Chapter 4 Guided Exercises and Labs

Add a coversheet with your name, module title, and pledge to the beginning of this document
Insert screenshots showing intermediate steps and completion of each of the guided exercises and labs at appropriate locations below and submit a pdf file

Guided Exercise: Change the SELinux Enforcement Mode

In this lab, you manage SELinux modes, both temporarily and persistently.

Outcomes

- View and set the current SELinux mode.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command prepares your environment and ensures that all required resources are available.

```
[student@workstation ~]$ lab start selinux-opsmode
```

Procedure 4.1. Instructions

1. On the `workstation` machine, use the `ssh` command to log in to the `servera` machine as the `student` user and then switch to the `root` user.

   ```
   [student@workstation ~]$ ssh student@servera
   ...output omitted...
   [student@servera ~]$ sudo -i
   [sudo] password for student: student
   [root@servera ~]#
   ```

2. Change the default SELinux mode to permissive.
   1. Use the `getenforce` command to verify the current SELinux mode on the `servera` machine.

      ```
      [root@servera ~]# getenforce
      Enforcing
      ```

   2. Use the `vim /etc/selinux/config` command to edit the configuration file. Change the `SELINUX` parameter from `enforcing` to `permissive` mode.

      ```
      [root@servera ~]# vim /etc/selinux/config
      ```

   3. Use the `grep` command to confirm that the `SELINUX` parameter displays the `permissive` mode.

      ```
      [root@servera ~]# grep '^SELINUX' /etc/selinux/config
      SELINUX=permissive
      SELINUXTYPE=targeted
      ```

   4. Use the `getenforce` command to confirm that the `SELINUX` parameter displays the `enforcing` mode.

      ```
      [root@servera ~]# getenforce
      Enforcing
      ```

   5. Use the `setenforce` command to change the `SELINUX` mode to `permissive` mode and verify the change.

      ```
      [root@servera ~]# setenforce 0
      [root@servera ~]# getenforce
      Permissive
      ```

3. Change the default SELinux mode back to the `enforcing` mode in the configuration file.
   1. Use the `vim /etc/selinux/config` command to edit the configuration file. Change the `SELINUX` parameter from `permissive` to `enforcing` mode.

      ```
      [root@servera ~]# vim /etc/selinux/config
      ```

   2. Use the `grep` command to confirm that the `SELINUX` parameter sets the `enforcing` mode on booting.

      ```
      [root@servera ~]# grep '^SELINUX' /etc/selinux/config
      SELINUX=enforcing
      SELINUXTYPE=targeted
      ```

4. Set the SELinux mode to `enforcing` in the command line. Reboot the `servera` machine and verify the SELinux mode.
   1. Use the `setenforce` command to set the current SELinux mode to the `enforcing` mode. Use the `getenforce` command to confirm that SELinux is set to the `enforcing` mode.

      ```
      [root@servera ~]# setenforce 1
      [root@servera ~]# getenforce
      Enforcing
      ```

2. Reboot the `servera` machine to implement the persistent configuration.

```
[root@servera ~]# systemctl reboot
Connection to servera closed by remote host.
Connection to servera closed.
[student@workstation ~]$
```

3. Log in to `servera` machine and verify the SELinux mode.

```
[student@workstation ~]$ ssh student@servera
...output omitted...
[student@servera ~]$ sudo -i
[sudo] password for student: student
[root@servera ~]# getenforce
Enforcing
```

5. Return to the `workstation` machine as the `student` user.

```
[root@servera ~]# exit
logout
[student@servera ~]$ exit
logout
Connection to servera closed.
[student@workstation ~]$
```

Finish

On the `workstation` machine, change to the `student` user home directory and use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish selinux-opsmode
```

This concludes the section.

Guided Exercise: Control SELinux File Contexts

In this lab, you persistently change the SELinux context of a directory and its contents.

Outcomes

- Configure the `Apache` HTTP server to publish web content from a non-standard document root.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command prepares your environment and ensures that all required resources are available.

```
[student@workstation ~]$ lab start selinux-filecontexts
```

Procedure 4.2. Instructions

1. Log in to `servera` as the `student` user and switch to the `root` user.

   ```
   [student@workstation ~]$ ssh student@servera
   ...output omitted...
   [student@servera ~]$
   [student@servera ~]$ sudo -i
   [sudo] password for student: student
   [root@servera ~]#
   ```

2. Configure Apache to use a document directory in a non-standard location.
   1. Create the `/custom` directory.

      ```
      [root@servera ~]# mkdir /custom
      ```

   2. Create the `index.html` file in the `/custom` directory. The `index.html` file should contain the `This is SERVERA.` text.

      ```
      [root@servera ~]# echo 'This is SERVERA.' > /custom/index.html
      ```

   3. Configure Apache to use the new directory location. Edit the Apache `/etc/httpd/conf/httpd.conf` configuration file and replace the two occurrences of the `/var/www/html` directory with the `/custom` directory. You can use the `vim /etc/httpd/conf/httpd.conf` command to do so. The following example shows the expected content of the `/etc/httpd/conf/httpd.conf` file.

```
[root@servera ~]# cat /etc/httpd/conf/httpd.conf
...output omitted...
DocumentRoot "/custom"
...output omitted...
<Directory "/custom">
...output omitted...
```

3. Start and enable the Apache web service and confirm that the service is running.
    1. Start and enable the Apache web service by using the `systemctl` command.

       ```
       [root@servera ~]# systemctl enable --now httpd
       Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service →
       /usr/lib/systemd/system/httpd.service.
       ```

    2. Verify that the service is running.

       ```
       [root@servera ~]# systemctl status httpd
       ● httpd.service - The Apache HTTP Server
            Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
            Active: active (running) since Wed 2022-04-06 05:21:19 EDT; 22s ago
              Docs: man:httpd.service(8)
          Main PID: 1676 (httpd)
       ...output omitted...
       Apr 06 05:21:19 servera.lab.example.com systemd[1]: Starting The Apache HTTP Server...
       Apr 06 05:21:19 servera.lab.example.com systemd[1]: Started The Apache HTTP Server.
       Apr 06 05:21:19 servera.lab.example.com httpd[1676]: Server configured, listening on: port 80
       ```

4. Open a web browser on `workstation` and try to view the `http://servera/index.html` web page. You get an error message that you do not have permission to access the file.
5. To permit access to the `index.html` file on `servera`, you must configure the SELinux context. Define an SELinux file context rule that sets the context type to `httpd_sys_content_t` for the `/custom` directory and all the files under it.

   ```
   [root@servera ~]# semanage fcontext -a \
   -t httpd_sys_content_t '/custom(/.*)?'
   ```

6. Correct the file contexts in the `/custom` directory.

   ```
   [root@servera ~]# restorecon -Rv /custom
   Relabeled /custom from unconfined_u:object_r:default_t:s0 to unconfined_u:object_r:httpd_sys_content_t:s0
   Relabeled /custom/index.html from unconfined_u:object_r:default_t:s0 to
   unconfined_u:object_r:httpd_sys_content_t:s0
   ```

7. Try to view `http://servera/index.html` again in the web browser on the `workstation` machine. You should see the `This is SERVERA.` message.
8. Return to the `workstation` machine as the `student` user.

   ```
   [root@servera ~]# exit
   logout
   [student@servera ~]$ exit
   logout
   Connection to servera closed.
   [student@workstation ~]$
   ```

Finish

On the `workstation` machine, change to the `student` user home directory and use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish selinux-filecontexts
```

This concludes the section.

Guided Exercise: Adjust SELinux Policy with Booleans

In this exercise, you configure Apache to publish web content from users' home directories.

Outcomes

- Configure Apache web service to publish web content from the user’s home directory.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command prepares your environment and ensures that all required resources are available.

```
[student@workstation ~]$ lab start selinux-booleans
```

Procedure 4.3. Instructions

1. On the `workstation` machine, use the `ssh` command to log in to the `servera` machine as the `student` user and then switch to the `root` user.

   ```
   [student@workstation ~]$ ssh student@servera
   ...output omitted...
   [student@servera ~]$ sudo -i
   [sudo] password for student: student
   [root@servera ~]#
   ```

2. Edit the `/etc/httpd/conf.d/userdir.conf` configuration file to enable the Apache feature so that users can publish web content from their home directory. Comment out the line in the `IfModule` section that sets the `UserDir` variable to the `disabled` value, and uncomment the line that sets the `UserDir` variable to the `public_html` value.

   ```
   [root@servera ~]# vim /etc/httpd/conf.d/userdir.conf
   <IfModule mod_userdir.c>
   ...output omitted...
       # UserDir disabled

   ...output omitted...
       UserDir public_html

   ...output omitted...
   </IfModule>
   ```

3. Start and enable the Apache web service.

```
[root@servera ~]# systemctl enable --now httpd
```

4. Open another terminal window, and use the `ssh` command to log in to the `servera` machine as the `student` user. Create the `index.html` web content file in the `~/public_html` directory.
    1. In another terminal window, use the `ssh` command to log in to the `servera` machine as the `student` user.

       ```
       [student@workstation ~]$ ssh student@servera
       ...output omitted...
       [student@servera ~]$
       ```

    2. Use the `mkdir` command to create the `~/public_html` directory.

       ```
       [student@servera ~]$ mkdir ~/public_html
       ```

    3. Create the `index.html` file with the following content:

       ```
       [student@servera ~]$ echo 'This is student content on SERVERA.' > \
       ~/public_html/index.html
       ```

    4. For the Apache web service to serve the contents of the `/home/student/public_html` directory, it must be allowed to share files and subdirectories in the `/home/student` directory. When you created the `/home/student/public_html` directory, it was automatically configured with permissions that allow anyone with home directory permission to access its contents.

       Change the `/home/student` directory permissions to allow the Apache web service to access the `public_html` subdirectory.

       ```
       [student@servera ~]$ chmod 711 ~
       [student@servera ~]$ ls -ld ~
       drwx--x--x. 16 student student 4096 Nov  3 09:28 /home/student
       ```

5. Open a web browser on the `workstation` machine and enter the `http://servera/~student/index.html` address. An error message states that you do not have permission to access the file.
6. Switch to the other terminal and use the `getsebool` command to see if any Booleans restrict access to home directories for the `httpd` service.

   ```
   [root@servera ~]# getsebool -a | grep home
   ...output omitted...
   httpd_enable_homedirs --> off
   ...output omitted...
   ```

7. Use the `setsebool` command to enable persistent access to the home directory for the `httpd` service.

   ```
   [root@servera ~]# setsebool -P httpd_enable_homedirs on
   ```

8. Verify that you can now see the `This is student content on SERVERA.` message in the web browser after entering the `http://servera/~student/index.html` address.
9. Return to the `workstation` machine as the `student` user.

   ```
   [root@servera ~]# exit
   logout
   [student@servera ~]$ exit
   logout
   Connection to servera closed.
   [student@workstation ~]$
   ```

Finish

On the `workstation` machine, change to the `student` user home directory and use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish selinux-booleans
```

This concludes the section.

Guided Exercise: Investigate and Resolve SELinux Issues

In this lab, you learn how to troubleshoot SELinux security denials.

Outcomes

- Gain experience with SELinux troubleshooting tools.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command prepares your environment and ensures that all required resources are available.

```
[student@workstation ~]$ lab start selinux-issues
```

Procedure 4.4. Instructions

1. From a web browser on the `workstation` machine, open the `http://servera/index.html` web page. An error message states that you do not have permission to access the file.
2. Use the `ssh` command to log in to `servera` as the `student` user. Use the `sudo -i` command to switch to the `root` user.

   ```
   [student@workstation ~]$ ssh student@servera
   ...output omitted...
   [student@servera ~]$ sudo -i
   [sudo] password for student: student
   [root@servera ~]#
   ```

3. Use the `less` command to view the contents of the `/var/log/messages` file. You use the **/** character and search for the `sealert` text. Press the **n** key until you reach the last occurrence, because previous exercises might also have generated SELinux messages. Copy the suggested `sealert` command so that you can use it in the next step. Use the **q** key to quit the `less` command.

   ```
   [root@servera ~]# less /var/log/messages
   ...output omitted...
   Apr  7 04:52:18 servera setroubleshoot[20715]: SELinux is preventing /usr/sbin/httpd from getattr access on the
   file /custom/index.html. For complete SELinux messages run: sealert -l 9a96294a-239b-4568-8f1e-9f35b5fb472b
   ...output omitted...
   ```

4. Run the suggested `sealert` command. Note the source context, the target objects, the policy, and the enforcing mode. Find the correct SELinux context label for the file that the `httpd` service tries to serve.
   1. Run the `sealert` command.

The output explains that the `/custom/index.html` file has an incorrect context label.

```
[root@servera ~]# sealert -l 9a96294a-239b-4568-8f1e-9f35b5fb472b
SELinux is preventing /usr/sbin/httpd from getattr access on the file /custom/index.html.

*****  Plugin catchall_labels (83.8 confidence) suggests   *******************

If you want to allow httpd to have getattr access on the index.html file
Then you need to change the label on /custom/index.html
Do
# semanage fcontext -a -t FILE_TYPE '/custom/index.html'
where FILE_TYPE is one of the following: NetworkManager_exec_t, NetworkManager_log_t, NetworkManager_tmp_t,
abrt_dump_oops_exec_t, abrt_etc_t, abrt_exec_t, abrt_handle_event_exec_t, abrt_helper_exec_t,
abrt_retrace_coredump_exec_t, abrt_retrace_spool_t, abrt_retrace_worker_exec_t, abrt_tmp_t,
abrt_upload_watch_tmp_t, abrt_var_cache_t, abrt_var_log_t, abrt_var_run_t, accountsd_exec_t, acct_data_t,
acct_exec_t, admin_crontab_tmp_t, admin_passwd_exec_t, afs_logfile_t, aide_exec_t, aide_log_t, alsa_exec_t,
alsa_tmp_t, amanda_exec_t, amanda_log_t, amanda_recover_exec_t, amanda_tmp_t, amtu_exec_t, anacron_exec_t,
anon_inodefs_t
...output omitted...

Additional Information:
Source Context                system_u:system_r:httpd_t:s0
Target Context                unconfined_u:object_r:default_t:s0
Target Objects                /custom/index.html [ file ]
Source                        httpd
Source Path                   /usr/sbin/httpd
Port                          <Unknown>
Host                          servera.lab.example.com
Source RPM Packages           httpd-2.4.51-7.el9_0.x86_64
Target RPM Packages
SELinux Policy RPM            selinux-policy-targeted-34.1.27-1.el9.noarch
Local Policy RPM              selinux-policy-targeted-34.1.27-1.el9.noarch
Selinux Enabled               True
Policy Type                   targeted
Enforcing Mode                Enforcing
Host Name                     servera.lab.example.com
Platform                      Linux servera.lab.example.com
                              5.14.0-70.2.1.el9_0.x86_64 #1 SMP PREEMPT Wed Mar
                              16 18:15:38 EDT 2022 x86_64 x86_64
Alert Count                   4
First Seen                    2022-04-07 04:51:38 EDT
Last Seen                     2022-04-07 04:52:13 EDT
Local ID                      9a96294a-239b-4568-8f1e-9f35b5fb472b

Raw Audit Messages
```

```
    type=AVC msg=audit(1649321533.406:1024): avc:  denied  { getattr } for  pid=20464 comm="httpd"
    path="/custom/index.html" dev="vda4" ino=25571802 scontext=system_u:system_r:httpd_t:s0
    tcontext=unconfined_u:object_r:default_t:s0 tclass=file permissive=0

    ...output omitted...
    ```

    2. Check the SELinux context for the directory from where the `httpd` service serves the content by default, `/var/www/html`. The `httpd_sys_content_t` SELinux context is appropriate for the `/custom/index.html` file.

       ```
       [root@servera ~]# ls -ldZ /var/www/html
       drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_content_t:s0 6 Mar 21 11:47 /var/www/html
       ```

5. The `Raw Audit Messages` section of the `sealert` command contains information from the `/var/log/audit/audit.log` file. Use the `ausearch` command to search the `/var/log/audit/audit.log` file. The `-m` option searches on the message type. The `-ts` option searches based on time. The following entry identifies the relevant process and file that cause the alert. The process is the `httpd` Apache web server, the file is `/custom/index.html`, and the context is `system_r:httpd_t`.

   ```
   [root@servera ~]# ausearch -m AVC -ts today
   ...output omitted...
   ----
   time->Thu Apr  7 04:52:13 2022
   type=PROCTITLE msg=audit(1649321533.406:1024): proctitle=2F7573722F7362696E2F6874747064002D44464F524547524F554E44
   type=SYSCALL msg=audit(1649321533.406:1024): arch=c000003e syscall=262 success=no exit=-13 a0=ffffff9c
   a1=7fefc403d850 a2=7fefc89bc830 a3=100 items=0 ppid=20461 pid=20464 auid=4294967295 uid=48 gid=48 euid=48 suid=48
   fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=4294967295 comm="httpd" exe="/usr/sbin/httpd"
   subj=system_u:system_r:httpd_t:s0 key=(null)
   type=AVC msg=audit(1649321533.406:1024): avc:  denied  { getattr } for  pid=20464 comm="httpd"
   path="/custom/index.html" dev="vda4" ino=25571802 scontext=system_u:system_r:httpd_t:s0
   tcontext=unconfined_u:object_r:default_t:s0 tclass=file permissive=0
   ```

6. Resolve the issue by applying the `httpd_sys_content_t` context.

   ```
   [root@servera ~]# semanage fcontext -a \
   -t httpd_sys_content_t '/custom(/.*)?'
   [root@servera ~]# restorecon -Rv /custom
   Relabeled /custom from unconfined_u:object_r:default_t:s0 to unconfined_u:object_r:httpd_sys_content_t:s0
   Relabeled /custom/index.html from unconfined_u:object_r:default_t:s0 to
   unconfined_u:object_r:httpd_sys_content_t:s0
   ```

7. Again, attempt to view `http://servera/index.html`. The `This is SERVERA.` message is displayed.
8. Return to the `workstation` machine as the `student` user.

   ```
   [root@servera ~]# exit
   ```

```
    logout
    [student@servera ~]$ exit
    logout
    Connection to servera closed.
    [student@workstation ~]$
```

Finish

On the `workstation` machine, change to the `student` user home directory and use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish selinux-issues
```

This concludes the section.

Lab: Manage SELinux Security

In this lab, you identify issues in system log files and adjust the SELinux configuration.

Outcomes

- Identify issues in system log files.
- Adjust the SELinux configuration.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command prepares your environment and ensures that all required resources are available.

```
[student@workstation ~]$ lab start selinux-review
```

Procedure 4.5. Instructions

1. Log in to the `serverb` machine as the `student` user and switch to the `root` user.
2. From a web browser on the `workstation` machine, view the `http://serverb/lab.html` web page. You see the error message: `You do not have permission to access this resource.`
3. Research and identify the SELinux issue that prevents the Apache service from serving web content.
4. Display the SELinux context of the new HTTP document directory and the original HTTP document directory. Resolve the SELinux issue that prevents the Apache server from serving web content.
5. Verify that the Apache server can now serve web content.
6. Return to the `workstation` machine as the `student` user.

Evaluation

As the `student` user on the `workstation` machine, use the `lab` command to grade your work. Correct any reported failures and rerun the command until successful.

```
[student@workstation ~]$ lab grade selinux-review
```

Finish

On the `workstation` machine, change to the `student` user home directory and use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish selinux-review
```

This concludes the section.