RHCEv8 Online Class 12062021 10:00pm
RHCSA-sysadmin
- Managing SELinux Security
- Implementing Advanced Storage Features

Managing SELinux Security
CHANGING THE SELINUX ENFORCEMENT MODE
we have 2security models:
1-Discretionary Access Control-DAC
- -permission
- -special permission
- -adv. permission
- -sudo

2-Mandetory Access Control-MAC
- -SELinux

SELinux,
-its set of security rules that determine which process can access to which file, directory and ports.
-every file, dir, process and port they have special security label called SELinux context.
-A context is a name used by SELinux policy to determine whether a process can access to file, dir and port.
-by default SELinux policy won't allow any interaction unless explicitly rule granted access.
-SELinux protects user's data by attaching security context label to file, dir, port, process
-SELinux security label has several parts:
user
role
type
sensitivity
between them **type** is important.

SELinux modes:

1-enforcing	SELinux security policy is enforced.
2-permissive	SELinux prints warnings instead of enforcing.
3-disabled	No SELinux policy is loaded.

current SELinux mode
```
# getenforce
Enforcing
# sestatus
```

change SELinux mode:
temporary
```
# setenforce 1/0
usage:  setenforce [ Enforcing | Permissive | 1 | 0 ]
```
persistently
```
# vim /etc/selinux/config              ->original
SELINUX=enforcing
:wq!
or
# vim /etc/sysconfig/selinux           ->softlink
```

watch SELinux security context
```
# touch /tmp/file1
# mkdir /tmp/testdir
```
watch DAC
```
# ls -l /tmp/file1
-rw-r--r--. 1 root root 0 Jun 12 10:54 /tmp/file1
```
watch MAC
```
# ls -lZ /tmp/file1
-rw-r--r--. 1 root root unconfined_u:object_r:user_tmp_t:s0 0 Jun 12 10:54 /tmp/file1
```

SELinux scenario:
-SELinux on port
-SELinux on file/dir

-SELinux on port
servera.lab.example.com 172.25.250.10
serverb.lab.example.com 172.25.250.11
order:
change ssh port on servera **22/tcp** to **22000/tcp** and make ssh from serverb to servera with new port
servera
netstat -lnptu
tcp 0 0 0.0.0.0:22
tcp6 0 0 :::22
systemctl status sshd.service
vim /etc/ssh/sshd_config
17 Port 22000
:wq!
firewall-cmd --permanent --add-port=22000/tcp
firewall-cmd --reload
firewall-cmd --list-all
systemctl restart sshd.service
Job for sshd.service failed because the control process exited with error code.
journalctl | grep "ssh"
SELinux is preventing sshd from name_bind access on the tcp_socket port 22000.

SELinux log
sealert -l 7786ad15-3f9b-42f2-86f1-a1ccd7ff71ef
getenforce
Enforcing

Solution
add port 22000 in to SELinux policy database for ssh_port_t
semanage port -l
semanage port -l | grep "ssh"
ssh_port_t tcp 22
semanage port -a -t ssh_port_t -p tcp 22000
-a add
-d delete
-t type of security context
-p protocol tcp/udp
systemctl restart sshd.service
netstat -lnptu
tcp 0 0 0.0.0.0:22000
tcp6 0 0 :::22000

verify from serverb
ssh root@servera.lab.example.com -p 22000

-SELinux on file/dir
yum search apache
httpd.x86_64 : Apache HTTP Server
yum info httpd.x86_64
yum list httpd.x86_64
yum install httpd.x86_64 -y
systemctl list-unit-files | grep "http"
httpd.service disabled
systemctl enable httpd.service
systemctl start httpd.service
systemctl status httpd.service
firewall-cmd --permanent --add-port=80/tcp
firewall-cmd --reload
firewall-cmd --list-all
-DocumentRoot ->/var/www/html/
-DirectoryIndex ->index.html
DocumentRoot/DirectoryIndex ->/var/www/html/index.html
by default, when Install Apache web-server DocumentRoot will create but DirectoryIndex should be
echo "Hello Class!" >/var/www/html/index.html
cat /var/www/html/index.html
Hello Class!
systemctl restart httpd.service
curl localhost
Hello Class!

check MAC
ls -ldZ /var/
drwxr-xr-x. 22 root root system_u:object_r:var_t:s0 4096 Jun 12 11:18 /var/
ls -ldZ /var/www/
drwxr-xr-x. 4 root root system_u:object_r:httpd_sys_content_t:s0 33 Jun 12 11:18 /var/www/
ls -ldZ /var/www/html/
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_content_t:s0 24 Jun 12 11:27 /var/www/html/
ls -ldZ /var/www/html/index.html
-rw-r--r--. 1 root root unconfined_u:object_r:httpd_sys_content_t:s0 13 Jun 12 11:27 /var/www/html/index.html

all got inherit from parent
-index.html from html dir
-html dir from www dir

semanage fcontext -l
semanage fcontext -l | grep "/var/www(/.*)?"
/var/www(/.*)? all files system_u:object_r:httpd_sys_content_t:s0
(/.*)? ->directory + whatever will create inside it.

order:
change DocumentRoot to /virtual
mkdir /virtual
echo "Hellooooooo!" >/virtual/index.html
ls -ldZ /virtual/
drwxr-xr-x. 2 root root unconfined_u:object_r:default_t:s0 24 Jun 12 11:36 /virtual/
ls -lZ /virtual/index.html
-rw-r--r--. 1 root root unconfined_u:object_r:default_t:s0 12 Jun 12 11:36 /virtual/index.html
vim /etc/httpd/conf/httpd.conf
122 DocumentRoot "/virtual"
127 <Directory "/virtual">
135 <Directory "/virtual">
:wq!
systemctl restart httpd.service
curl localhost
it shows you:
HTTP SERVER TEST PAGE
but me expect is:
Hellooooooo!
because SELinux stopped access to /virtual/index.html

journalctl | grep "http"
SELinux is preventing httpd from getattr access on the file /virtual/index.html.

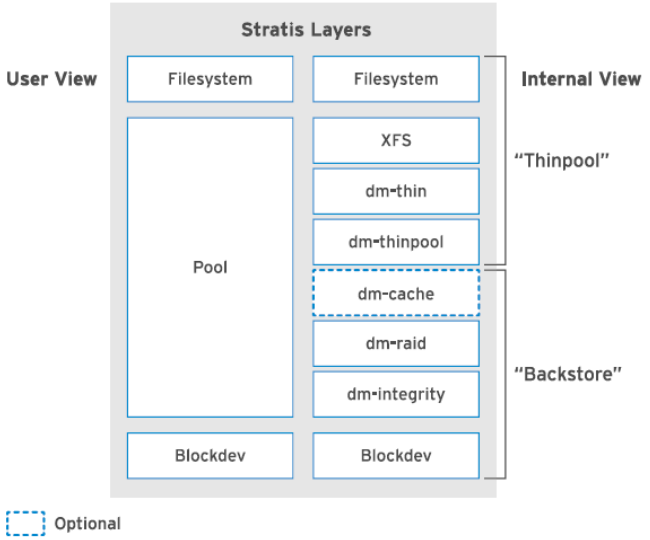
SELinux log
sealert -l 10f00d67-26b2-4a26-b025-d5f91d82831f
SELinux is preventing httpd from getattr access on the file /virtual/index.html.
If you want to allow httpd to have getattr access on the index.html file
Then you need to change the label on /virtual/index.html
Do
semanage fcontext -a -t FILE_TYPE '/virtual/index.html'

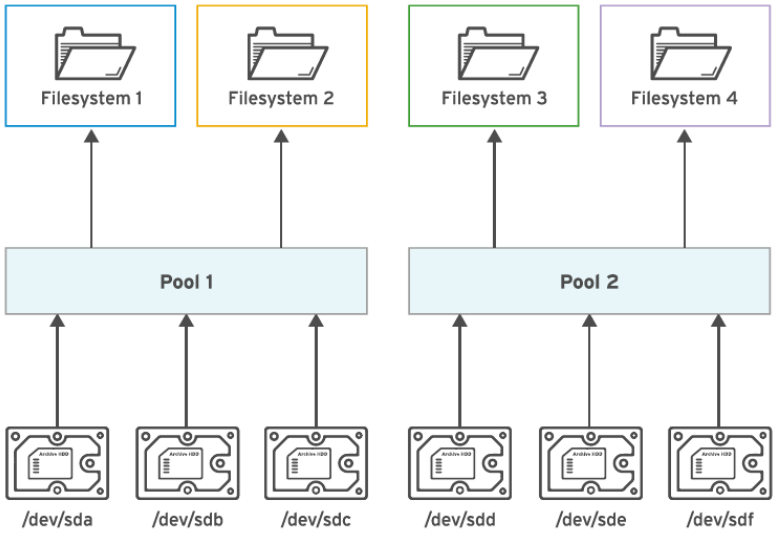
Solution
change /virtual directory and whatever inside it available context to apache acceptable context for SELinux.
1-temporary:
chcon -Rv -t httpd_sys_content_t /virtual/
changing security context of '/virtual/index.html'
changing security context of '/virtual/'
ls -ldZ /virtual/
drwxr-xr-x. 2 root root unconfined_u:object_r:httpd_sys_content_t:s0 24 Jun 12 11:36 /virtual/
ls -lZ /virtual/index.html
-rw-r--r--. 1 root root unconfined_u:object_r:httpd_sys_content_t:s0 12 Jun 12 11:36 /virtual/index.html
curl localhost
Hellooooo!

restore to default SELinux security context
restorecon -Rv /virtual/
Relabeled /virtual from unconfined_u:object_r:httpd_sys_content_t:s0 to unconfined_u:object_r:default_t:s0
Relabeled /virtual/index.html from unconfined_u:object_r:httpd_sys_content_t:s0 to unconfined_u:object_r:default_t:s0
curl localhost
it shows you:
HTTP SERVER TEST PAGE

2-persitently
change /virtual security context in to SELinux policy database from current to what should be:
semanage fcontext -a -t httpd_sys_content_t "/virtual(/.*)?"
restorecon -Rv /virtual/
Relabeled /virtual from unconfined_u:object_r:default_t:s0 to unconfined_u:object_r:httpd_sys_content_t:s0
Relabeled /virtual/index.html from unconfined_u:object_r:default_t:s0 to unconfined_u:object_r:httpd_sys_content_t:s0
curl localhost
Hellooooo!

Implementing Advanced Storage Features
MANAGING LAYERED STORAGE WITH STRATIS
DESCRIBING THE STRATIS ARCHITECTURE
-The current local storage solution in Red Hat Enterprise Linux (RHEL) includes many stable and mature technologies, including the device mapper (dm), the logical volume manager (LVM) and the XFS file system.
-With RHEL8, Red Hat introduces the Stratis local storage management solution.
Instead of developing from scratch, as other storage projects attempted, Stratis works with existing RHEL storage components.





Implement Stratis on Linux
stratis info
package: stratis-cli.noarch, stratisd.x86_64
daemon: stratid.service
working dir: /dev/stratis

steps:
-create pool
-create filesystem
-mount and use it
-create snapshot
Install
yum install stratis* -y
systemctl enable stratisd.service
systemctl start stratisd.service
systemctl status stratisd.service
lsblk
sdb 8:16 0 5G 0 disk
sdc 8:32 0 5G 0 disk
sdd 8:48 0 5G 0 disk
stratis
blockdev daemon filesystem -h --help key pool --propagate report --version
-create pool
stratis pool create <pool name> <storage name>
stratis pool create pool1 /dev/sdb
stratis pool list
stratis blockdev list
append storage to existing pool
stratis pool add-data <existing pool> <new storage>
stratis pool add-data pool1 /dev/sdc
stratis pool list
stratis blockdev list
-create filesystem
stratis filesystem create <existing pool> <file-system name>
stratis filesystem create pool1 fs1
stratis filesystem list
blkid
/dev/mapper/stratis-1-52a745e6ba094c3eafcd54c57b988238-thin-fs-43779909bd5e4af7a071f3d4ae4707aa: UUID="43779909-bd5e-4af7-a071-f3d4ae4707aa" TYPE="xfs"
tree /dev/stratis/
/dev/stratis/
└── pool1
    └── fs1 -> ../../dm-7

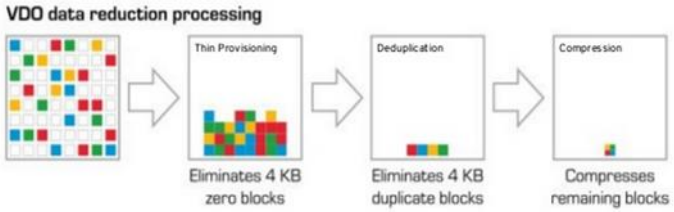
-mount and use it
mkdir /mnt/disk1
echo "/dev/mapper/stratis-1-52a745e6ba094c3eafcd54c57b988238-thin-fs-43779909bd5e4af7a071f3d4ae4707aa /mnt/disk1 xfs defaults,x-systemd.requires=stratisd.service 0 0" >>/etc/fstab
mount -a
df -hT
/dev/mapper/stratis-1-52a745e6ba094c3eafcd54c57b988238-thin-fs-43779909bd5e4af7a071f3d4ae4707aa xfs 1.0T 7.2G 1017G 1% /mnt/disk1
-create snapshot
stratis filesystem snapshot <existing pool> <existing filesystem> <snapshot name>
stratis filesystem snapshot pool1 fs1 fs1-snap
stratis filesystem list
restore snapshot
umount /mnt/disk1
vim /etc/fstab
delete fs1 record
:wq!
mount -a
stratis filesystem destroy pool1 fs1
stratis filesystem list
blkid
/dev/mapper/stratis-1-52a745e6ba094c3eafcd54c57b988238-thin-fs-81293b8746fb47e4b0350db88a727158: UUID="81293b87-46fb-47e4-b035-0db88a727158" TYPE="xfs"
echo "/dev/mapper/stratis-1-52a745e6ba094c3eafcd54c57b988238-thin-fs-81293b8746fb47e4b0350db88a727158 /mnt/disk1 xfs defaults,x-systemd.requires=stratisd.service 0 0" >>/etc/fstab
mount -a
df -hT
/dev/mapper/stratis-1-52a745e6ba094c3eafcd54c57b988238-thin-fs-81293b8746fb47e4b0350db88a727158 xfs 1.0T 7.2G 1017G 1% /mnt/dis
ls /mnt/disk1
file1 shadow

COMPRESSING AND DEDUPLICATING STORAGE WITH VDO
DESCRIBING VIRTUAL DATA OPTIMIZER
-RHEL8 includes the Virtual Data Optimizer-VDO driver, which optimizes the data footprint on block devices.
-VDO is a Linux device mapper driver that reduces disk space usage on block devices, and minimizes the replication of data, saving disk space and even increasing data throughput.
-VDO includes two kernel modules:
the kvdo module to transparently control data compression
the uds module for deduplication.

VDO How it works
VDO applies three phases to data in the following order to reduce the footprint on storage devices:
1. Zero-Block Elimination filters out data blocks that contain only zeroes (0) and records the information of those blocks only in the metadata. The nonzero data blocks are then passed to the next phase of processing. This phase enables the thin provisioning feature in the VDO devices.

2. Deduplication eliminates redundant data blocks. When you create multiple copies of the same data, VDO detects the duplicate data blocks and updates the metadata to use those duplicate blocks as references to the original data block without creating redundant data blocks.
The universal deduplication service-UDS kernel module checks redundancy of the data through the metadata it maintains.

3. Compression is the last phase. The kvdo kernel module compresses the data blocks using LZ4 compression and groups them on 4 KB blocks.



Implement VDO on Linux
vdo info
package: vdo.x86_64, kmod.x86_64
daemon: vdo.service

Install
yum install kmod.x86_64 vdo.x86_64 -y
systemctl enable vdo.service
systemctl start vdo.service
create
vdo create --name vdo1 --device /dev/sdd --vdoLogicalSize 50G
VDO instance 0 volume is ready at /dev/mapper/vdo1
mkfs.xfs -K /dev/mapper/vdo1
mkdir /mnt/disk2

```
# blkid
/dev/mapper/vdo1: UUID="d147065b-6c72-4470-8c21-faf569b1a68a" TYPE="xfs"
# echo "/dev/mapper/vdo1 /mnt/disk2 xfs defaults,x-systemd.requires=dvo.service 0 0" >>/etc/fstab
# mount -a
# df -hT
/dev/mapper/vdo1         xfs       50G  390M   50G   1% /mnt/disk2
# vdo status --name vdo1
# vdo status --name vdo1 | grep -i "deduplication"
# vdo status --name vdo1 | grep -i "compression"
```