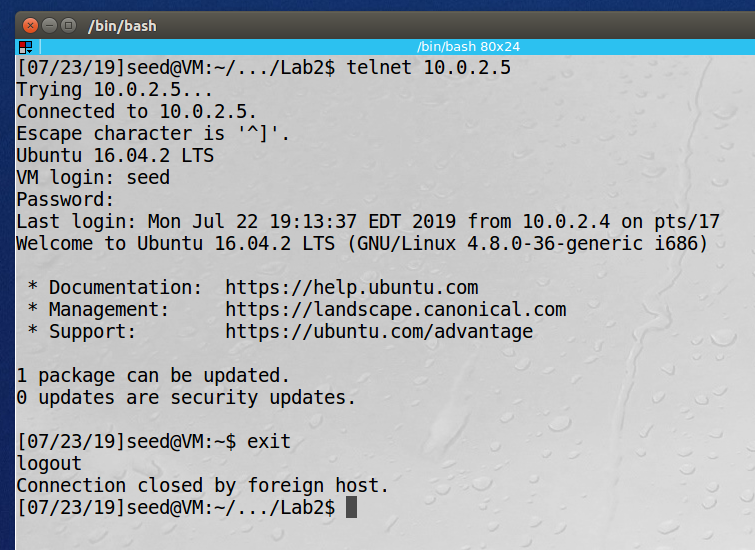
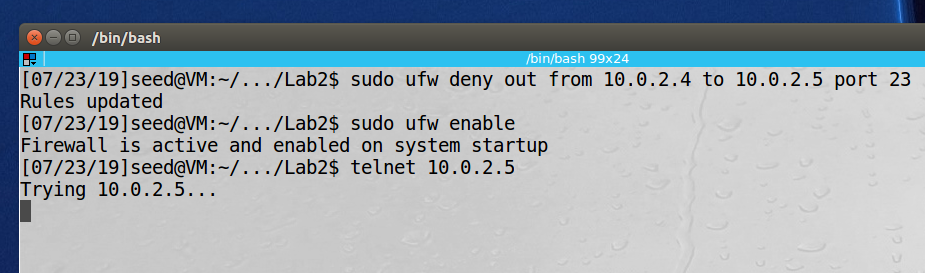
Internet Security - Lab 2 Report

# Using Firewall

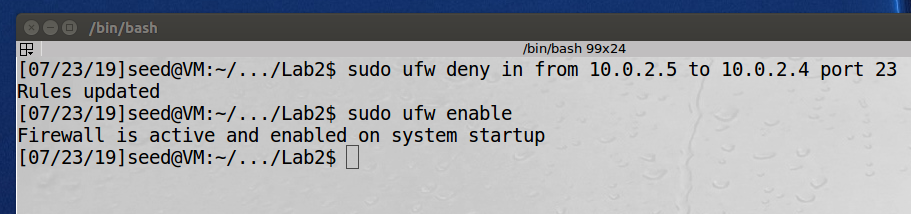
From Machine A connecting to Machine B before the firewall rule was created/enabled:



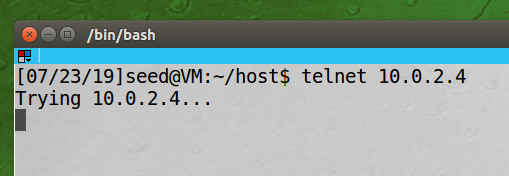
The firewall rule created and enabled on Machine A blocking telnet traffic to Machine B and showing that it is not able to connect:



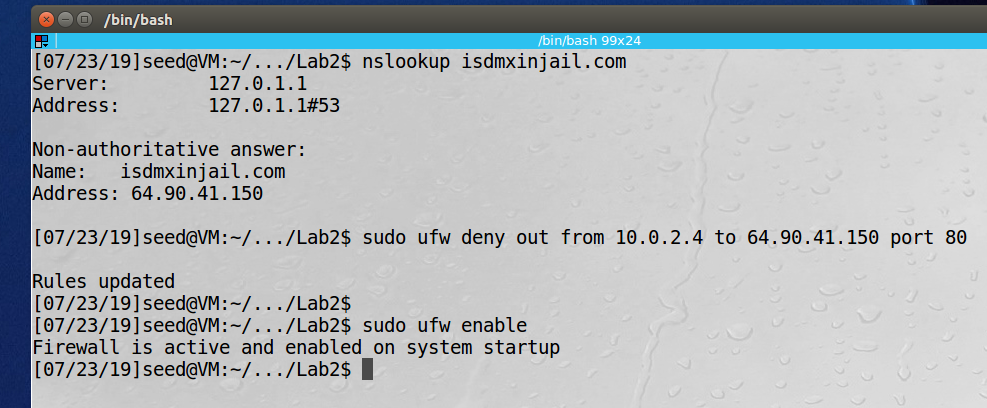
Firewall rule created on Machine A blocking telnet from Machine B:



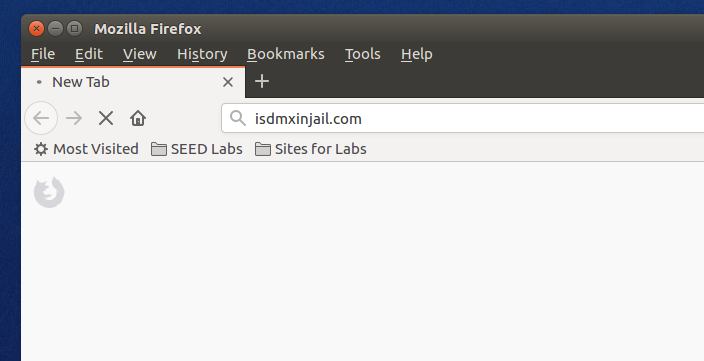
Showing that Machine B can not telnet into Machine A:



Creating and enabling a rule to block all web traffic to the website, isdmxinjail.com:



Showing that the site is unreachable:

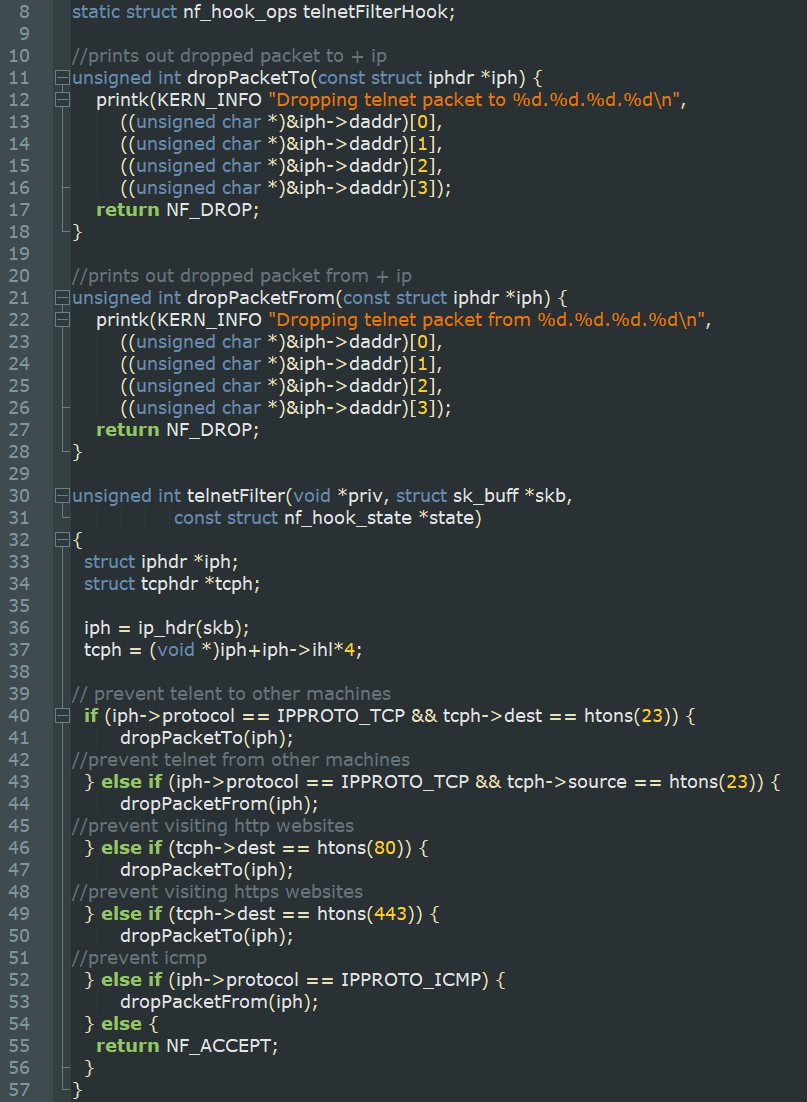


**Observation:** The screenshots above demonstrate the use of the UFW program to create simple packet filtering firewall rules. We first blocked telnet traffic from Machine A to Machine B, then we blocked telnet traffic from Machine B to Machine A and finally we blocked port 80 (http) traffic from Machine A to the website isdmxinjail.com.

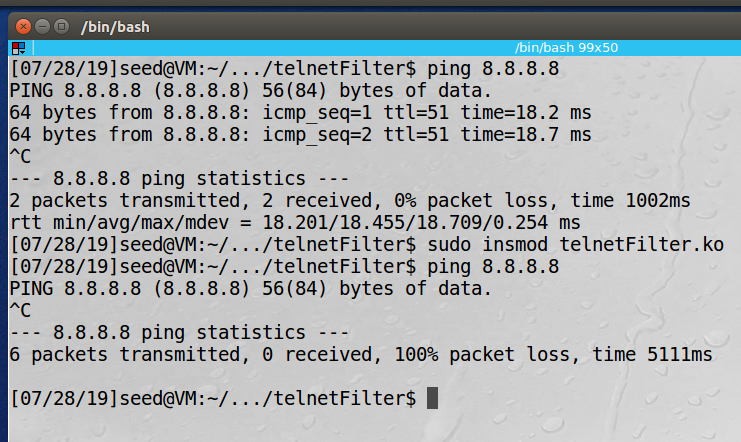
**Explanation:** The UFW program provides an easy way to implement simple packet filtering rules for iptables. Traffic can be blocked between specific hosts, networks or protocols. Root privileges are required to configure rules, enable the firewall or disable the firewall.

# Implementing a Simple Firewall

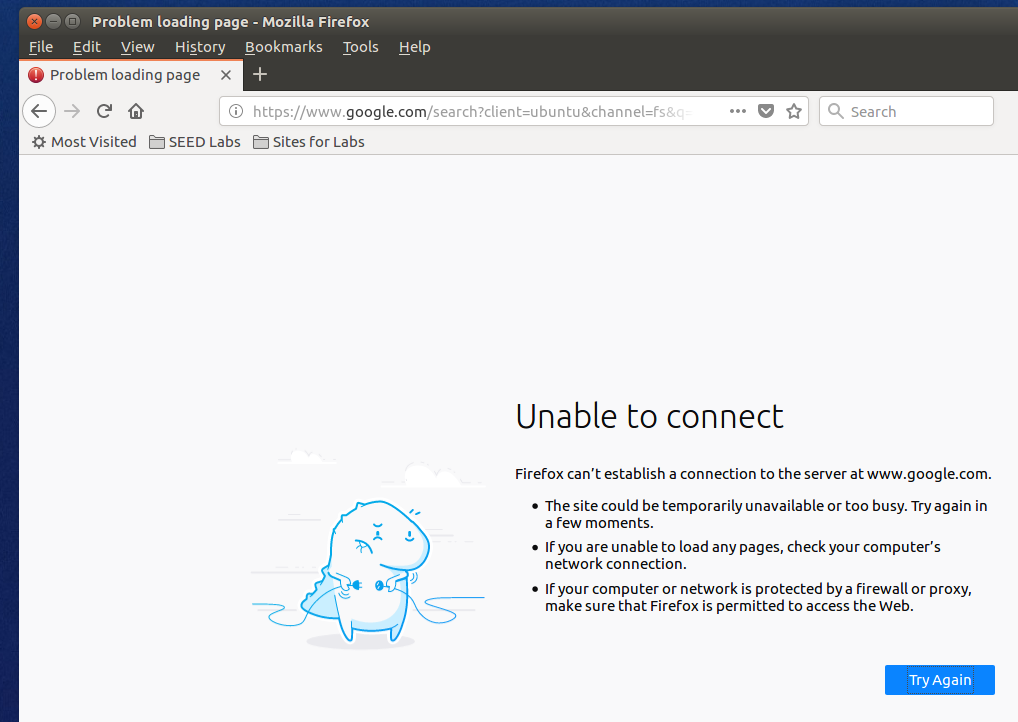
Code for the packet filter, based on the code in the textbook and the sample:



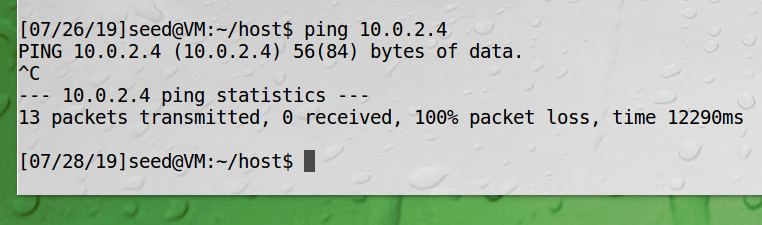
Trying to ping before implementing the filter, implementing the filter, and trying to telnet again:



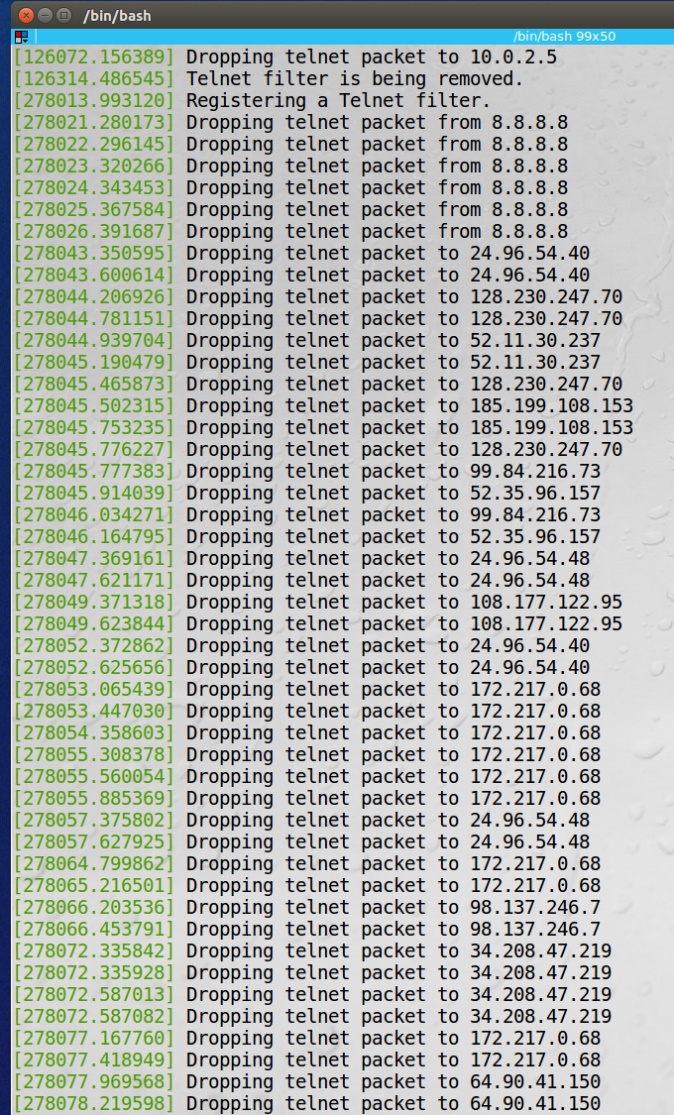
Also, websites no longer load:



Trying to ping from another machine:



Looking at the kernel log buffer (dmesg):



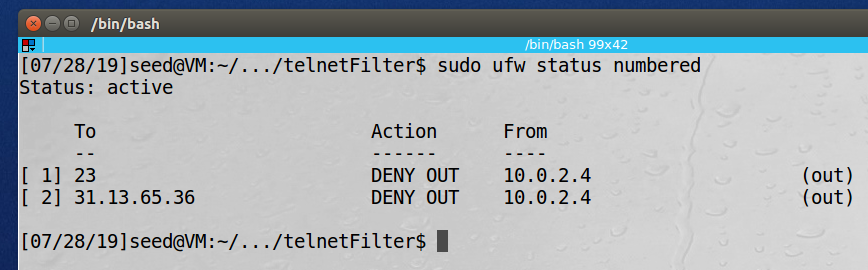
**Observation:** The screenshots above demonstrate the use of the C program to create simple packet filtering firewall. Inside the telnet filter function, we inspect the packet, it if meets one of our blocking criteria we block it, if not we accept it. There are two functions that that will print out the IP address of the packet that was blocked.

**Explanation:** The NETFILTER application provides hooks for installing a packet-filter firewall. We created a firewall and implemented it at the “NF\_IP\_POST\_ROUTING” hook. We were able to inspect, and block packets based on different attributes, like destination IP, port number, or protocol used.

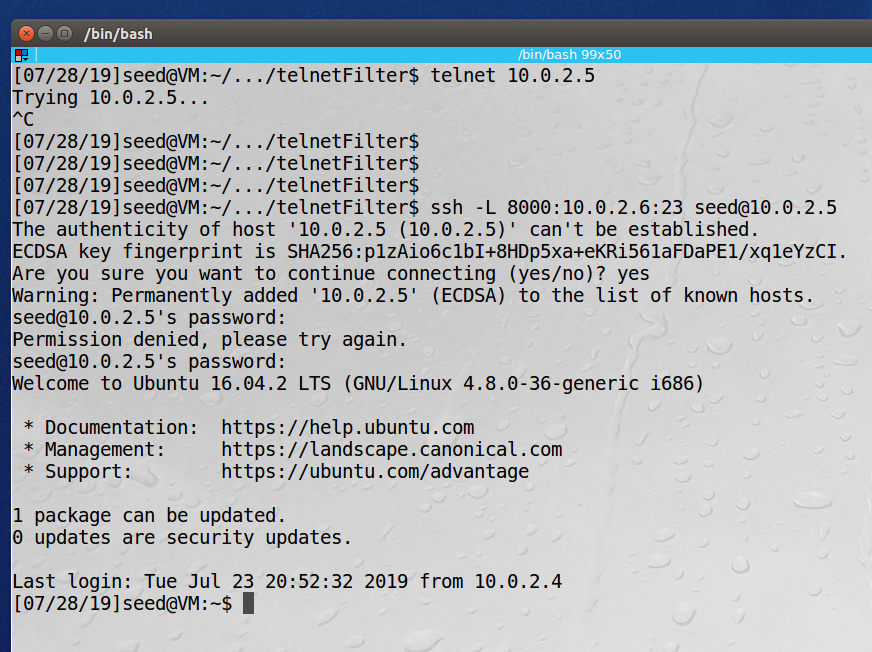
# Evading Egress Filtering

## 3.a Telnet to Machine B through Firewall

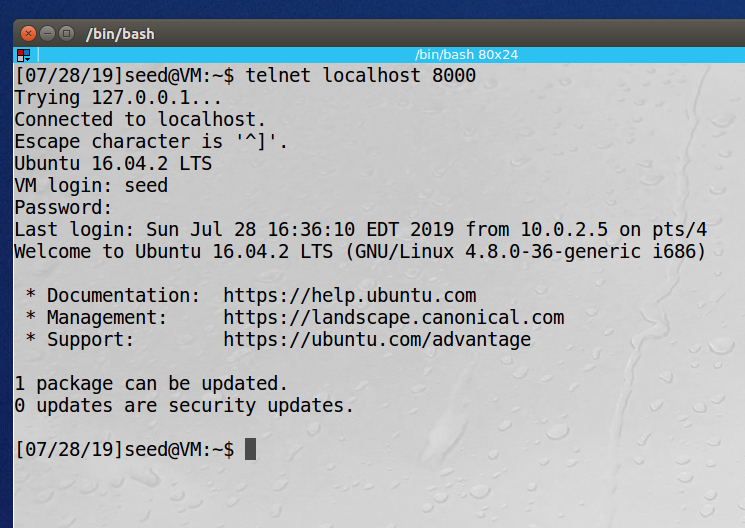
UFW firewall rules:



From machine A, building the SSH Tunnel between machine A and Machine B so that we can eventually TELNET into Machine C:



Opening another Terminal window on Machine A, we can now telnet into machine C by typing the command: telnet localhost 8000:

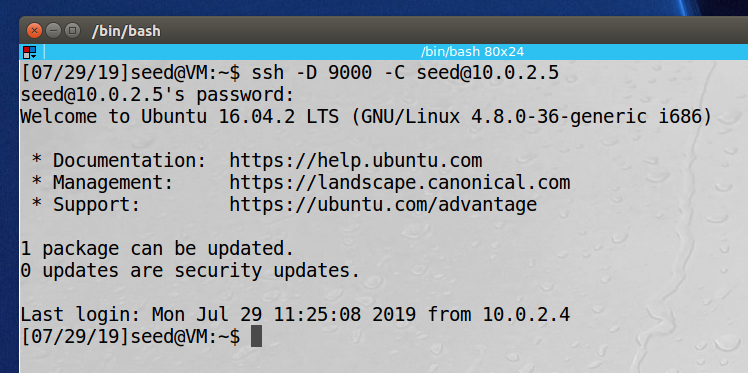


**Observation:** In this task the goal was to TELNET from Machine A to Machine C but there is a firewall on machine A that was blocking all TELNET traffic. To get around the firewall, we established an SSH tunnel from Machine A to Machine B, and then once the connection was established, we could TELNET into machine C. This essentially connects Machine A to Machine C via TELNET but is not blocked by the firewall due to the SSH tunnel.

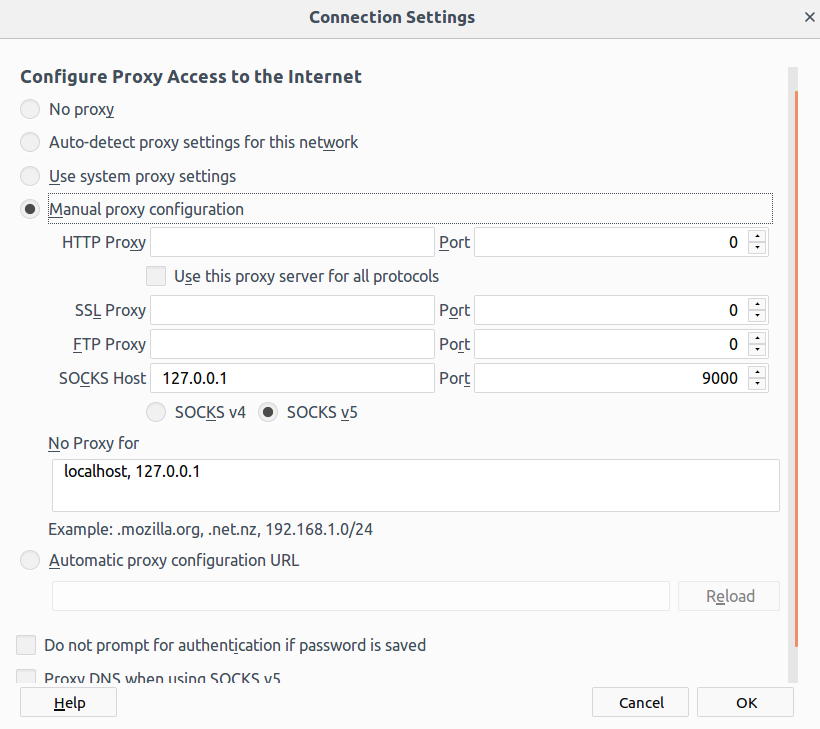
**Explanation:** Since the only thing the firewall sees is an SSH tunnel and cannot examine the encrypted packets inside the tunnel we are able to bypass its telnet restrictions.

## 3.b Connect to Facebook using SSH Tunnel

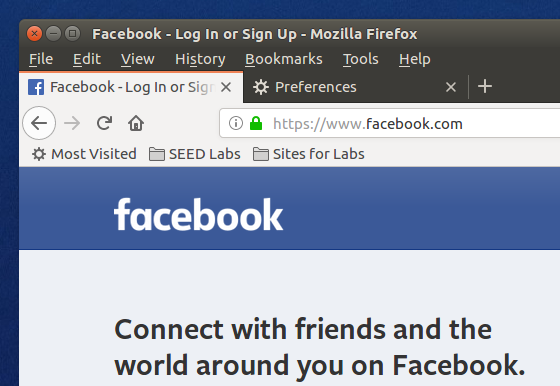
First, we connect Machine A to Machine B via an SSH tunnel:



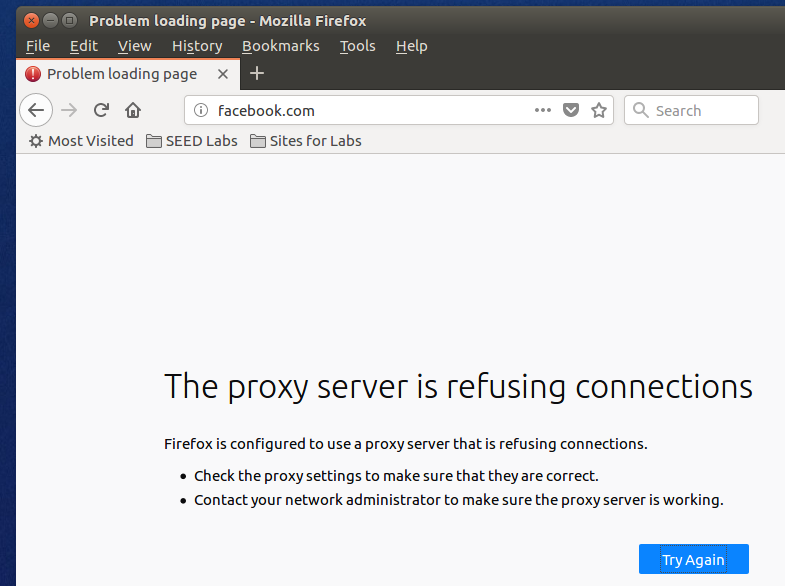
Then we adjust the Firefox browser settings to connect to localhost port 9000 (now Machine B) when connecting to a web server:



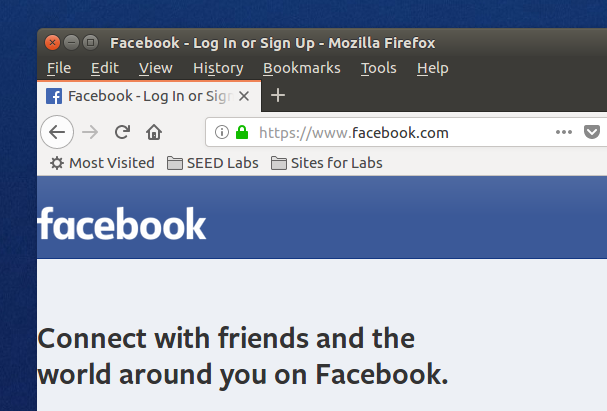
Now we can visit Facebook on Machine A using our SSH tunnel:



Next we break the SSH tunnel, clear the browser cache, and try to visit Facebook again, we receive an error from Firefox that the proxy server is refusing connections:



Upon reestablishing the SSH tunnel we can visit Facebook again:



**Observation:** The goal of this task was to bypass the firewall which is blocking Machine A’s access to the Facebook website. We established an SSH tunnel from Machine A and forward all traffic from localhost port 9000 to Machine B. Next we can adjust our Firefox settings to route all web traffic to localhost port 9000, which is then forwarded to Machine B. Now we can access Facebook on Machine A even though there is a rule in the firewall trying to prevent it.

**Explanation:** Inside the SSH tunnel the traffic is encrypted and the firewall only sees that there is a connection between Machine A and Machine B. By routing all web traffic through this tunnel we can bypass the packet filtering on Machine A.

# Evading Ingress Filtering

The objective of this activity is to access the webserver running on the Work Machine, from the Home Machine. There is a firewall on Work Machine’s network that is blocking Home Machine from accessing Work Machine’s ports 80 (webserver) and 22 (SSH server). We can use a reverse SSH tunnel to bypass this firewall and forwarding the port from the Home Machine to the Work Machine Webserver. For example, let’s say Work Machine’s IP is 123.123.123.123 and we forward port 33333 to the web server. On the Home Machine we will browse to <http://123.123.123.123:33333> and we will be able to access the companies web server even though port 80 is being blocked.