Lab-6

Configuring and Securing Central Log Server

Mohamed Yasser Kaleelurrahman

17th November, 2016

Seneca College.

Contents

Lab-6

Objective

Configure the rsyslog server and forward all the log messages to the central log server and investigate the traffic between the log servers and the central log server. Another objective of this lab is to implement an access policy to the central log server using netfilters/iptables.

Log server Configuration

VM Name	c7min-log
Hostname	loghost.mykaleelurrahman.net, loghost
Static IP	192.168.34.51
Syslog version	rsyslog-7.4.7-7.el7_0.x86_64
OS	Centos 7 Minimal

In order to open the port for the log server on port 514 I configured the rsyslog.conf file and edited the lines

```
ModLoad imudp
UDPServerRun 514
```

I uncommented these lines to accept udp traffic and run rsyslog on the port 514. Then I restarted the rsyslog service.

```
Systemctl restart rsyslog.service
```

VM's configuration

Now that our central log server is configured, our next objective is to forward certain logs to our central log server. In order to do that we need to be able to resolve the hostname of the logserver, we can edit our /etc/hosts file and give the IP a name to be resolved. Here we are giving it loghost.

In order to send only selected log messages from the VM, I edited the /etc/rsyslog file and added these lines.

```
*.info;mail.none;news.none;cron.none        @loghost
Authpriv.*                                  @loghost
```

The above lines forward the selected logs to the log host.

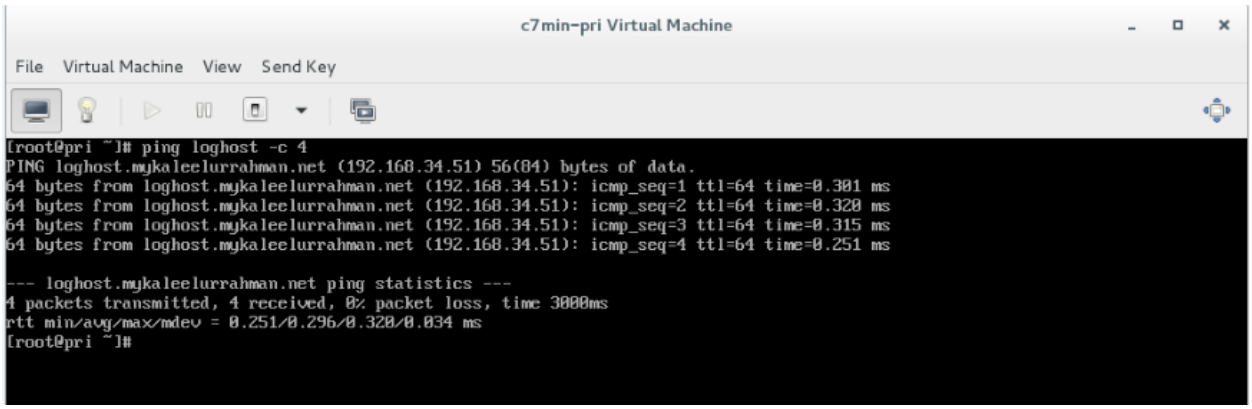
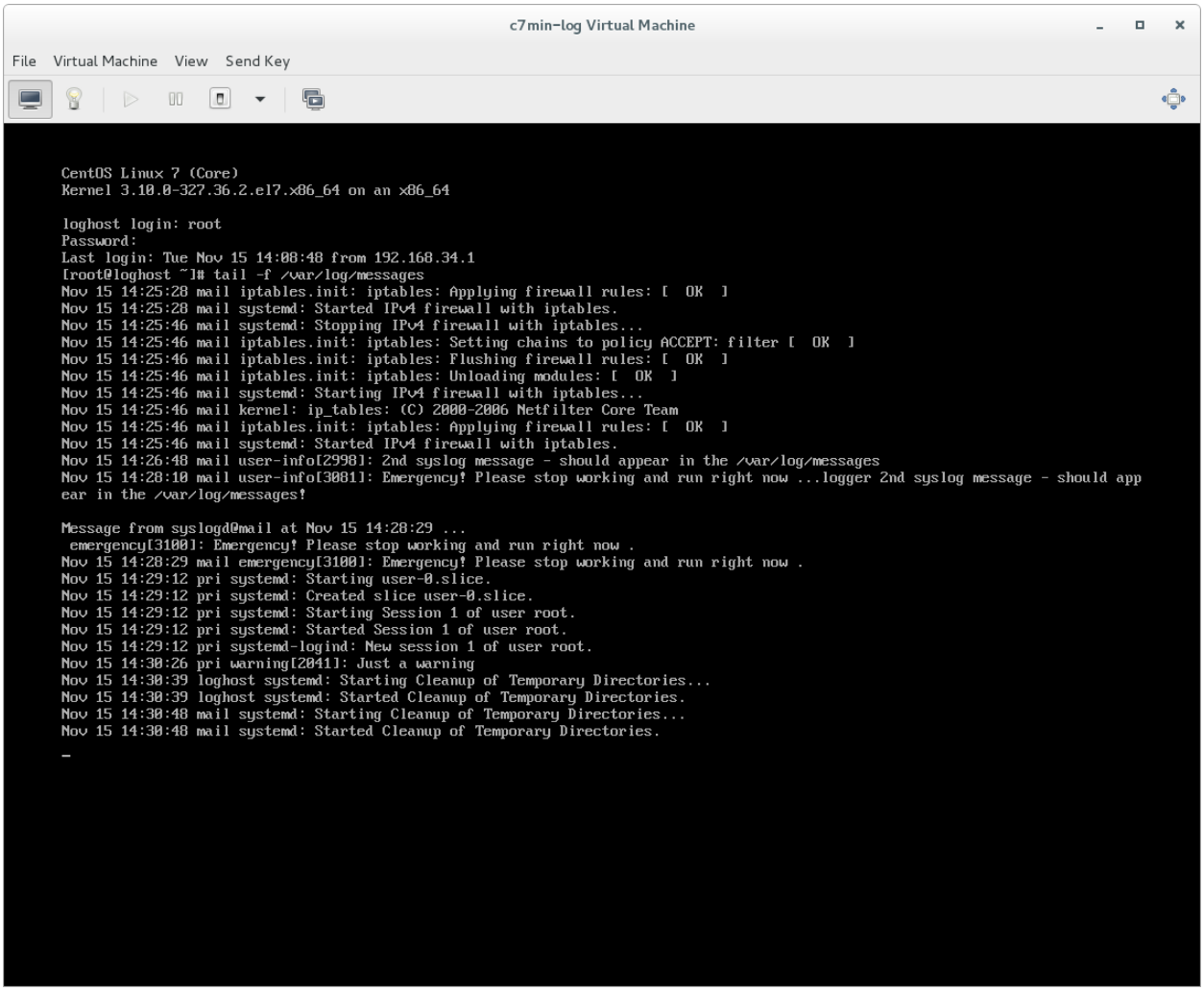


Fig.0 shows that loghost can be resolved.

Preliminary Tests

Log server accepting log messages from mail and primary



Lab-6

Fig 1 Log host (log messages from mail and primary server)

Selinux Status

```
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      28
```

Mail-Firewall

```
# Generated by iptables-save v1.4.21 on Wed Nov 16 17:27:07 2016
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT DROP [2:142]
-A INPUT -s 192.168.34.153/32 -p udp -m udp --sport 53 -j ACCEPT
-A INPUT -s 192.168.34.153/32 -p tcp -m tcp --sport 53 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 25 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 25 -j ACCEPT
-A INPUT -s 172.16.99.1/32 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -s 192.168.99.0/24 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -s 192.168.34.0/24 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j LOG --log-prefix "Rejected SSH
Packets: "
-A INPUT -p tcp -m tcp --dport 22 -j REJECT --reject-with icmp-port-
unreachable
-A INPUT -j LOG --log-prefix " Rejected Network Packets: "
-A INPUT -j REJECT --reject-with icmp-port-unreachable
-A OUTPUT -p udp -m udp --dport 514 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 25 -j ACCEPT
-A OUTPUT -p tcp -m tcp --sport 22 -j ACCEPT
COMMIT
# Completed on Wed Nov 16 17:27:07 2016
```

Access Policy (INPUT CHAIN: Default: DROP)

Protocol	Source IP	Destination IP	Destination Port	Policy
TCP	172.16.99.1	*	22	ACCEPT
TCP	192.168.34.0/24	*	22	ACCEPT
TCP	192.168.99.0/24	*	22	ACCEPT
UDP	192.168.34.0/24	192.168.34.51	514	ACCEPT
ICMP	*	*	*	ACCEPT
TCP	*	*	22	LOG
TCP	*	*	22	REJECT
*	*	*	*	LOG
*	*	*	*	REJECT

Access Policy (OUTPUT CHAIN: Default: DROP)

Protocol	Source IP	Destination IP	Source Port	Destination Port	Policy
ICMP	*	*	*	*	ACCEPT
TCP	192.168.34.51	*	*	*	ACCEPT

Ssh root@192.168.34.51 "iptables-save" > log-iptables

```
# Generated by iptables-save v1.4.21 on Wed Nov 16 15:43:10 2016
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [34:3527]
-A INPUT -p udp -m state --state INVALID -j DROP
-A INPUT -s 192.168.34.0/24 -d 192.168.34.51/32 -p udp -m udp --dport
514 -j ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
-A INPUT -s 172.16.99.1/32 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -s 192.168.34.0/24 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -s 192.168.99.0/24 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j LOG --log-prefix "SSH Rejected
Packets: "
-A INPUT -p tcp -m tcp --dport 22 -j REJECT --reject-with icmp-port-
unreachable
-A INPUT -j LOG --log-prefix "Network Packets Rejected: "
-A INPUT -j REJECT --reject-with icmp-port-unreachable
-A OUTPUT -p icmp -m icmp --icmp-type 0 -j ACCEPT
```

```
-A OUTPUT -s 192.168.34.51/32 -p tcp -m tcp --sport 22 -j ACCEPT
COMMIT
# Completed on Wed Nov 16 15:43:10 2016
```

Test Case Scenario

Test case #1: SSH

Description:

To allow SSH connections from the Private network IP and the Lab server network only.

IPtable rule:

```
iptables -A INPUT -s 172.16.99.1/32 -p tcp -m tcp --dport 22 -j ACCEPT
iptables -A INPUT -s 192.168.34.0/24 -p tcp -m tcp --dport 22 -j
ACCEPT
iptables -A INPUT -s 192.168.99.0/24 -p tcp -m tcp --dport 22 -j
ACCEPT
```

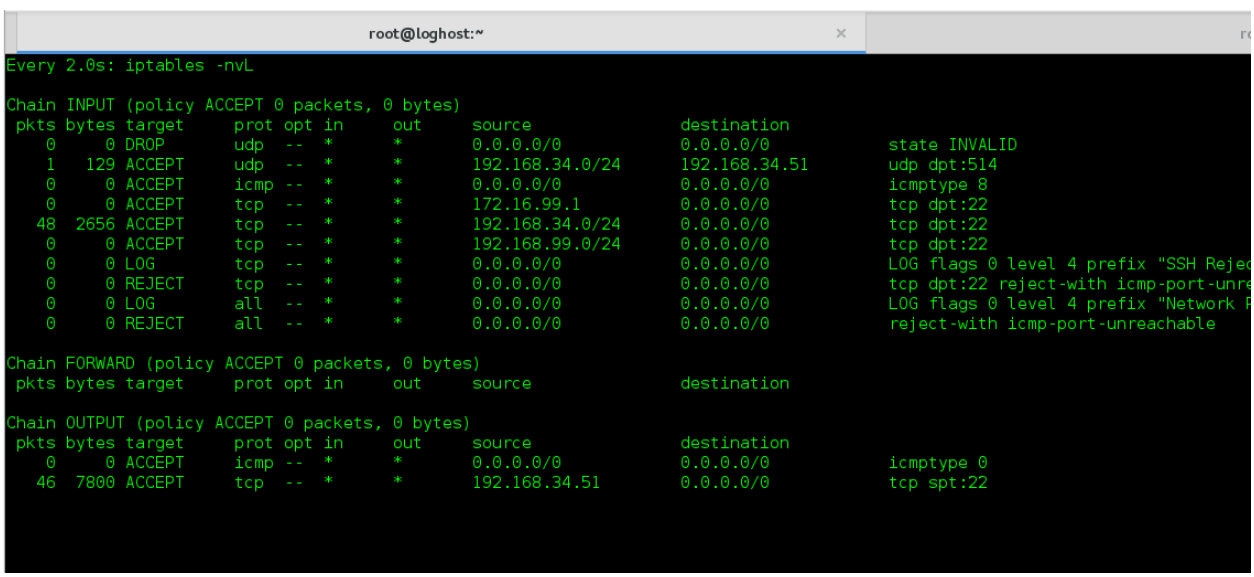
Implementation

I dropped all the invalid packets at the beginning and accepted all other packets which are not invalid.

Design

In order to validate this rule I watched the iptables while I tried to establish a ssh connection from my local network IP.

Proof of validation



```
root@loghost:~
Every 2.0s: iptables -nvL

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            state INVALID
    1   129 ACCEPT     udp  --  *      *       192.168.34.0/24      192.168.34.51        udp dpt:514
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0            icmptype 8
    0     0 ACCEPT     tcp  --  *      *       172.16.99.1          0.0.0.0/0            tcp dpt:22
   48  2656 ACCEPT     tcp  --  *      *       192.168.34.0/24      0.0.0.0/0            tcp dpt:22
    0     0 ACCEPT     tcp  --  *      *       192.168.99.0/24      0.0.0.0/0            tcp dpt:22
    0     0 LOG        tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            LOG flags 0 level 4 prefix "SSH Rejec
    0     0 REJECT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:22 reject-with icmp-port-unre
    0     0 LOG        all  --  *      *       0.0.0.0/0            0.0.0.0/0            LOG flags 0 level 4 prefix "Network R
    0     0 REJECT     all  --  *      *       0.0.0.0/0            0.0.0.0/0            reject-with icmp-port-unreachable

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0            icmptype 0
   46  7800 ACCEPT     tcp  --  *      *       192.168.34.51        0.0.0.0/0            tcp spt:22
```

Fig 1.1 Shows packets are being accepted by the applied rule

Expected result: Populate the rule applied.

Result: Pass

Test case #2: Log Traffic

Description:

To accept all selected log traffic which has been forwarded to the localhost by the configured VM's. It should accept traffic only from the my private network 192.168.34.0/24 and be destined to 192.168.34.51

IPtable rule:

```
iptables -A INPUT -s 192.168.34.0/24 -d 192.168.34.51/32 -p udp -m udp
--dport 514 -j ACCEPT
```

Implementation

I forwarded the log messages from the other VM and configured the Central log server to accept incoming log traffic in the config file as well as the firewall.

Design

In order to validate this rule, I watched the iptables as well as the log messages on the central log server and send some log messages using the logger command as well as the system log itself.

Proof of validation

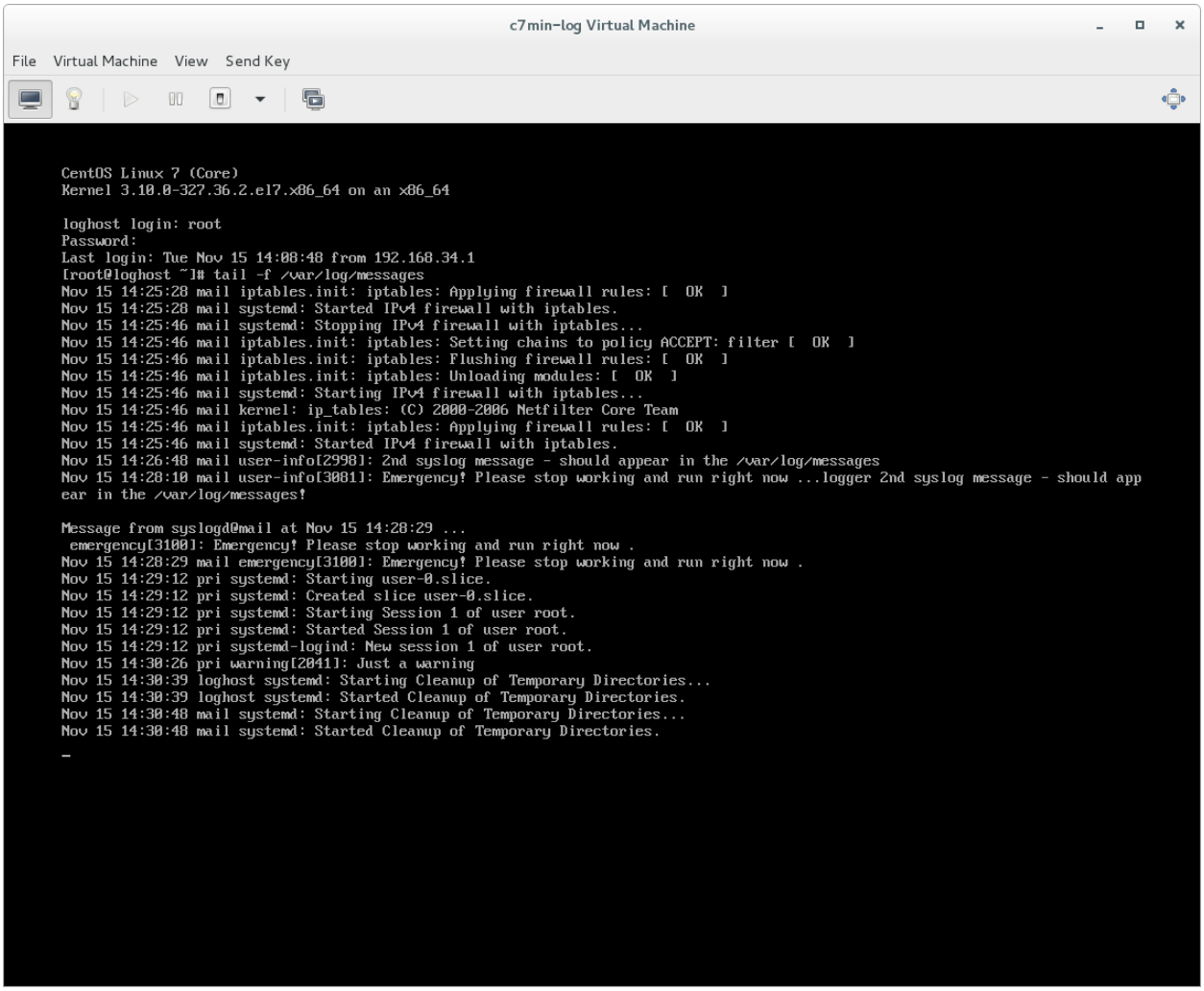


Fig 2.1. Shows the log messages from primary and mail server (It uses the hostname to identify)

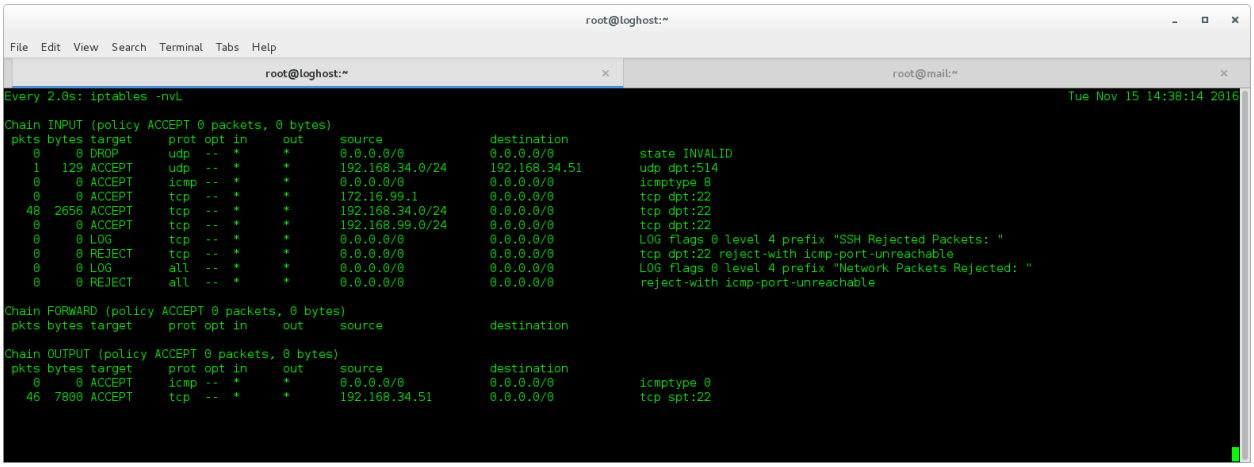


Fig2.2. Shows the rule allowing port 514 (Log) getting populated

Expected result: Populate the rule applied.

Result: Pass

Test case #3: SSH Log and Reject

Description:

The reason to implement this rule is to log any users who try to ssh into the server without any permission or access. We have to make sure we know who has tried to illegitimately access the log server.

IPtable rule:

```
iptables -A INPUT -p tcp -m tcp --dport 22 -j LOG --log-prefix "SSH
Rejected Packets: "
iptables -A INPUT -p tcp -m tcp --dport 22 -j REJECT --reject-with
icmp-port-unreachable
```

Implementation

Allowed only the network IP which has an access to the log server at the beginning and then logged all other ssh access and rejected them.

Design

To validate these rules, I watched the log file and also the iptables while trying to access the server from another network which does not have the access grant.

Proof of validation

Lab-6

```
    172.16.40.1.39067 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x2734 (correct), seq 2491559967, win 14600, options [mss
 1460,sackOK,TS val 906100 ecr 0,nop,wscale 7], length 0
    172.16.40.1.39067 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x1f60 (correct), seq 2491559967, win 14600, options [mss
 1460,sackOK,TS val 908104 ecr 0,nop,wscale 7], length 0
    172.16.40.1.39067 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x0fb8 (correct), seq 2491559967, win 14600, options [mss
 1460,sackOK,TS val 912112 ecr 0,nop,wscale 7], length 0
    172.16.40.1.39067 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0xf067 (correct), seq 2491559967, win 14600, options [mss
 1460,sackOK,TS val 920128 ecr 0,nop,wscale 7], length 0
    172.16.40.1.39067 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0xb1c7 (correct), seq 2491559967, win 14600, options [mss
 1460,sackOK,TS val 936160 ecr 0,nop,wscale 7], length 0
    172.16.40.1.39067 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x34a7 (correct), seq 2491559967, win 14600, options [mss
 1460,sackOK,TS val 968192 ecr 0,nop,wscale 7], length 0
    192.168.40.153.60281 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x4099 (correct), seq 3109792996, win 14600, options [
mss 1460,sackOK,TS val 6336527 ecr 0,nop,wscale 7], length 0
    192.168.40.153.60281 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x3cb0 (correct), seq 3109792996, win 14600, options [
mss 1460,sackOK,TS val 6337528 ecr 0,nop,wscale 7], length 0
    192.168.40.153.60281 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x34dc (correct), seq 3109792996, win 14600, options [
mss 1460,sackOK,TS val 6339532 ecr 0,nop,wscale 7], length 0
    192.168.40.153.60281 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x2538 (correct), seq 3109792996, win 14600, options [
mss 1460,sackOK,TS val 6343536 ecr 0,nop,wscale 7], length 0
    192.168.40.153.60281 > pri.mykaleelurrahman.net.ssh: Flags [S], cksum 0x05e8 (correct), seq 3109792996, win 14600, options [
mss 1460,sackOK,TS val 6351552 ecr 0,nop,wscale 7], length 0
```

Fig3.1 shows the dump of ssh packets rejected

```
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0           0.0.0.0/0           icmptype 8 state NEW,RELATED,ESTABLISHE
D
  138 21640 ACCEPT     tcp  --  *      *       192.168.34.0/24     0.0.0.0/0           tcp dpt:22 state NEW,ESTABLISHED
    0     0 ACCEPT     tcp  --  *      *       172.16.34.1         0.0.0.0/0           tcp dpt:22 state NEW,ESTABLISHED
    0     0 ACCEPT     tcp  --  *      *       172.16.99.1         0.0.0.0/0           tcp dpt:22 state NEW,ESTABLISHED
   13   780 LOG        tcp  --  *      *       0.0.0.0/0           0.0.0.0/0           tcp dpt:22 LOG flags 0 level 4 prefix "
SSH REJECTED: "
   13   780 REJECT     tcp  --  *      *       0.0.0.0/0           0.0.0.0/0           tcp dpt:22 reject-with icmp-port-unreac
hable
```

Fig3.2 shows the logged and rejected packets in the iptables

Expected result: Log the action while trying to ssh from a different server

Result: Pass

Test case #4: Network Access Reject and Log

Description:

The reason to implement this rule is to log any other attempts to gain access to the server either through ftp, web, or any other access should be logged and rejected.

IPtable rule:

```
iptables -A INPUT -j LOG --log-prefix "Network Packets Rejected: "
iptables -A INPUT -j REJECT --reject-with icmp-port-unreachable
```

Implementation

In order to block all other traffic, I accepted the traffic needed for this server to run smoothly and access it remotely by the authorized users and denied all other access to the server.

Design

To validate this rule I tried to telnet to the server and watched the log messages and the iptables while trying to get access through telnet.

Proof of validation

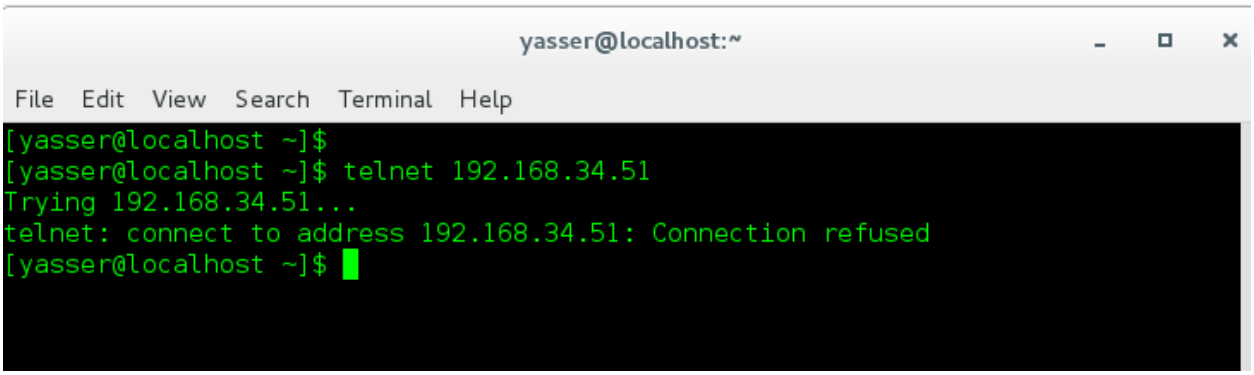


Fig.4.1 Shows that telnet is getting rejected

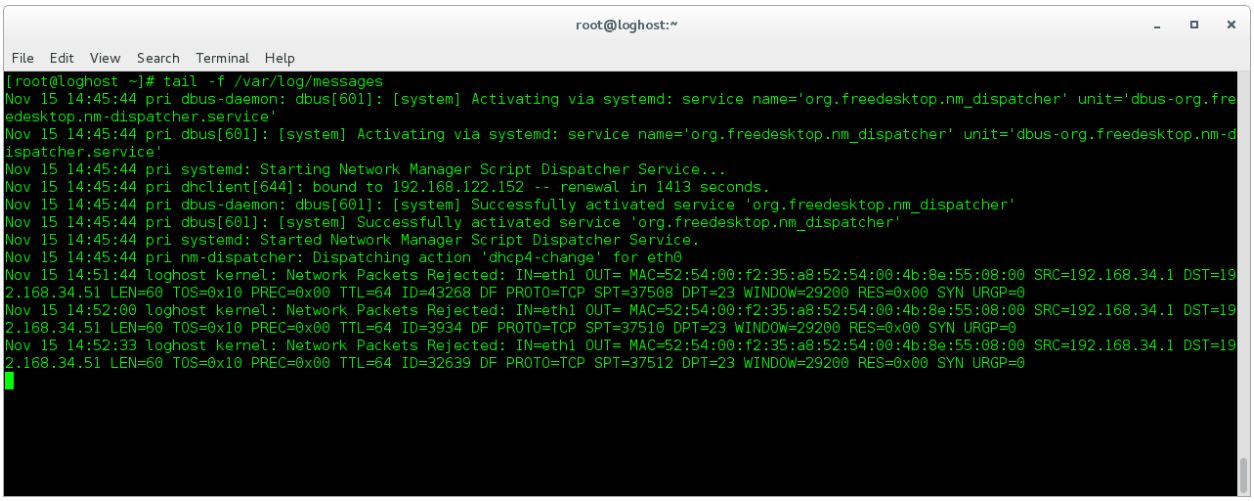


Fig4.2 shows the rejected telnet log traffic

Test Case #5 – Mail server Log rule

Description

In the primary and the mail server we had to allow outgoing logs to the destined central log server, inorder to do that I configured it to forward the logs to the loghost in the rsysconf file. I also implemented an output policy for the log and mail to send only to the logserver.

IPtable rule

```
iptables -A OUTPUT -p udp -m udp -d 192.168.34.51 --dport 514 -j
ACCEPT
```

Implementation

To implement this rule, I made sure the log server accepted the logs before implementing this rule and then dropped the output chain by default and implemented this rule.

Design

To validate this rule, I watched the iptables rule while I send the log requests using logger command from the mail and primary. For proof I have shown only from the mail because In Fig1.1 I have shown from both the mail and primary was being accepted in the log server.

Proof of validation

Lab-6

```
root@loghost:~                                                                    root
Every 2.0s: iptables -nvL

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     udp  --  *      *       192.168.34.153       0.0.0.0/0            udp spt:53
    0     0 ACCEPT     tcp  --  *      *       192.168.34.153       0.0.0.0/0            tcp spt:53
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:25
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp spt:25
    0     0 ACCEPT     tcp  --  *      *       172.16.99.1          0.0.0.0/0            tcp dpt:22
    0     0 ACCEPT     tcp  --  *      *       192.168.99.0/24      0.0.0.0/0            tcp dpt:22
 1005 52576 ACCEPT     tcp  --  *      *       192.168.34.0/24      0.0.0.0/0            tcp dpt:22
    0     0 LOG        tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:22 LOG flags 0 level 4 prefix "
    0     0 REJECT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:22 reject-with icmp-port-unreac
   61 23889 LOG        all  --  *      *       0.0.0.0/0            0.0.0.0/0            LOG flags 0 level 4 prefix " Rejected N
   61 23889 REJECT     all  --  *      *       0.0.0.0/0            0.0.0.0/0            reject-with icmp-port-unreachable

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 115 packets, 21656 bytes)
 pkts bytes target     prot opt in     out     source               destination
  113 23270 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:514
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:25
 1003  140K ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp spt:22
```

Fig5.1 shows that packets are being accepted in the output chain at port 514

Keyless SSH login

For this lab, we have disabled password authentication for accessing the logserver. It allows only Public Key authentication. In order to achive this I configured the file /etc/ssh/sshd_config and edited the line

PasswordAuthentication no

This line will disable ssh login using password. Before implementing I installed my public keys to the log server from where I needed access from. In order to do this, I created a ssh-keygen on the host and then copied the ID to the loghost using the command ***ssh-copy-id root@192.168.34.51.***

Proof

```
[root@pri ~]# ssh root@192.168.34.51
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[root@pri ~]# _
```

Fig 6.1 shows permission denied to ssh to server

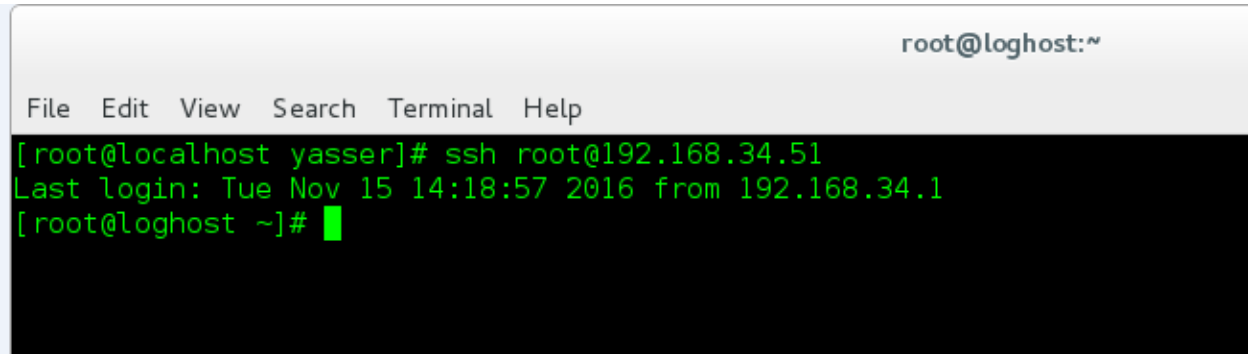


Fig6.2 shows ssh has been successful with public key

Conclusion

This lab has helped us to learn how to configure the central log server and forward selective logs to the central log server from other servers and also secure the server with the help of net filters and iptables.