=4821 Win=849 Len=0 TSval=45756
  59 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TLSv1    1291 Application Data
  60 2015-12-05 19:28:31.4 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TCP        86 35425 > https [ACK] Seq=667 Ack=6026 Win=842 Len=0 TSval=45756
  61 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TLSv1    1291 Application Data
  62 2015-12-05 19:28:31.4 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TCP        86 35425 > https [ACK] Seq=667 Ack=7231 Win=849 Len=0 TSval=45756
  63 2015-12-05 19:28:31.4 2607:f8b0:4000:80a::2 2601:883:c000:41cb:24 TLSv1    1291 Application Data
  64 2015-12-05 19:28:31.4 2601:883:c000:41cb:24 2607:f8b0:4000:80a::2 TCP        86 35425 > https [ACK] Seq=667 Ack=8436 Win=842 Len=0 TSval=45756

▸ Frame 28: 672 bytes on wire (5376 bits), 672 bytes captured (5376 bits)
▸ Ethernet II, Src: CadmusCo_e8:30:88 (08:00:27:e8:30:88), Dst: CadmusCo_e8:30:88 (08:00:27:e8:30:88)
▸ Internet Protocol Version 4, Src: 10.0.0.77 (10.0.0.77), Dst: 10.0.0.36 (10.0.0.36)
▸ Transmission Control Protocol, Src Port: 60624 (60624), Dst Port: ndl-aas (3128), Seq: 1, Ack: 1, Len: 606
▸ Hypertext Transfer Protocol
```

Task 4.b: Using Web Proxy to evade Firewall

The ufw firewall can be evaded by using the web proxy. I chose to block access to Ulta.com (72.32.99.232) (I could not block all the IPs belonging to Facebook). I turned off the proxy on VM A, and enabled ufw and created the rule to deny out from the IP (see first screenshot).



Unable to connect

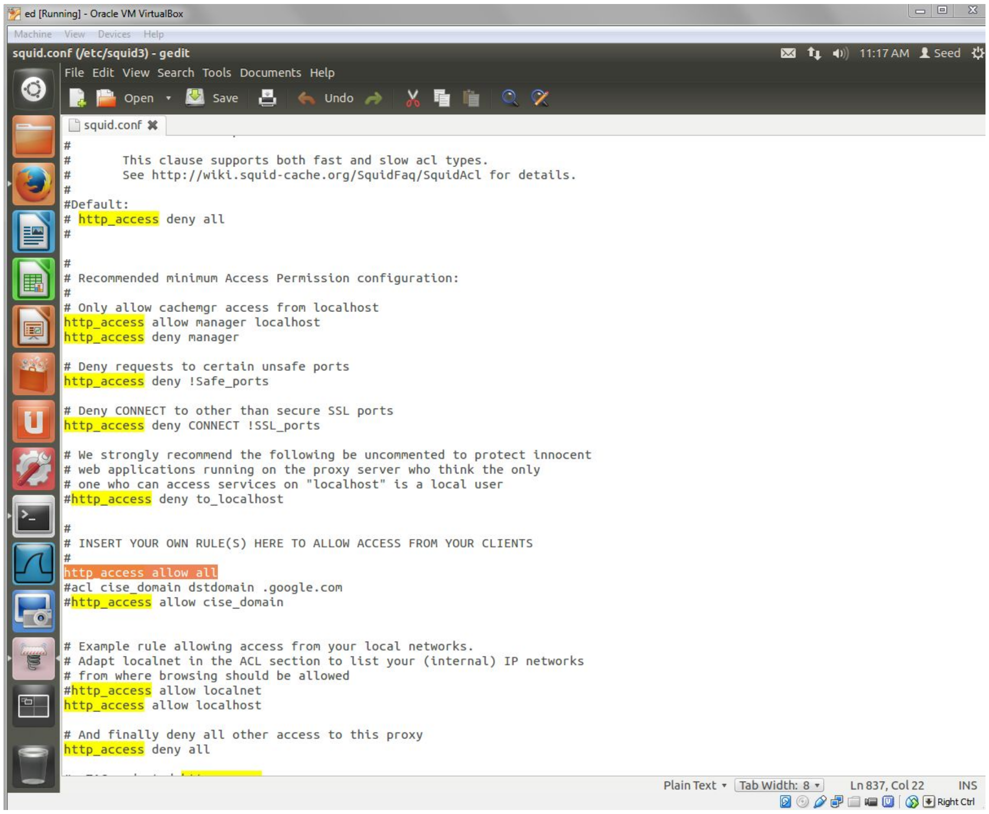
Firefox can't establish a connection to the server at www.ulta.com.

- The site could
- If you are unab
- If your comput
  Firefox is perm

Try Again

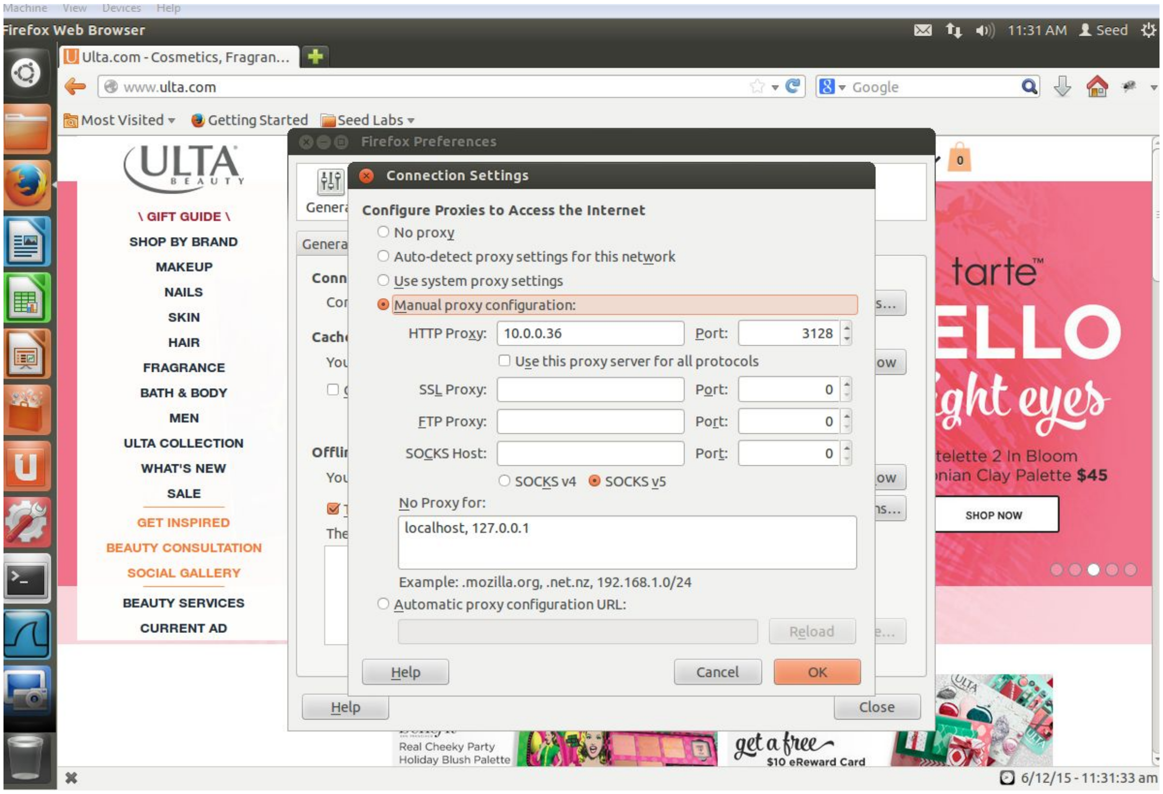
Terminal

```
Rule deleted
[12/06/2015 11:29] seed@ubuntu:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: allow (incoming), allow (outgoing)
New profiles: skip

To                         Action      From
--                         ------      ----
Anywhere                   DENY OUT    72.32.99.232

[12/06/2015 11:29] seed@ubuntu:~$ sudo ufw deny out to any from 72.32.99.232
Skipping adding existing rule
[12/06/2015 11:29] seed@ubuntu:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: allow (incoming), allow (outgoing)
New profiles: skip

To                         Action      From
--                         ------      ----
Anywhere                   DENY OUT    72.32.99.232

[12/06/2015 11:30] seed@ubuntu:~$
```



ed [Running] - Oracle VM VirtualBox
Machine View Devices Help
squid.conf (/etc/squid3) - gedit
11:17 AM Seed
File Edit View Search Tools Documents Help
Open Save Undo
squid.conf
Plain Text Tab Width: 8 Ln 837, Col 22 INS
Right Ctrl

```
#
#       This clause supports both fast and slow acl types.
#       See http://wiki.squid-cache.org/SquidFaq/SquidAcl for details.
#
#Default:
# http_access deny all
#


#
# Recommended minimum Access Permission configuration:
#
# Only allow cachemgr access from localhost
http_access allow manager localhost
http_access deny manager

# Deny requests to certain unsafe ports
http_access deny !Safe_ports

# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports

# We strongly recommend the following be uncommented to protect innocent
# web applications running on the proxy server who think the only
# one who can access services on "localhost" is a local user
#http_access deny to_localhost

#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
http_access allow all
#acl cise_domain dstdomain .google.com
#http_access allow cise_domain


# 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
```



Configure Proxies to Access the Internet

○ No proxy
○ Auto-detect proxy settings for this network
○ Use system proxy settings
● Manual proxy configuration:

HTTP Proxy: 10.0.0.36 Port: 3128

□ Use this proxy server for all protocols

SSL Proxy: Port: 0

FTP Proxy: Port: 0

SOCKS Host: Port: 0

○ SOCKS v4 ● SOCKS v5

No Proxy for:

localhost, 127.0.0.1

Example: .mozilla.org, .net.nz, 192.168.1.0/24

○ Automatic proxy configuration URL:

Reload

In squid.conf, I added the rule "http_acess allow all" to allow all websites. I re-enabled the proxy and was able to connect to Ulta.com even though ufw was blocking it.

Question 5: If ufw blocks the TCP port 3128, can you still use the web proxy to evade the firewall?

Yes, it is still possible to use the web proxy to evade the firewall even if ufw blocks port 3128. This is done by changing the port in squid.conf from 3128 to some other port.

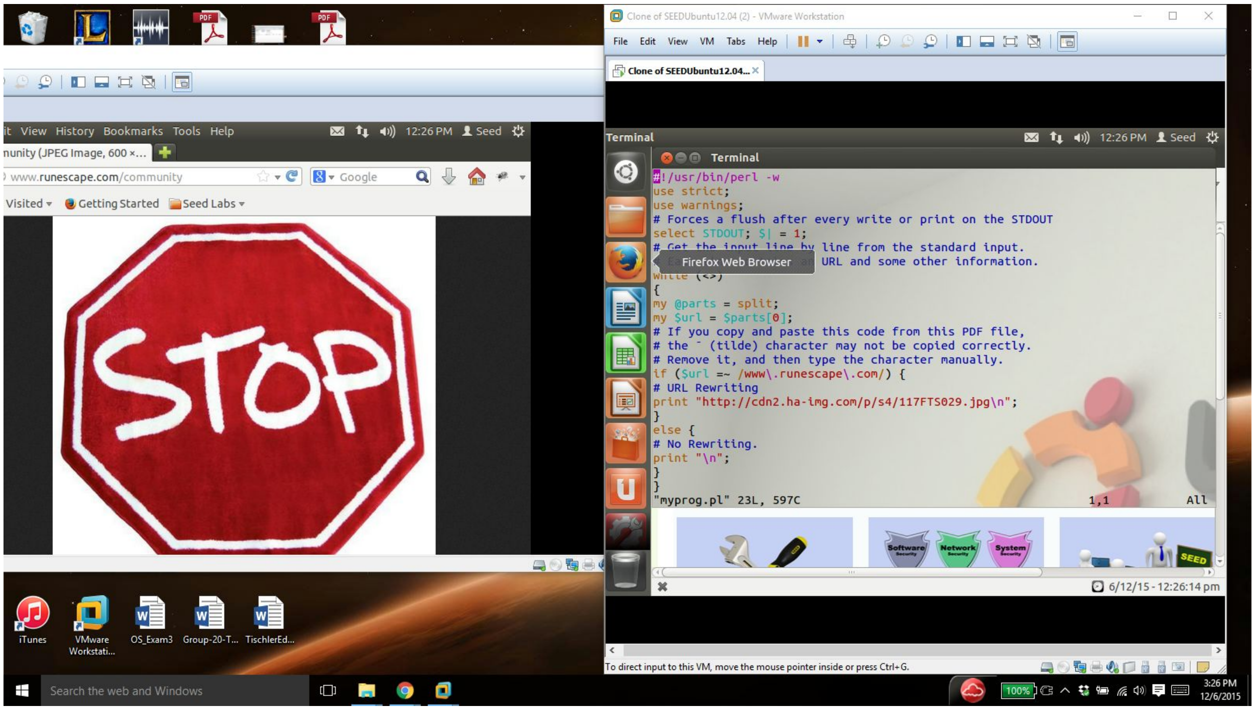
Task 4.c: URL Rewriting/Redirection

Please describe what the above program does (e.g. what URLs got rewritten and what your observations are).

- First, the program above does not work. The script is supposed to send someone using the proxy that attempts to connect to www.cis.syr.edu to www.yahoo.com. However, www.cis.syr.edu does not exist anymore so it does not quite work. However, changing the site to one that exists causes the script to work.

Please modify the above program, so it replaces all the Facebook pages with a page that shows a big red stop sign

- Facebook is not an http website anymore. Facebook uses https so it does not work with the url_rewriter. Instead we blocked www.runescape.com which is equally still solves the same task that is meant to be completed.

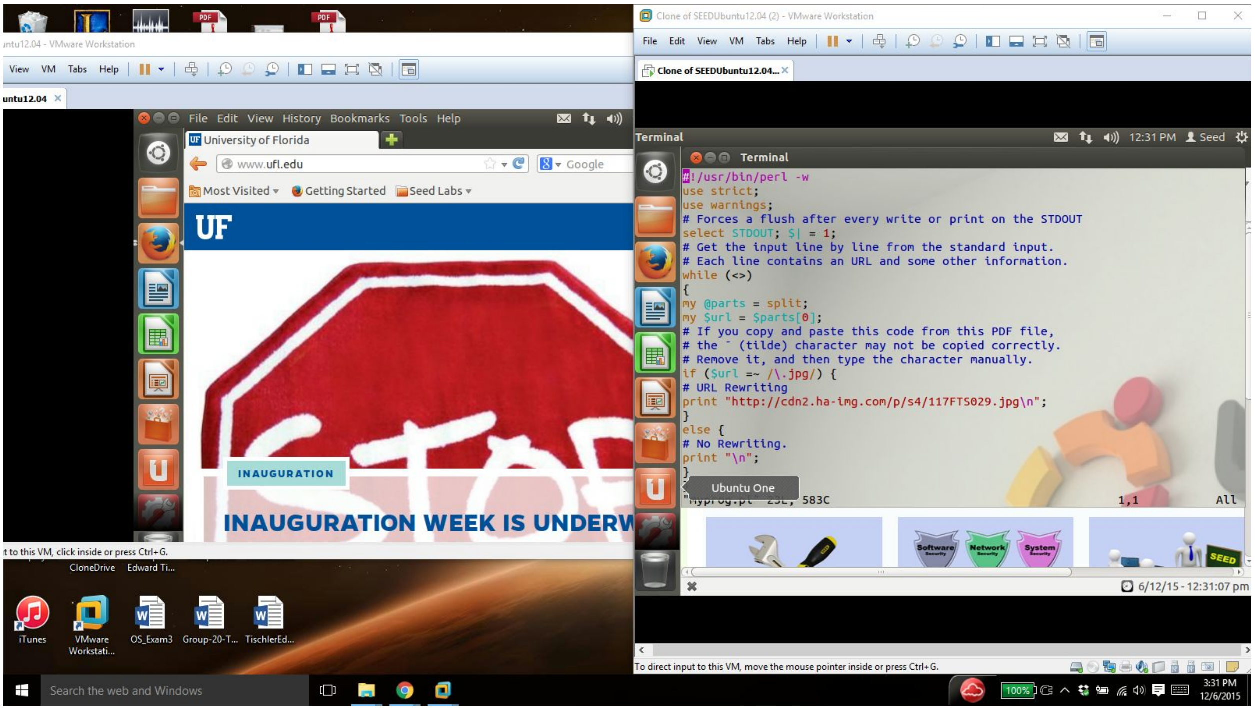


Clone of SEEDUbuntu12.04 (2) - VMware Workstation
File Edit View VM Tabs Help
it View History Bookmarks Tools Help
nunity (JPEG Image, 600 ×...
www.runescape.com/community
Google
Visited
Getting Started
Seed Labs
12:26 PM
Seed
Terminal
Firefox Web Browser
Software
Network
System
SEED
6/12/15 - 12:26:14 pm
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
iTunes
VMware Workstati...
OS_Exam3
Group-20-T...
TischlerEd...
Search the web and Windows
100%
3:26 PM
12/6/2015

```
#!/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.
# Ea          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\.runescape\.com/) {
# URL Rewriting
print "http://cdn2.ha-img.com/p/s4/117FTS029.jpg\n";
}
else {
# No Rewriting.
print "\n";
}
}
"myprog.pl" 23L, 597C                                   1,1           All
```

Please modify the above program, so it replaces all the images (e.g. .jpg or .gif images) inside any page with a picture of your choice. When an HTML web page contains images, the browser will identify those image URLs, and send out an URL request for each image. You should be able to see the URLs in your URL rewriting program. You just need to decide whether a URL is trying to fetch an image file of a particular type; if it is, you can replace the URL with another one (of your choice).

- As the picture below shows. We successfully blocked jpg images on ufl.edu. If using the proxy and attempting to connect to a website with jpg images you will now only see stop signs.



untu12.04 - VMware Workstation
View VM Tabs Help
File Edit View History Bookmarks Tools Help
University of Florida
www.ufl.edu
Most Visited
Getting Started
Seed Labs
UF
INAUGURATION
INAUGURATION WEEK IS UNDERW
Clone of SEEDUbuntu12.04 (2) - VMware Workstation
File Edit View VM Tabs Help
Clone of SEEDUbuntu12.04...
Terminal
12:31 PM
Seed
Ubuntu One
Software Security
Network Security
System Security
SEED
6/12/15 - 12:31:07 pm
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
iTunes
VMware Workstati...
OS_Exam3
Group-20-T...
TischlerEd...
Search the web and Windows
3:31 PM
12/6/2015

```
#!/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 =~ /\.jpg/) {
# URL Rewriting
print "http://cdn2.ha-img.com/p/s4/117FTS029.jpg\n";
}
else {
# No Rewriting.
print "\n";
}
"myprog.pl" 23L, 583C                                        1,1           All
```

Question 6: We can use SSH and HTTP protocols as tunnels to evade the egress filtering. Can we use the ICMP protocol as a tunnel to evade the egress filtering? Please briefly describe how.

- Yes, ICMP protocol can be used for tunnelling to evade egress filtering. A computer may inject data into an ICMP echo packet and send it to the intended recipient. The recipient can similarly send a ping reply with the reply data attached. Effectively, a client-proxy connection is established that can bypass firewall rules. Most firewalls will not block ICMP traffic, so cleverly using echos/pings to store requests of arbitrary length and receive packets of arbitrary length achieves tunneling as SSH and HTTP did. This is all possible due to IEEE RFC 792, which allows for the arbitrary length of data to be attached to an ICMP packet of type 0 (echo reply) or type 8 (echo message). Had that RFC restricted data size, this method would not work.