Internet Security – IP Tables Firewall configuration

(Instructions)

1.0 RFC1918 filtering

1.1 RFC1918 filtering of packets with internal addresses as sources but coming through the external interface.

In order to complete this requirement you need to know the reserved internal addresses that you wish to block. This has to be obtained and the command can be a sequence of commands that covers all the addresses one after the other. Use the following command replacing the ipaddr with the IP address you need to block if coming through the external interface.

```
# iptables -A INPUT -s ipaddr –j DROP
```

ipaddr can be a single ip address or a range of ip addresses specified as (eg. 10.0.0.0/24 where 24 says which part of the ip address is significant[1].

1.2 RFC2827 filtering of packets from sources other than internal addresses flowing out of the external interface or packets destined for internal addresses flowing out of the internal interface.

For this process also the internal addresses need to be known. You may make an inversion rule which will eliminate all addresses other than the internal addresses to be blocked if trying to flow out of your subnet.

```
#iptables –A OUTPUT –s ! localhost –j DROP
#iptables –A OUTPUT –d localhost –j DROP
```

In the first of the commands, we say that any packet that comes to the firewall for OUTPUT that has a source address which is not local address has to be dropped. Command 2 is any packet that has the destination address as one of the local reserved addresses, then the packet needs to be dropped and not carried forward[2].

1.3 Webserver access and State full filtering

We need to provide access to the web server only from the specified IP addresses 192.168.*.* .

Step 1: Let us eliminate access from all other addresses other than these.

#iptables –A INPUT –s ! 192.168.0.0/16 –d xxx.xxx.xxx.xxx --dport p –j DROP

Please replace the xxx.xxx.xxx.xxx with your webserver ip address and p with the port number you have configured access in the webserver. Port address is normally 80 for http access and 443 for https. If you want to give access to both add one more --dport switch. The step 1 appends to the list of rules. If the source of the packet is not one of 192.168.0.0 to 192.168.255.255 and if the destination is your webserver, then do not allow the packet[3].

Step 2: We will use stateful filtering for reducing return traffic.

#iptables –A INPUT –m state --state ESTABLISHED, RELATED –j ACCEPT

This will make sure that all input to the firewall server comes from established or related connections alone.

1.4 Allowing target machine to access external webservers

If the IP address of the target machine is xxx.xxx.xxx.xxx, then this would be the command that you might have to give to get access to external webservers through iptables.

#iptables -A OUTPUT -s ! xxx.xxx.xxx.xxx -p tcp --dport www -j DROP

We have already given a stateful filter for return INPUT which has to be established, related to get accepted by the firewall. If you have not given that you might have to append that rule as well along with the ones already entered.

1.5 Access Firewall machine over SSH through internal machines only

#iptables –A INPUT –s ! localhost --dport 22 -j DROP

If it is not local host ip numbers and if the port number is 22 which is the number for SSH, then the packet has to be dropped.

2.0 Port address translation

We need to implement a network address translation that would have any of the target machines within the specified network trying to access the internet. Primarily a many-to-one NATing. This would make sure that for all the machines within the protected network, the firewall would reflect a single IP address. To have NAT to work properly, you need to have iptables_nat module loaded along with the modprobe command[4]. If you have not done it already, also enable the IP forwarding by setting the /proc/sys/net/ipv4/ip_forward to 1 in lieu of the default value of 0.

This piece of code has been taken from http://www.siliconvalleyccie.com/linux-hn/iptables-intro.htm for NATing purposes.

```
#---------------------------------------------------------------
# Load the NAT module
```

```
#

# -----------------------------------------------------------


modprobe iptable_nat

#------------------------------------------------------------

# Enable routing by modifying the ip_forward /proc filesystem file

#

# -----------------------------------------------------------


echo 1 > /proc/sys/net/ipv4/ip_forward


#------------------------------------------------------------

# Allow masquerading

# - Interface eth0 is the internet interface

# - Interface eth1 is the local network interface

#------------------------------------------------------------


iptables -A POSTROUTING -t nat -o eth0 -s 192.168.1.0/24 -d 0/0 \

        -j MASQUERADE


#------------------------------------------------------------

# Prior to masquerading, the packets are routed via the filter

# table's FORWARD chain.
# Allowed outbound: New, established and related connections
# Allowed inbound : Established and related connections
#-----------------------------------------------------------

iptables -A FORWARD -t filter -o eth0 -m state \
        --state NEW,ESTABLISHED,RELATED -j ACCEPT

iptables -A FORWARD -t filter -i eth0 -m state \
        --state ESTABLISHED,RELATED -j ACCEPT
```

This would provide the required NATing; following this, you may conduct the required tests to check out the results.

3.0 Static address translation for Webserver access within the protected network

Our target is to have the multiple accesses from external internet to access the webserver in our protected network through the ip address of the firewall. In order to do this the following steps need to be followed[4].

```
modprobe iptable_nat


#-----------------------------------------------------------
# Enable routing by modifying the ip_forward /proc filesystem file
#
```

```
#---------------------------------------------------------------

echo 1 > /proc/sys/net/ipv4/ip_forward

#---------------------------------------------------------------
# NAT ALL traffic:
###########
# REMEMBER to create aliases for all the internet IP addresses below
###########
#
# TO server:       FROM:          FW server
# xxx.xxx.xxx.xxx Anywhere        yyy.yyy.yyy.yyy (1:1 NAT - Inbound)
#
# PREROUTING:
#   NATs destination IP addresses. Frequently used to NAT
#   connections from the Internet to your home network
#
# - Interface eth0 is the internet interface
# - Interface eth1 is the private network interface
#---------------------------------------------------------------

# PREROUTING statements for 1:1 NAT
# (Connections originating from the Internet)
```

```
iptables -t nat -A PREROUTING -d yyy.yyy.yyy.yyy -i eth0 \
        -j DNAT --to-destination xxx.xxx.xxx.xxx

# POSTROUTING statements for 1:1 NAT
# (Connections originating from the home network servers)

iptables -t nat -A POSTROUTING -s xxx.xxx.xxx.xxx -o eth0 \
        -j SNAT --to-source yyy.yyy.yyy.yyy

# Allow forwarding to each of the servers configured for 1:1 NAT
# (For connections originating from the Internet. Notice how you
# use the real IP addresses here)

iptables -A FORWARD -p tcp -i eth0 -o eth1 -d xxx.xxx.xxx.xxx \
    -m multiport --dport 80,443,22 \
    -m state --state NEW -j ACCEPT


# Allow forwarding for all New and Established SNAT connections
# originating on the home network AND already established
# DNAT connections

iptables -A FORWARD -t filter -o eth0 -m state \
          --state NEW,ESTABLISHED,RELATED -j ACCEPT

# Allow forwarding for all 1:1 NAT connections originating on
# the Internet that have already passed through the NEW forwarding
# statements above

iptables -A FORWARD -t filter -i eth0 -m state \
          --state ESTABLISHED,RELATED -j ACCEPT
```

4.0 Static Address translation for FTP server access within the protected network

In addition to the webserver if you require a similar translation for FTP server in addition to the existing webserver access, the same could be provided following this procedure.

```
modprobe iptable_nat

#---------------------------------------------------------------
# Enable routing by modifying the ip_forward /proc filesystem file
#---------------------------------------------------------------

echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
#------------------------------------------------------------
# NAT ALL traffic:
###########
# REMEMBER to create aliases for all the internet IP addresses below
###########
#
# TO FTP server:     FROM:          internet IP:
# xxx.xxx.xxx.xxx   Anywhere       yyy.yyy.yyy.yyy (1:1 NAT - Inbound)
#
# SNAT is used to NAT all other outbound connections initiated
# from the protected network to appear to come from
# IP address yyy.yyy.yyy.yyy
#
# POSTROUTING:
#   NATs source IP addresses. Frequently used to NAT connections from
#   your home network to the Internet
#
# PREROUTING:
#   NATs destination IP addresses. Frequently used to NAT
#   connections from the Internet to your home network
#
# - Interface eth0 is the internet interface
# - Interface eth1 is the private network interface
```

```
#-------------------------------------------------------------

# PREROUTING statements for 1:1 NAT for ftp
# (Connections originating from the Internet)

iptables -t nat -A PREROUTING -p ftp -d yyy.yyy.yyy.yyy -i eth0 \
    -j DNAT --to-destination xxx.xxx.xxx.xxx

# POSTROUTING statements for 1:1 NAT
# (Connections originating from the home network servers)

iptables -t nat -A POSTROUTING -p ftp -s xxx.xxx.xxx.xxx -o eth0 \
    -j SNAT --to-source yyy.yyy.yyy.yyy


iptables -A FORWARD -p tcp -i eth0 -o eth1 -d yyy.yyy.yyy.yyy \
  -m multiport --dport 80,443,22 \
  -m state --state NEW -j ACCEPT



iptables -A FORWARD -t filter -o eth0 -m state \
    --state NEW,ESTABLISHED,RELATED -j ACCEPT
```

```
iptables -A FORWARD -t filter -i eth0 -m state \
    --state ESTABLISHED,RELATED -j ACCEPT
```

4.1 Other minor rules

Here is the rule to deactivate the icmp reply echo.

```
# Allow pings, but reject the rest
$iptables -A INPUT -i eth0 -p icmp -j ACCEPT
$iptables -A INPUT -i eth0 -j REJECT
$iptables -A INPUT -p udp -i $INET_IFACE -j DROP
$iptables -A INPUT -p icmp -i $INET_IFACE -j DROP
```

These are for blocking bad udp and icmp packets.

Bibliography

1. Y.Rekhter, B.Moskovitz, D.Karrenberg, G.J.DeGroot & E.Lear, 1996, *Address Allocation for Private Internets,* Network Working Group, available at http://www.ietf.org/rfc/rfc1918.txt

2. P.Ferguson & D.Senie, 2000, Network Ingress Filtering: *Defeating Denial of Service Attacks which employ IP Source Address Spoofing*, Network Working Group, available at http://www.ietf.org/rfc/rfc2827.txt

3. Peter Harrison, *Linux Firewalls Using iptables*, Linux Home Networking, available at http://www.siliconvalleyccie.com/linux-hn/iptables-intro.htm

4. Beret, 2004, iptables: Creating an open source firewall, Enterprise Linux Resource, available at http://tips.linux.com/print.pl?sid=04/11/23/2022252

Internet Security – IP Tables Firewall configuration

(Report)

1.0 Network Address Translation

Every packet header comes with the source and destination address. On receiving this at the firewall, this is routed to the destination IP address. Most of the local intranet has addresses that are defined by RFC1918 rules for private networks. When we access from these local nodes to an external internet IP, then the local address needs to be translated to an acceptable internet IP address. Similarly when the packet comes from the external internet addresses, it needs to be translated and distributed to the node that requested for the packet or to which the packet is destined. This is called Network Address Translation or NAT.

2.0 Masquerading

Masquerading happens when the firewall computer 'acts' as the source IP address for all the packets sent out from the network. This is also called as a Many to One NAT-ting. All computers in the intranet reflect the same IP address of the firewall to the external internet. When you configure iptables to DHCP, it takes the firewall's ip address as the IP address to be reflected in case of all OUTPUT NAT-ting.

Masquerade can be used as follows:

```
iptables -A POSTROUTING -t nat -o eth0 -s 192.168.1.0/24 -d 0/0 \
        -j MASQUERADE
```

The reverse of the masquerade explanation given above should also be true. Once you configure your firewall to masquerade, then all the machines inside should use the firewall's IP address as their gateway address.

3.0 NAT and FORWARD

Port forwarding is used when we require forwarding an INPUT packet to a specific web server within our network. Destination ip need to be NAT-ted using the PREROUTING and then using the FORWARD-ing chain to forward to the required web server's internal IP address.

A typical rule list is given below[2]:

```
#-------------------------------------------------------------
# Load the NAT module
#-------------------------------------------------------------

modprobe iptable_nat

#-------------------------------------------------------------
# Get the IP address of the Internet interface eth0
#
# This is best when your firewall gets its IP address using DHCP.
# The external IP address could just be hard coded ("typed in
# normally")
#-------------------------------------------------------------

external_int="eth0"
external_ip="`ifconfig $external_int | grep 'inet addr' | \
                awk '{print $2}' | sed -e 's/.*://'`"

#-------------------------------------------------------------
# Enable routing by modifying the ip_forward /proc filesystem file
#
#-------------------------------------------------------------

echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
#------------------------------------------------------------
# Allow port forwarding for traffic destined to port 80 of the
# firewall's IP address to be forwarded to port 8080 on server
# xxx.xxx.xxx.xxx
#
# - Interface eth0 is the internet interface
# - Interface eth1 is the private network interface
#------------------------------------------------------------

iptables -t nat -A PREROUTING -p tcp -i eth0 -d $external_ip \
    --dport 80 --sport 1024:65535 -j DNAT --to xxx.xxx.xxx.xxx:8080

#------------------------------------------------------------
# After DNAT, the packets are routed via the filter table's
# FORWARD chain.
# Connections on port 80 to the target machine on the private
# network must be allowed.
#------------------------------------------------------------

iptables -A FORWARD -p tcp -i eth0 -o eth1 -d xxx.xxx.xxx.xxx \
   --dport 8080 --sport 1024:65535 -m state --state NEW -j ACCEPT

iptables -A FORWARD -t filter -o eth0 -m state \
       --state NEW,ESTABLISHED,RELATED -j ACCEPT

iptables -A FORWARD -t filter -i eth0 -m state \
       --state ESTABLISHED,RELATED -j ACCEPT
```

The xxx.xxx.xxx.xxx is the web server IP address inside the network. Here all INPUT to the firewall if it is NEW, ESTABLISHED or RELATED are forwarded to the web server IP address. All the OUTPUT only if ESTABLISHED or RELATED are forwarded to the required address.

4.0 Network Ranges and NAT

A typical requirement of a many to one NAT is to assign a range or the entire intranet to a specific IP address as the source. This is achieved in a NAT by using the following POSTROUTING statements[2].

```
# POSTROUTING statements for Many:1 NAT
# (Connections originating from the entire home network)

iptables -t nat -A POSTROUTING -s 192.168.1.0/24 \
         -j SNAT -o eth0 --to-source 97.158.253.29
```

Network range specification that we give here, as 24 mean 24 bits on the LSB to be varied from 0 to the maximum. When we say 24, the last three units are varied from 0 to 255.

5.0 Tripwire installation and operation

To get your tripwire installation going, you need to first set your system for running the same. You should be the supervisor or root to do the job. You might also have to set the LOOSEDIRECTORYCHECKING to true in the configuration file before commencing the installation process. You need to set two passwords or two key pass phrases which will be asked during the installation process.

You need to run the install script (./twinstall.sh). Once the installation is completed, you need to initialize the database[3].

tripwire --init

In case you face errors while running this command, you may run it under verbose mode by adding the -v switch to the command given above.

Installing of tripwire alone does not start off any protection measure for your network. Periodic checks need to be run using :

tripwire --check

This needs to be run repeatedly to know the status.

6.0 Snort

Snort needs to be downloaded and then installation process may be started[4]. Snort is an intrusion deduction system (IDS) while tripwire is an activity monitor. As per the topology that snort advocates, the IDS may be put either before the firewall or at NAT. If it is put ahead of the firewall, we will be able to monitor even those attempts which failed. If we monitor at the NAT point, we will be able to get the source IP address of the packet, enabling us to trace the origin of the attempt. Possibly both the sensors together would be better choice.

Apart from the Apache Web Server and MySQL database, snort would require Webmin and ACID (Analysis Console for Intrusion Detection). If the system is already set for installing snort, then we need to start snort using

```
# /etc/rc.d/init.d/snortd start
```

Snort may be stopped by issuing the same command with stop in place of start. If the server is not set for the operation then configurations need to be altered to put these in place as in the reference 4 of the bibliography.

Bibliography

5. Beret, 2004, iptables: Creating an open source firewall, Enterprise Linux Resource, available at http://tips.linux.com/print.pl?sid=04/11/23/2022252

6. Peter Harrison, *Linux Firewalls Using iptables*, Linux Home Networking, available at http://www.siliconvalleyccie.com/linux-hn/iptables-intro.htm
7. Meryll Larkin, 2003, How To Linux - Setting Tripwire, Alwanza: How To Linux, available at http://www.alwanza.com/howto/linux/tripwire.html
8. Steven. J. Scott, 2002, *Snort Installation Manual*, available at http://home.earthlink.net/~sjscott007/