Lab 4: Managing Firewall – 20 Minutes

Step 1: Install the httpd server on serverX

```
# yum -y install httpd

# systemctl enable httpd

# systemctl start httpd
```

Step 2: Create the default index.html file for displaying the landing page

```
# echo “welcome to server”  > /var/www/html/index.html
```

Step 3: Access the website from localhost and then from clientX

On server

```
# curl http://localhost
```

the website should be accessible

On client

```
# curl http://ipofserver
```

the website should not be accessible

Step 4: On serverX, stop nftables service and start firewalld service and create the rule to allow http service

```
#  systemctl status nftables

#  systemctl mask nftables

#  systemctl status firewalld

#  firewall-cmd –permanent –add-service=http

# firewall-cmd –reload
```

Verify the access from the client machine.

```
# curl http://ipofserver
```

Lab 5: Configure Selinux – 20 Minutes

You should be able to configure the Apache HTTP server to publish web content from a non standard document root.

Step 1: Login into serverX machine and install the httpd web server

```
# yum -y install httpd
```

```
# systemctl enable httpd
```

```
# systemctl start httpd
```

Step 2: Verifty that selinux should be enforcing mode

```
 # getenforce
```

```
# setenforce 1
```

Step 3: Install the setroubleshoot-server package on the server to send SELinux messages to /var/log/messages. setroubleshoot-server listens for audit messages in /var/log/audit/audit.log and sends a short summary to /var/log/messages.

```
#  yum install setroubleshoot-server -y
```

Step 4: Create the file in /root and mv it to document root of apache. Access the site to generate the selinux error

```
# touch "/root/file1"
```

```
# mv "/root/file1" "/var/www/html"
```

```
# systemctl restart httpd
```

```
# curl http://localhost/file1
```

Step 5: Verify the logs generated in audit.log and messages

```
# tail /var/log/audit/audit.log
```

```
# tail /var/log/messages
```

Both log files indicate that an SELinux denial is the culprit. The sealert command that is part of the output in /var/log/messages provides extra information, including a possible fix.

```
# sealert -l idofmessage
```

Step 6: To resolve the issue use the semanage and restorecon commands. The context to manage is httpd_sys_content_t

```
# semanage fcontext -a -t httpd_sys_content_t '/ var/www/html/file1’
```

```
# restorecon -Rv “/var/www/html”
```

```
# curl http://localhost/file1
```

Lab 6: Working with Samba – 30 Minutes

Choose one machine as serverX and another machine as clientX, set the hostname accordingly

```
# hostnamectl set-hostname serverx.example.com
```

```
# hostnamectl set-hostname clientX.example.com
```

Server Message Block (SMB) is the standard file-sharing protocol for Microsoft Windows servers and clients. SMB file servers can be configured in a number of different ways. One of the simplest is to configure the file servers and their clients as members of a common Windows workgroup, which announces servers and clients to the local subnet. The file servers each manage their own local user accounts and passwords independently

Share a directory with SMB on serverX according to the given requirements, then mount it on clientX.

Step 1: Deploy the required RPM packages to run the SMB service on serverX

```
# yum install samba
```

Step 2: Create the auxiliary system group marketing and the /smbshare directory on serverX. The marketing system group owns the /smbshare directory.

Adjust the permissions Providing File-based Storage on the /smbshare directory to have the SGID bit set, and write is prohibited by others.

The SELinux context type on the /smbshare directory and all newly created files and subdirectories is samba_share_t.

```
# groupadd -r marketing

# mkdir -p /smbshare

# chgrp marketing /smbshare

# chmod 2775 /smbshare

# semanage fcontext -a -t samba_share_t '/smbshare(/.*)?'

# restorecon -vvFR /smbshare
```

Step 3: Change the /etc/samba/smb.conf configuration file on serverX to reflect the configuration requested. Modify or confirm the following:

```
[global]

workgroup = mycompany
```

```
security = user
 passdb backend = tdbsam
```

Add a section at the end of the file as follows.

```
[smbshare]
path = /smbshare
write list = @marketing
```

save and quit the file

Step 4: Create the Samba-only user brian, who is part of the marketing team. The user brian has read and write access to the smbshare SMB share.

A new Samba user rob is created who is not part of the marketing team. The user rob has read access to the smbshare SMB share. Both newly added users have the SMB password redhat.

```
# yum -y install samba-client
# useradd -s /sbin/nologin -G marketing brian
# useradd -s /sbin/nologin rob
# smbpasswd -a brian   ## set the password to redhat
# smbpasswd -a rob   ## set the password to redhat
# systemctl start smb nmb
# systemctl enable smb nmb
```

Step 5: Verify the newly created SMB share works as expected on the clientX system with the created Samba-only users brian and rob.

```
# yum -y install cifs-utils
# mkdir “/mnt/brain”
# mount -o username=brian //serverX/smbshare /mnt/brian
# echo "Hello World" >/mnt/brian/brian1.txt
```

you should be able to write data in it

```
# mkdir /mnt/rob
# mount -o username=rob //serverX/smbshare /mnt/rob
# touch /mnt/rob/rob1.txt
```

You should get the permission denied error