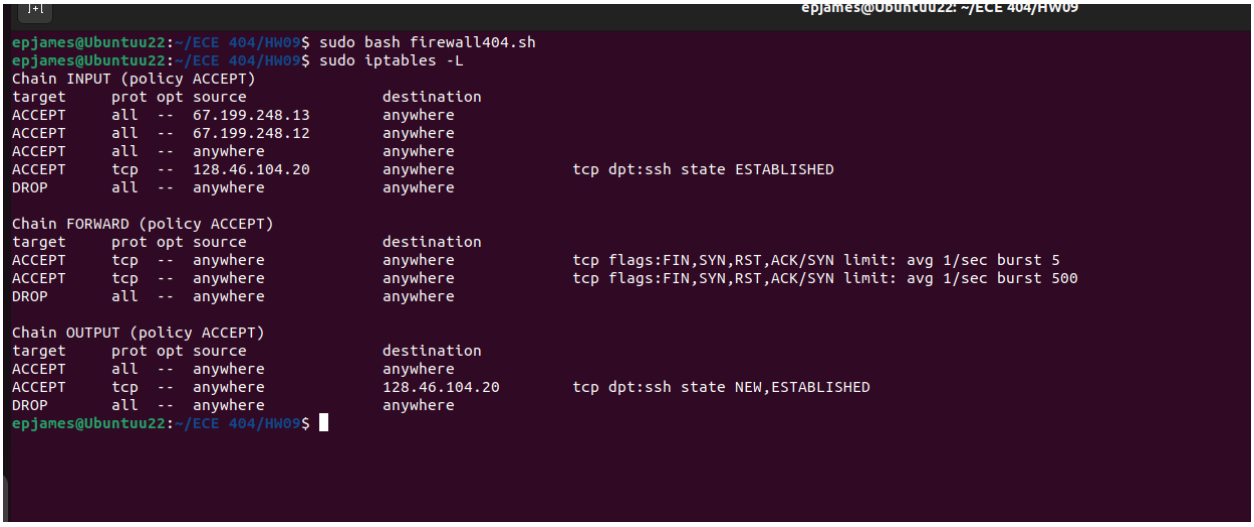
Ethan James
33378430
Epjames

HW09

Output:



Section 1:

```
# 1. Flush and delete all previously defined rules and chains
#Flush previous rules
iptables -t filter -F
iptables -t filter -X
iptables -t mangle -F
iptables -t mangle -X
iptables -t nat    -F
iptables -t nat    -X
iptables -t raw    -F
iptables -t raw    -X
```

The “F” means flush, the “-X” means delete, and the “-t” specifies which table we want. In the first line, we are flushing all the rules in the filter table. In the second line, we are deleting all the chains in the filter table. In the third line, we are flushing all the rules in the mangle table. In the fourth line, we are deleting all user defined chains in the mangle table. In the fifth line, we are flushing all the rules in the nat table. In the sixth line, we are deleting all chains in the nat table. In the seventh line, we are flushing all the rules in the raw table. In the last line, we are deleting the user defined chains from the raw table. I also adapted this code from AVI KAK lecture slides.

Section 2:

```
#*********************************************************************
# 2. Write a rule that only accepts packets that originate from f1.com.
iptables -A INPUT -s f1.com -j ACCEPT
```

The "-A" lets the program know we are going to append. The rule we are adding should be appended to the input chain. The "-s" stands for source. We only want to accept packets from the source f1.com. The "-j Accept" lets the program know that if the rules are satisfied, then allow the packet through.

Section 3:

```
# 3. For all outgoing packets, change their source IP address to your own
# machine's IP address (Hint: Refer to the MASQUERADE target in the
# nat table).
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE #May be eth1
```

The "-t" lets the program know that we want to operate on the nat table. The "-A "indicates that the rule will be appended to the Postrouting chain. The POSTROUTING chain is traversed by packets after they have been routed and before they are sent out of the network interface. The '-o' indicates the outgoing network interface. In this case I specify the interface to be eth0. The "-j" indicates that if the rules before it are satisfied then, the source IP address of outgoing packets should be masqueraded, meaning it should be replaced with the IP address of the machine running the iptables command.

Section 4:

```
#*********************************************************************
# 4. Write a rule to protect yourself against indiscriminate and nonstop
# scanning of ports on your machine.
iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST SYN -m limit --limit 1/s -j ACCEPT #Check to make sure correct
```

The "-A" means the rule will be appended to the Forward chain. The "-p" specifies the protocol which in this case is tcp. The "--tcp-flags SYN,ACK,FIN,RST SYN" matches tcp packets with the SYN flag and any combination of the ACK, FIN, or RST flags unset. The "-m" helps me use the limit module to only allow one packet per second. This protects me from non-stop scanning scanning as we limit the packet rate. The "-j" says that if all these rules are met than accept the packet.

Section 5:

```
# 5. Write a rule to protect yourself from a SYN-flood Attack by limiting
# the number of incoming 'new connection' requests to 1 per second once
# your machine has reached 500 requests.

#NOT SURE ABOUT THIS ONE
iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST SYN -m limit --limit 1/s --limit-burst 500 -j ACCEPT
```

The "-A" means the rule will be appended to the Forward chain. The "-p" specifies the protocol which in this case is tcp. The "--tcp-flags SYN,ACK,FIN,RST SYN" matches tcp packets with the SYN flag and any combination of the ACK, FIN, or RST flags unset. The "-m" helps me use the limit module to only allow one packet per second. The "—limit-burst" lets the machine know that only request the 1 packet per second once the machine has reached 500 requests. If all these rules are met than allow the packet through.

Section 6:

```
# 6. Write a rule to allow full loopback access on your machine i.e. access
# using localhost
# (Hint: You will need two rules, one for the INPUT chain and one the
# OUTPUT chain on the FILTER table. The interface is 'lo'.)

#Loopback for input chain
iptables -A INPUT -i lo -j ACCEPT

#Loopback for output chain
iptables -A OUTPUT -o lo -j ACCEPT
```

In the loopback for the input chain:
We know that the rule will be appended to the input chain based on the "-A" command. The "-i lo" specifies the incoming network interface as loopback interface. This will allow full loopback access on my machine.

In the loopback for output chain:
We know that the rule will be appended to the output chain based on the "-A" command. The "-o lo" specified the outgoing interface as loopback interface. This will allow loopback access on my machine. Both these rules together allow full loopback access.

Section 7:

```
# 7. Write a port forwarding rule that routes all traffic arriving on port
# 8888 to port 25565. Make sure you specify the correct table and chain.
# Subsequently, the target for the rule should be DNAT.
iptables -t nat -A PREROUTING -p tcp --dport 8888 -j DNAT --to-destination 127.0.0.1:25565
```

The “-t” specified the table as nat. We also use the “-A” command to specify the chain the rule will be appended to as Prerouting. We then use the “-p” to get the tcp protocol. The “—dport 8888” matches the packet with the destination port which in this case is specified to be 8888. The “-j DNAT” indicates the target for this rule to be Destination Network Address. Then the last part specifies the new destination address given your IP and the new destination being the port 25565. These rules forward packets from the port 8888 to port 25565.

Section 8:

```
# 8. Write a rule that only allows outgoing ssh connections to
# engineering.purdue.edu. You will need two rules, one for the INPUT
# chain and one for the OUTPUT chain and one the FILTER table. Make
# sure to specify the correct options for the --state suboption for both rules

#Input chains
iptables -A INPUT -p tcp --dport 22 -s 128.46.104.20 -m state --state ESTABLISHED -j ACCEPT

#Ouptut chains
iptables -A OUTPUT -p tcp --dport 22 -d 128.46.104.20 -m state --state NEW,ESTABLISHED -j ACCEPT
```

For the input chain:

The “-A” specifies that rule will be appended to the input chain. The “-p” specifies that the protocol we want is the tcp protocol. The “—dport 22” specifies that the current destination port is port 2. This is the standard port for ssh. The “-s” helps me to specify the source of the incoming packets as we only want packets with the specific ip address given which in this case is the engineering.purdue.edu website. The “-m” portion helps me use the state module to match packets based on their connection state. We want packets that are part of an established connection. If all these rules are met, the packet is accepted and will reach the destination port.

For the output chain:

The “-A” specifies that rule will be appended to the output chain. The “-p” specifies that the protocol we want is the tcp protocol. The “—dport 22” specifies that the current destination port is port 2. This is the standard port for ssh. The “-d” helps me to specify the destination ip of the outgoing packets as we only want packets with the specific ip address given which in this case is the engineering.purdue.edu website. The “-m” portion helps me use the state module to match packets based on their connection state. We want packets that are part of an established connection or a new connection. If all these rules are met, the packet is accepted and will reach the destination port.

Section 9:

```
# 9. Drop any other packets if they are not caught by the above rules.

#Rule for dropping incoming packets that don't satisfy above rules
iptables -A INPUT -j DROP

#Rule for dropping outgoing packets that don't satisfy above rules
iptables -A OUTPUT -j DROP

#RUle for dropping packets that don't satisfy above rules
iptables -A FORWARD -j DROP
```

The first line has “-A” which says that this rule will be appended to the input chain. The “-j Drop” specifies that if the above rules are not met the packet will be dropped.
The second line has “-A” which says that this rule will be appended to the output chain. The “-j Drop” specifies that if the above rules are not met the packet will be dropped.
The first line has “-A” which says that this rule will be appended to the Forward chain. The “-j Drop” specifies that if the above rules are not met the packet will be dropped.