System Configuration & Management Part 2 Objectives

Tasks in PDF file

System Configuration and Management(Part2)

1. On server machine, forward the http traffic coming on **port 444** from **client.example.com** to port 80 on same machine.

Commands: (On server.example.com)

`firewall-cmd --list-all` **(To list all firewall configurations)**

To add rich rule to forward http traffic coming on port 444 from client.example.com to port 80 on server.example.com:

```
firewall-cmd --add-rich-rule 'rule family="ipv4" source address="192.168.122.20" forward-port to-port="80" protocol="tcp" port="444"' --permanent
```

`firewall-cmd --reload` **(To reload the firewall to make the changes done in above command to be effective)**

`firewall-cmd --list-all` **(To list all firewall configurations and check if newly created rule is now available)**

To test this :

Configure virtual host on server.example.com as explained in apache objectives with ServerName=port.example.com (for example).

port.example.com should be resolved by DNS to IP address

First access this Virtual host from server.example.com machine using text or firefox browser :

`yum install elinks` **(To install text browser)**

`elinks http://port.example.com` **(Web page with contents of html must be shown)**

Go to client machine to test working of port forwarding :

Commands: (On client.example.com)

`yum install elinks` **(To install text browser)**

`elinks http://port.example.com:444` **(Access the Virtual host using 444 port and Web page with contents of html must be shown)**

2. On server machine configure port forwarding for SSH traffic coming on **111** port to **client** machine.

Commands: (On server.example.com)

firewall-cmd --list-all (To list all firewall configurations)

To add rich rule to forward SSH traffic coming on port 111 on server.example.com from any host to client.example.com :

firewall-cmd --add-rich-rule 'rule family="ipv4" forward-port to-addr="192.168.122.20" to-port="22" protocol="tcp" port="111"' –permanent

firewall-cmd --add-masquerade --permanent (Don't forget to set masquerade=yes in case you want to forward the traffic to some other machine)

firewall-cmd --reload (To reload the firewall to make the changes done in above command to be effective)

firewall-cmd --list-all (To list all firewall configurations and check if newly created rule is now available and masquerade is set to yes)

To test this : Go to client.example.com

Commands: (On client.example.com)

ssh server.example.com (Establish SSH connection to server machine on default port and it will be successful)

hostname (Display the hostname and it will be server.example.com, this command will be actually executed on server machine because of ssh connection)

logout (Logout to terminate the SSH connection)

ssh -p 111 server.example.com (Establish SSH connection to server machine on non-default port 111 and it will be successful but this time Connection request is redirected from server machine back to client machine due to SSH port forwarding and you will be connected to client machine, not the server machine)

hostname (Display the hostname and it will be client.example.com)

Note – In task 1, port forwarding was configured to forward traffic on same machine and so masquerade was not set to yes but in this case it must be set to yes.

3. Configure firewall rich rules on server to allow **http** traffic from **192.168.122.0/24** network only.

<u>Commands: (On server.example.com)</u>

```
firewall-cmd --list-all
```
(To list all firewall configurations)

To add rich rule to accept http traffic only from 192.168.122.0/24 Network :

```
firewall-cmd --add-rich-rule ‘rule family=“ipv4” source address=“192.168.122.0/24” service name=“http” accept’ –permanent
```

Remove http service from services list ,if you don’t remove http service ,then rich rule configured to accept http traffic from 192.168.122.0/24 network only will not be effective. This is due to order in which firewalld evaluates the different definitions on firewall.

If firewalld will find http service on services list, it will allow access irrespective of accessing network and rich rule will be ignored.

```
firewall-cmd --remove-service=http --permanent
```

```
firewall-cmd --reload
```
(To reload the firewall to make the changes done in above command to be effective)

```
firewall-cmd --list-all
```
(To list all firewall configurations and check if newly created rule is now available)

<u>To test this</u> :

We have only one network so it is not possible to test this. To test this, you just add this rule to allow access from client machine(replace 192.168.122.0/24 with 192.168.122.20) and then check http access from client machine and it should work and it should be denied from client2 machine.

Same is applicable in case of networks.

4. Configure SSH access restrictions on both server and client machines in such a way that machines on domain **insecure.com(10.1.1.0/24)** should not be able to access the machines.

Commands: (On server.example.com and client.example.com)

We will configure access by using TCP Wrappers :

vim /etc/hosts.deny **(To Deny SSH access for 10.1.1.0/24 Network)**

```
SSHD:10.1.1.0/24
:wq
```

How it works :

First hosts.allow file is checked if access is allowed and no other file is checked and access is allowed.

Then hosts.deny file is checked (If no entry in hosts.allow file) ,if entry is there access will be denied.

If there is no entry in any of these two files, access will be allowed by default.

For this task, we used hosts.deny file only and so access will be denied as per above explained flow. You can use files in combination to configure access as per task requirements.

5. Configure server to forward incoming traffic on **port 8080/tcp** to **192.168.122.40:80** (**client2.example.com**).

Also configure server for firewalld SSH logging with a prefix of "**SSH_**" and a debug level, limit to 2 log entries per minute. The changes should persist after reboot.

Commands: (On server.example.com)

firewall-cmd --list-all **(To list all firewall configurations)**

To add rich rule on server.example.com to forward http traffic coming on port 8080 from any host to client2.example.com on port 80 :

firewall-cmd --add-rich-rule 'rule family="ipv4" forward-port to-addr="192.168.122.40" to-port="80" protocol="tcp" port="8080"' –permanent

firewall-cmd --add-masquerade --permanent **(Don't forget to set masquerade=yes in case you want to forward the traffic to some other machine)**

firewall-cmd --add-rich-rule 'rule service name="ssh" log prefix "SSH_" level "debug" limit value="2/m" accept' --permanent **(To configure SSH logging)**

firewall-cmd --remove-service=ssh --permanent **(We need to remove ssh service from services list as explained in task 3)**

firewall-cmd --reload **(To reload the firewall to make the changes done in above command to be effective)**

firewall-cmd --list-all **(To list all firewall configurations and check if newly created rules are now available)**

Commands: (On client2.example.com)

Configure firewall on client2 to accept incoming http traffic, This is required when server machine will forward traffic coming on port 8080 to client2 machine, firewall on client2 must allow this. Don't forget this!!

firewall-cmd --add-service=http --permanent **(To add http service on firewall)**

firewall-cmd --reload **(To reload the firewall to make the changes done in above command to be effective)**

To test http port forwarding :

On client.example.com machine, access the Virtual Host port.example.com configured on server.example.com with non-default port of 8080 and on receiving this request server.example.com will forward http traffic to client2 machine on port 80 and contents of webpage on client2 should be displayed .

Make sure web server port.example.com is also configured on client2 machine otherwise default web server contents will be shown.

6. Configure **/etc/hosts.deny** and **/etc/hosts.allow** file on **server.example.com** to block the Samba service for **10.1.1.0/24** network allowing only one **IP 10.1.1.3** from the same network.

Commands: (On server.example.com)

We will configure access by using TCP Wrappers :

```
vim /etc/hosts.allow
```
(To Allow Samba service access for 10.1.1.3 Host)

```
smbd:10.1.1.3
:wq
```

```
vim /etc/hosts.deny
```
(To deny Samba service access for 10.1.1.0/24 Network)

```
smbd:10.1.1.0/24
:wq
```

How it works :

First hosts.allow file is checked if access is allowed and no other file is checked and access is allowed.

Then hosts.deny file is checked (If no entry in hosts.allow file) ,if entry is there access will be denied.

If there is no entry in any of these two files, access will be allowed by default.

For this task,Access will be allowed to host 10.1.1.3 host as defined in hosts.allow file which is checked first and rest all host on same network will be denied access based on definition in hosts.deny file.

7. Create a firewall service with name **new_service** which uses same port information as NFS service.

Commands: (On server.example.com)

cd /usr/lib/firewalld/services/   (Move to directory where firewalld services port information is present in XML files)

ls   (List the files in this directory)

more nfs.xml (To display the port information of nfs service)

cp nfs.xml /etc/firewalld/services/nfs_service (To define new firewalld service with name nfs_service and with same information as that of nfs service)

systemctl restart firewalld (To make changes effective)

firewall-cmd --get-services (To display the firewalld services)

You will find the new service in the list☺

Make sure New service should be added under /etc/firewalld/services and not under /usr/lib/firewalld/services/

8. Configure iscsi target on **server.example.com** machine.

   a> iscsi disk name is iqn.2018-10.com.example:server

   b> iscsi should use default port as 3260.

   c> target should use 1 GiB backing volume named as iscsi.

   d> target should available to only **client.example.com** machine.

Commands: (On server.example.com)

yum install targetcli (To install Packages required)

systemctl start target.service (To start Service)

systemctl enable target.service (To enable service)

systemctl status target.service (To check the status service)

Create LVM with name lv (in my case) of size 1 GiB which we will use for iscsi target. (I am not including commands for this here because I assume being RHCSA you know how to do this.(Refer to lecture)

targetcli (To access the tagetcli prompt, here we will configure iscsi target)

cd /backstores/block

create dev=/dev/vg/lv name=iscsi (To create block storage object,/dev/vg/lv is name in my case, you can use any name)

cd iscsi (To configure iscsi target)

create wwn=iqn.2018-10.com.example:server (Use specific format for wwn name)

cd iscsi/iqn.2018-10.com.example:server/tpg1/luns (To link the Block storage to Logical Unit number)

create /backstores/block/iscsi

cd iscsi/iqn.2018-10.com.example:server/tpg1/acls (To define the ACL to accept traffic only from client.example.com)

create wwn=iqn.2018-10.com.example:client

cd iscsi/iqn.2018-10.com.example:server/tpg1/portals (To create the portals to define the service to listen on specific IP:PORT combination)

create ip_address=192.168.122.10 ip_port=3260 (Target service is bind to listen on eth0 interface and default port 3260)

exit (exit targetcli prompt,All definitions will be saved automatically)

systemctl restart target (Restart the service)

firewall-cmd --add-port=3260/tcp --permanent (To add port on firewall to accept inbound traffic)

firewall-cmd --add-port=3260/udp --permanent (To add port on firewall to accept inbound traffic)

firewall-cmd --reload (To reload the firewall to make the changes done in above command to be effective)

Now we will define iscsi initiator on client.example.com in next task(Refer to lecture in case of doubts)

9. Configure **client.example.com** machine for iscsi intiator.

a> Iscsi device should be automatically mounted at booting time.

b> Iscsi should contain a block of **500MB** and should have **xfs** file system on it.

c> The partition must be mounted on **/init/iscsi** and it should be automatically mounted.

Commands: (On client.example.com)

yum install iscsi-initiator-utils (To install Packages required)

systemctl start iscsi (To start service, don't worry if it fails at this point)

systemctl enable iscsi (To enable service)

vim /etc/iscsi/initiatorname.iscsi (In this file we will define initiator name same as we defined in ACL on server side)

InitiatorName=iqn.2018-10.com.example:client

:wq

systemctl restart iscsi (To restart service)

iscsiadm --mode discoverydb --type sendtargets --portal 192.168.122.10 --discover (To discover iscsi targets)

iscsiadm --mode node --targetname iqn.2018-10.com.example:server --portal 192.168.122.10:3260 --login (To login iscsi targets)

yum install lsiscsi (Install this package to use lsiscsi utility to list targets)

lsscsi (To list iscsi target and you will find target in the list)

Now create a partition of 500 MB-MegaBytes using fdisk on this new iscsi disk and format this with xfs file system.

Create the mount directory /init/iscsi and mount the partition using UUID through fstab file. Make sure you are using UUID to mount this.

This is because disk name can change in case of iscsi disk.

10. Configure fileio based iscsi target of size 200 MiB on **server.example.com** machine.

a> iscsi disk name should be **iqn.2018-10.com.example:file**

b> Mount iscsi device persistently on client machine on **/iscsifile** with ext4 file system on it.

Commands: (On server.example.com)

yum install targetcli (To install Packages required)

systemctl start target.service (To start Service)

systemctl enable target.service (To enable service)

systemctl status target.service (To check the status service)

targetcli (To access the tagetcli prompt, here we will configure iscsi target)

cd /backstores/fileio

create file_or_dev=/root/iscsi_file size=200MB name=iscsi_file (To create fileio backend storage)

cd iscsi (To configure iscsi target)

create wwn=iqn.2018-10.com.example:file (Use specific format for wwn name)

cd iscsi/iqn.2018-10.com.example:file/tpg1/acls (To define the ACL to accept traffic only from client.example.com)

create wwn=iqn.2018-10.com.example:client

cd iscsi/iqn.2018-10.com.example:file/tpg1/luns (To link the fileio storage to Logical Unit number)

create /backstores/fileio/iscsi_file

cd iscsi/iqn.2018-10.com.example:file/tpg1/portals (To create the portals to define the service to listen on specific IP:PORT combination)

create ip_address=192.168.122.10 ip_port=3260 (Target service is bind to listen on eth0 interface and default port 3260)

exit (exit targetcli prompt,All definitions will be saved automatically)

systemctl restart target (Restart the service)

Commands: (On client.example.com)

yum install iscsi-initiator-utils (To install Packages required)

vim /etc/iscsi/initiatorname.iscsi (In this file we will define initiator name same as we defined in ACL on server side, Already done in previous task)

InitiatorName=iqn.2018-10.com.example:client

:wq

systemctl restart iscsi (To restart service)

iscsiadm --mode discoverydb --type sendtargets --portal 192.168.122.10 --discover (To discover iscsi targets,You will see wwn of new target on list)

iscsiadm --mode node --targetname iqn.2018-10.com.example:file --portal 192.168.122.10:3260 --login (To login iscsi target)

yum install lsiscsi (Install this package to use lsiscsi utility to list targets)

lsscsi (To list iscsi target and you will find target disk in the list)

Format the complete disk with ext4 filesystem.

Create the mount directory /iscsifile and mount the device using UUID through fstab file. Make sure you are using UUID to mount this.

This is because disk name can change in case of iscsi disk.

11. Configure CHAP authentication for iSCSI target.

On IPA Server (I used IPA server for this task) , Configure the iscsi target as we did in last two tasks, can be block based or fileio based.

under iscsi ,execute below commands :

targetcli (To access the tagetcli prompt, here we will configure iscsi target)

cd iscsi

set discovery_auth enable =1 (To enable the discovery authentication)

set discovery_auth userid=riya password=password (set userid and password)

exit

systemctl restart target (To restart service)

On Client side(client2, I used client2 for this task)

yum install iscsi-initiator-utils (To install Packages required)

systemctl start iscsi (To start service, don't worry if it fails at this point)

systemctl enable iscsi (To enable service)

vim /etc/iscsi/initiatorname.iscsi (In this file we will define initiator name same as you defined in ACL on server side)

InitiatorName=iqn.2018-10.com.example:client1

:wq

systemctl restart iscsi (To restart service)

```
vim /etc/iscsi//iscsid.conf
discovery.sendtargets.auth.authmethod = CHAP
```
(uncomment this line, to enable CHAP auth on client side)

```
discovery.sendtargets.auth.username = riya
discovery.sendtargets.auth.password =  password
:wq
systemctl restart iscsi
```
(To restart service)

```
iscsiadm --mode discoverydb --type sendtargets --portal 192.168.122.10 --discover
```
(To discover iscsi targets)

Discovery od iscsi target at client side would be possible only after enabling CHAP authentication at client side and provide the userid and password as above, otherwise this will not be possible.

12. Create a custom command on server with name **status** which should execute **ps –ef**

Commands: (On server.example.com)

```
vim /etc/bashrc
alias status="ps -ef"
```

(Add command alias in this file)

```
:wq
```

Execute status command and it should be successful and give same output as ps -ef.