LAB

MANAGING LOCAL USERS AND GROUPS

PERFORMANCE CHECKLIST

In this lab you will set a default local password policy, create a supplementary group for three users, allow that group to use **sudo** to run commands as `root`, and modify the password policy for one user.

OUTCOMES

You should be able to:

- Set a default password aging policy of the local user's password.
- Create a group and use the group as a supplementary group for new users.
- Create three new users with the new group as their supplementary group.
- Configure the group members of the supplementary group to run any command as any user using **sudo**.
- Set a user-specific password aging policy.

BEFORE YOU BEGIN

Log in to `workstation` as `student` using `student` as the password.

On `workstation`, run **`lab users-review start`** to start the exercise. This script creates the necessary files to ensure that the environment is set up correctly.

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

1. From `workstation`, open an SSH session to `serverb` as `student`.
2. On `serverb`, ensure that newly created users have passwords that must be changed every 30 days.
3. Create the new group called `consultants` with a GID of **`35000`**.
4. Configure administrative rights for all members of `consultants` to be able to execute any command as any user.
5. Create the `consultant1`, `consultant2`, and `consultant3` users with `consultants` as their supplementary group.
6. Set the `consultant1`, `consultant2`, and `consultant3` accounts to expire in 90 days from the current day.
7. Change the password policy for the `consultant2` account to require a new password every 15 days.

8. Additionally, force the `consultant1`, `consultant2`, and `consultant3` users to change their passwords on the first login.

Evaluation

On `workstation`, run the **`lab users-review grade`** command to confirm success of this exercise.

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

Finish

On `workstation`, run **`lab users-review finish`** to complete this lab. This script deletes the user accounts and files created throughout the lab to ensure that the environment is clean.

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

This concludes the lab.

LAB

CONTROLLING ACCESS TO FILES

PERFORMANCE CHECKLIST

In this lab, you will configure permissions on files and set up a directory that users in a particular group can use to conveniently share files on the local file system.

OUTCOMES

You should be able to:

- Create a directory where users can work collaboratively on files.
- Create files that are automatically assigned group ownership.
- Create files that are not accessible outside of the group.

BEFORE YOU BEGIN

Log in to `workstation` as `student` using `student` as the password.

On `workstation`, run the **`lab perms-review start`** command. The command runs a start script that determines if `serverb` is reachable on the network. The script also creates the `techdocs` group and three users named `tech1`, `tech2`, and `database1`.

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

1. Use the **ssh** command to log in to `serverb` as the `student` user. Switch to `root` on `serverb` using `redhat` as the password.
2. Create a directory called **`/home/techdocs`**.
3. Change the group ownership of the **`/home/techdocs`** directory to the `techdocs` group.
4. Verify that users in the `techdocs` group can create and edit files in the `/home/techdocs` directory.
5. Set permissions on the **`/home/techdocs`** directory. On the **`/home/techdocs`** directory, configure setgid (2), read/write/execute permissions (7) for the owner/user and group, and no permissions (0) for other users.
6. Verify that the permissions are set properly.
7. Confirm that users in the `techdocs` group can now create and edit files in the `/home/techdocs` directory. Users not in the `techdocs` group cannot edit or create files in the `/home/techdocs` directory. Users `tech1` and `tech2` are in the `techdocs` group. User `database1` is not in that group.
8. Modify the global login scripts. Normal users should have a umask setting that prevents others from viewing or modifying new files and directories.
9. Log off from `serverb`.

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

Evaluation

On `workstation`, run the **`lab perms-review grade`** script to confirm success on this exercise.

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

Finish

On `workstation`, run the **`lab perms-review finish`** script to complete the lab.

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

This concludes the lab.

LAB

MANAGING NETWORKING

PERFORMANCE CHECKLIST

In this lab, you will configure networking settings on a Red Hat Enterprise Linux server.

OUTCOMES

You should be able to configure two static IPv4 addresses for the primary network interface.

BEFORE YOU BEGIN

Log in as the `student` user on `workstation` using `student` as the password.

From `workstation` run the **`lab net-review start`** command. The command runs a start script that determine if the host, `serverb`, is reachable on the network.

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

1. Use the **`ssh`** command to log in to `serverb` as the `student` user. The systems are configured to use SSH keys for authentication, so a password is not required to log in to `serverb`.
2. Use the **`sudo -i`** command to switch to the `root` user. If prompted, use `student` as the password.
3. Create a new connection with a static network connection using the settings in the table.

PARAMETER	SETTING
Connection name	lab
Interface name	en*X* (might vary, use the interface that has `52:54:00:00:fa:0b` as its MAC address)
IP address	172.25.250.11/24
Gateway address	172.25.250.254
DNS address	172.25.250.254

4. Configure the new connection to be autostarted. Other connections should not start automatically.
5. Modify the new connection so that it also uses the address 10.0.1.1/24.
6. Configure the **`hosts`** file so that 10.0.1.1 can be referenced as **`private`**.
7. Reboot the system.
8. From `workstation` use the **`ping`** command to verify that `serverb` is initialized.

Evaluation

On workstation, run the **lab net-review grade** script to confirm success on this lab.

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

Finish

On `workstation`, run the **lab net-review finish** script to finish this lab.

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

This concludes the lab.

LAB

INSTALLING AND UPDATING SOFTWARE PACKAGES

PERFORMANCE CHECKLIST

In this lab, you will manage software repositories and module streams, and install and upgrade packages from those repositories and streams.

OUTCOMES

You should be able to:

- Manage software repositories and module streams.
- Install and upgrade packages from repositories and streams.
- Install an RPM package.

BEFORE YOU BEGIN

Log in to `workstation` as `student` using `student` as the password.

On `workstation`, run the **`lab software-review start`** command. This script ensures that `serverb` is available. It also downloads any packages required for the lab exercise.

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

1. On `serverb` configure a software repository to obtain updates. Name the repository as `errata` and configure the repository in the **`/etc/yum.repos.d/errata.repo`** file. It should access `http://content.example.com/rhel8.0/x86_64/rhcsa-practice/errata`. Do not check GPG signatures.
2. On `serverb`, install new package *xsane-gimp* and the *Apache HTTP Server* module from the `2.4` stream and the `common` profile.
3. For security reasons, `serverb` should not be able to send anything to print. Achieve this by removing the *cups* package. Exit from the `root` account.
4. The start script downloads the *rhcsa-script-1.0.0-1.noarch.rpm* package in the **`/home/student`** directory on `serverb`.

   Confirm that the package *rhcsa-script-1.0.0-1.noarch.rpm* is available on `serverb`. Install the package. You will need to gain superuser privileges to install the package. Verify that the package is installed. Exit from `serverb`.

Evaluation

On `workstation`, run the **`lab software-review grade`** script to confirm success on this lab.

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

Finish

On `workstation`, run the **`lab software-review finish`** script to complete this exercise. This script removes the repository and packages created during this exercise.

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

This concludes the lab.