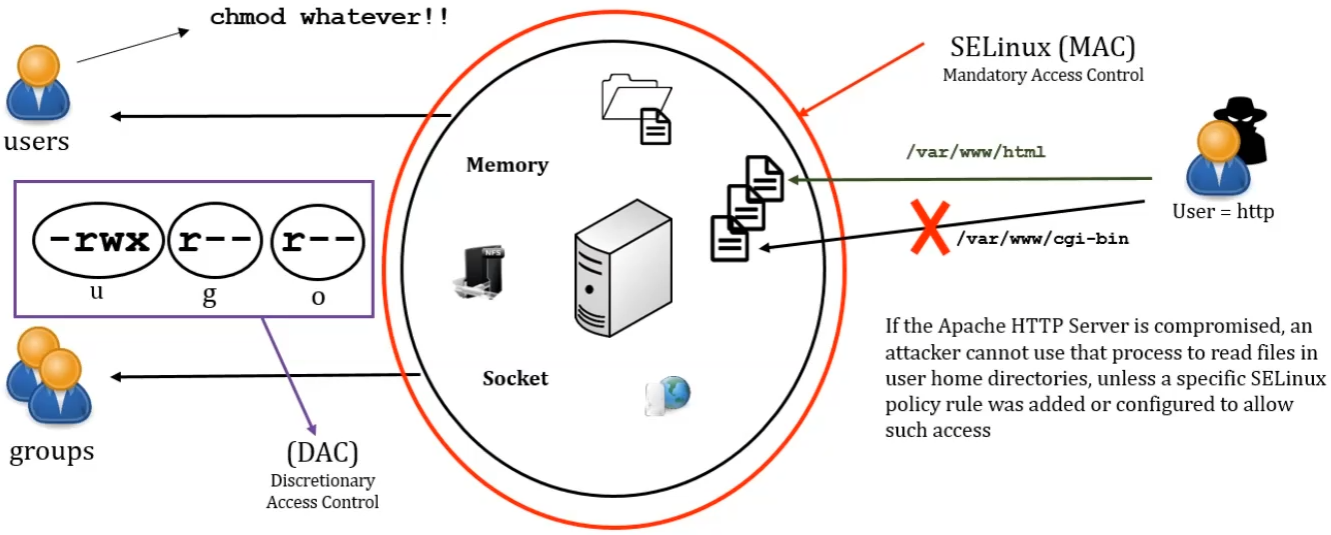
Managing SELinux Security

Linux #redhat #SELinux #labels #booleans

- What is SELinux?
  - Security-Enhanced Linux is a Linux kernel security module that provides a mechanism for supporting access control security policies, including mandatory access controls. (Wikipedia)
  - It is a project of the United States National Security Agency (NSA) and the SELinux community



- Graphical explanation of what SELinux is
- SELinux can give access to a user's files but it also can stop a user to access cascade files or other files within the same directory or group. In other words giving a user strict access to only certain files but not others.
- It is a good preventive tool for attacks and in case of compromised servers.
- **SELinux options**
  - **Enforcing** = Enabled (enabled by default in Redhat, CentOS and Fedora)
    - SELinux comes as enabled in these distributions
  - **Permissive** = Disabled but logs the activity
    - Tells SELinux to do not enable enforcing but to log the activity like if it was enable.
    - Recommended mode for beginners in order to try and error
  - **Disable** = Disabled and not activity logs
    - If you do not want SELinux at all you can disable it and none of its functionality would work at all.
    - The number one reason for being SELinux disabled in the environment its because it is complicated and a lot of people do find it complicated and simply they could go in and disable it.
    - It is recommended to enable SELinux if you are running in a DMZ environment, which is like public-facing environment.
- **To check SELinux status**
  - `sestatus` OR `getenfroce`
    - This will tell you if your system has SELinux enabled or not
- **SELinux setting**
  - `setenforce 0` = Permissive/Disable
  - `setenforce 1` = Enable
- **Modify SELinux config for permanent setting**

    - `/etc/selinux/config`
        - `SELINUX=enforcing`
        - `SELINUX=disabled`
        - `SELINUX=permissive`
            - Change that parameter to any mode to set the permanent SELinux setting
- **Before modifying SELinux config file**
    - Before you play around with SELinux, please if it's a virtual machine, create a snapshot, if it is a physical computer create a backup.
    - SELinux is built around security, if you make changes that is not something that you wanted to do, maybe you're gonna have a problem booting up your system or it's gonna take a long time to label every filesystem. Labeling will be cover later.
- **Before rebooting create a file**
    - `/ .autorelabel`
        - This is going to tell the system that we just enabled, disabled SELinux (specially enabling it) so it can go ahead and relabel all the filesystem directories, so they are aware of SELinux enablement.
        - If you do not do this then you have to boot, and it has to go through the relabeling again and it's gonna take you a really long time. And you're probably gonna think your system is not booting up and then you are gonna try different things and mess up things.

Check SELinux status

Example using `sestatus` command:

```
[root@localhost ~]# sestatus
```

Output:

```
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
Memory protection checking:     actual (secure)
Max kernel policy version:      33
```

- Check the "Current mode" line.

Example using `getenforce` command:

```
[root@localhost ~]# getenforce
```

Output:

```
Enforcing
```

Check SELinux config file

- `cd /etc/selinux/`
  - Traveling to the SELinux config directory

Example using `cat` command:

```
[root@localhost selinux]# cat config
```

Output:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
# See also:
# https://access.redhat.com/documentation/en-
us/red_hat_enterprise_linux/9/html/using_selinux/changing-selinux-states-and-modes_using-
selinux#changing-selinux-modes-at-boot-time_changing-selinux-states-and-modes
#
# NOTE: Up to RHEL 8 release included, SELINUX=disabled would also
# fully disable SELinux during boot. If you need a system with SELinux
# fully disabled instead of SELinux running with no policy loaded, you
# need to pass selinux=0 to the kernel command line. You can use grubby
# to persistently set the bootloader to boot with selinux=0:
#
#    grubby --update-kernel ALL --args selinux=0
#
# To revert back to SELinux enabled:
#
#    grubby --update-kernel ALL --remove-args selinux
#
SELINUX=enforcing
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

- You can use `vi /etc/selinux/config` to open this file and make modifications
  - You can add `SELINUX=enforcing`, `SELINUX=disabled`, or `SELINUX=permissive` after the few first lines and uncomment one of them to set the SELinux mode and save it.
  - For you to make this change globally or to system-wide you have to do a `reboot`

File editor:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
#SELINUX=disbaled
```

```
...
:wq!
```

- This will enable SELinux enforcing since boot.
- **When SELinux is disabled:**
  - If you check `getenforce` it will only output `SELinuyx status: disabled`
  - If you check any file's label by running `ls -lZ` or `ls -dZ`, it will still show the SELinux label the whole label will not be enforced at that moment, that means you could move the file anywhere and it will work based on the permissions which is defined in the ownership of the file (`rwx`).

Two main concepts

- **Labeling**
  - SELinux labels every file and directory in your filesystem. Again, it's an enhanced way of enhancing your security.
  - A label has 4 different type of information
    - User:
    - role:
    - type:
      - Most of the time a type is the one that is used that keeps all the files ownership or security separate from one another.
    - level
- **Type enforcement**

To list the label of a file

- `ls -lZ /usr/sbin/httpd`
- If you wanted to see what type of label it has you have to use the `-Z` option from the `ls` command.
- The file on this example is just used as an example of any file.

Example using `ls -lZ` command:

```
[root@localhost ~]# ls -lZ /usr/sbin/httpd
```

Output:

```
-rwxr-xr-x. 1 root root system_u:object_r:httpd_exec_t:s0 0 Jul  1 13:42 /usr/sbin/httpd
```

- Look at the label: `system_u:object_r:httpd_exec_t:s0`
  - It has system, which is for user (denoted with a `u`)
  - It has object, which is for role (denoted with a `r`)
  - It has httpd_exec, which is for type (denoted with a `t`)
  - It has s0, which is the level
- You will find that every file or directory if they are in different directories, would have a different type.
- As you know, this is an executable file, that is why its given the `_exec_t` type.

To list the label of a directory

- `ls -dZ /etc/httpd`
- The `-d` option stands for "directory"
  ls
  Example using `ls -dZ` command:

```
[root@localhost ~]# ls -dZ /etc/httpd
```

Output:

```
drwxr-xr-x. 1 root root system_u:object_r:httpd_config_t:s0 0 Jul  1 13:42 /etc/httpd
```

- This is a different type than the previous example. See it is labeled as a config file. (`_config_t`)
- Type is the most important part that you will be working with.
- If you are going to copy the file, let's say, that is in `usr/sbin` and you move it somewhere else, remember one thing, it is not taking the permissions off the other or the secure destination, and it is taking the information of the source. So you have to modify the type when you actually move it. Otherwise, SELinux won't allow you to read.

Example using `ls -dZ` command:

```
[root@localhost ~]# ls -dZ /home/mmarin
```

Output:

```
drwx------. 22 mmarin mmarin unconfined_u:object_r:user_home_dir_t:s0 4096 Jul  1 12:22 /home/mmarin
```

- You will see it is coming as `user_home_dir` type.
- This is how you can differentiate this is a home directory.
- We could also make a change on the user as well as on the object.

As the webserver runs its process is labeled in memory as `httpd_t`

- Not only directories or files are labeled by SELinux, the processes are also labeled. You could check that process label only if it's running right? Because kernel starts up the process, then it keeps that process information in its memory. Without starting it upo, there won't be any label.
- To check while the process is running:
  - `ps axZ | grep httpd`
    - Note that `ps ax` is used to see every process on the system using BSD syntax, and we are adding the `Z` option to get security info.
    - You can also use `ps -efZ` for a different format.

Example using `ps axZ` command:

```
[root@localhost ~]# ps axZ | grep httpd
```

- Looking if we are currently running the httpd process to check the SELinux type of those processes.

Output:

```
system_u:system_r:httpd_t:s0       4060 ?        Ss     0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0       4061 ?        S      0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0       4062 ?        Sl     0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0       4063 ?        Sl     0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0       4064 ?        Sl     0:00 /usr/sbin/httpd -DFOREGROUND
```

- Note the type of these processes is `httpd_t`

The SELinux assigns the label at the socket level

- To check the socket level
  - `netstat -tnlpZ | grep httpd`

Example using `netstat` command:

```
[root@localhost ~]# netstat -tnlpZ | grep http
```

- The `-t` option stands for "tcp" and specifies to list sockets.
- The `-n` option stands for "numeric" and specifies to don't resolve names.
- The `-l` option stands for "listening" and specifies to display listening server sockets.
- The `-p` option stands for "programs" and specifies to display PID/Program name for sockets.
- The `-Z` option stands for "context" and specifies to display SELinux security context for sockets.

Output:

```
tcp6       0      0 :::80                   :::*                    LISTEN      4060/httpd
system_u:system_r:httpd_t:s0
```

- Look at the label of the socket. Again it has the `httpd_t` type.

Command to manage SELinux settings

- `semanage` -> to label
  - `login`
  - `user`
  - `port`
  - `interface`
  - `module`
  - `node`
  - `file context`
  - boolean
    - This is mostly used while you're using SELinux
  - `permissive state`
  - `dontaudit`
- There is another command to change SELinux label:

- `chcon` - change file SELinux security context

Boolean

- ON / OFF Switch
- There are pre-defined out of box Booleans that come with SELinux
  - e.g. do we allow ftp server to access home directories?
    - If ON = yes, if OFF = no
  - e.g. can httpd talk to ldap...
    - ON / OFF
- **To get a list of all booleans**
  - `getsebool -a`
    - The `-a` option stands for "all" and specifies to list all booleans
  - OR `semanage boolean -l`
- **To enable or turn on a boolean**
  - Keep in mind, this is where we are making changes, only when SELinux is enabled you could do all those different configuration within SELinux
  - `setsebool -P [boolean_name] on/off`
- **Check error messages related to SELinux**
  - If you defined anything in your SELinux, if you set any boolean and there is an error message in it or if you are running an Apache server, for example. And apache server was supposed to read a file from a user home directory. You start the Apache server and you get an error message that it cannot read that file. (Even if there is an `r` permission at others level), so you probably have a level that prevents reading this file to some users, or allows only a single user to read it.
  - `journalctl`
    - This command is kind of a log, it keeps track of everything, every error message that is related to SELinux.
- **To change the type in a label**
  - `chcon -t httpd_sys_content_t FILENAME`
    - If this is a system file you would give that label, if its a home directory you give a `_home_dir_` label
    - The `-t` option stands for "type" and specifies to set the type in the target security context.
  - `semanage -t httpd_sys_content_t FILENAME`

Get list of booleans
Example using `getsebool -a` command:

```
[root@localhost ~]# getsebool -a
```

Output:

```
abrt_anon_write --> off
abrt_handle_event --> off
abrt_upload_watch_anon_write --> on
antivirus_can_scan_system --> off
...
httpd_can_connect_ftp --> off
...
```

- For example if the `httpd_can_connect_ftp` is off then your web server won't be able to access an ftp server to transfer any file even if the service is running.

Find out how many booleans we have
Example using `getsebool` command:

```
[root@localhost ~]# getsebool -a | wc -l
```

- Counting the lines of the `getsebool` output

Output:

```
356
```

Check for a specific boolean
Example using `getsebool` command:

```
[root@localhost ~]# getsebool -a | grep httpd_can_connect_ftp
```

Output:

```
httpd_can_connect_ftp --> off
```

Enable a boolean
Example using `setsebool` command:

```
[root@localhost ~]# setsebool -P httpd_can_connect_ftp on
```

- It could take anywhere from 2 to 10 seconds depending on your system

More on: # RHCSA RHEL 8 - Restore default file contexts

Diagnose and Address Routine SELinux Policy Violations

As a Linux administrator, it is essential to diagnose and address routine SELinux policy violations. The RHCSA exam expects candidates to know how to diagnose and troubleshoot SELinux policy violations.

One way to diagnose SELinux policy violations is to check the SELinux audit log using the ausearch command. For example, to list all SELinux policy violations for the last 24 hours, you can run the following command:

```
$ sudo ausearch -m avc -ts yesterday
```

Copy

This command lists all SELinux policy violations for the last 24 hours. To address an SELinux policy violation, you can use the audit2allow command to generate a custom policy module that allows the violated operation.

For example, if the SELinux policy violation was caused by a denied read operation, you can run the following command to generate a custom policy module that allows the read operation:

```
$ sudo grep denied /var/log/audit/audit.log | audit2allow -M mypol
```

Copy

This command generates a custom policy module named mypol that allows the read operation. You can then load the module using the semodule command:

```
$ sudo semodule -i mypol.pp
```

Copy

This command installs the mypol policy module.

More on: # RHCSA9 EXAM SERIES: Manage Security