<epam>

Linux Users and Groups

Linux Essentials

TRAINING

CENTER

<epam>

Users

Linux OS family defines three types of users:

- **Superuser (root)** – user with full access rights for executing any actions on a system.
- **Simple user** - non-privileged user account. It has unrestricted access to own files only.
- **System user** – service account used for specific programs work.

User Accounts Storage

- **/etc/passwd** - file contains the user account information. It has a record per user account in the following format:

```
[username]:[x]:[UID]:[GID]:[Comment]:[Home directory]:[Default shell]
```

  - Fields **[username]** and **[Comment]** are self explanatory.
  - The **x** in the second field indicates that the account is protected by a shadowed password (in **/etc/shadow**), which is needed to logon as **[username]**.
  - The **[UID]** and **[GID]** fields are integers that represent the User Identification and the primary Group Identification to which **[username]** belongs, respectively.
  - The **[Home directory]** indicates the absolute path to **[username]**'s home directory, and
  - The **[Default shell]** is the shell that will be made available to this user when he or she logins the system.

User Accounts Storage

- **/etc/shadow** - contains passwords for user's account in encrypted format. Each record has the following format:

```
[username]:[password]:[Last Password Changed]:[Min]:[Max]:[Warn]:[Inactive]:[Expire]
```

  - **[username]:** is the account name.
  - **[password]:** password in encrypted format.
  - **[Last Password Changed]**: days since Jan 1, 1970 that password was last changed.
  - **[Min]**: The minimum number of days required between password changes, i.e. the number of days left before the user is allowed to change his/her password
  - **[Max]**: The maximum number of days the password is valid (after that user is forced to change his/her password)
  - **[Warn]**: The number of days before password is to expire that user is warned that his/her password must be changed
  - **[Inactive]**: The number of days after password expires that account is disabled
  - **[Expire]:** days since Jan 1, 1970 that account is disabled i.e. an absolute date specifying when the login may no longer be used

Management User Accounts

- **Creating a user account:**

```
$ useradd [name]  [options]
```

Issuing command **$useradd user1** will create a user account named **user1** with default parameters contained in configuration file **/etc/default/useradd**

- **Set password**

```
$ passwd [username] [options]
```

Set or change password for user account

- **Deleting a user account:**

```
$ userdel [username] [options]
```

By default, **userdel** command only removes user from a system without deleting its home directory. In order to delete user with home directory, it is used option **-r**

Management User Accounts

- **Modifying a user account:**

```
$ usermod [options] [username]
```

Adding user to supplementary groups:
Use the combined **–aG** or **–append –groups** options, followed by a comma separated list of groups

```
$ usermod –aG wheel user1
```

Management User Groups

- **/etc/group** - contains group's name and lists of group members. Each record has the following format:

```
[Group name]:[Group password]:[GID]:[Group members]
```

  - **[Group name]** is the name of group.
  - An **x** in **[Group password]** indicates group passwords are not being used.
  - **[GID]**: same as in /etc/passwd.
  - **[Group members]**: a comma separated list of users who are members of **[Group name]**.

Management User Groups

- **Adding a new group:**

```
$ groupadd [group_name] [options]
```

- **Deleting user group:**

```
$ groupdel [group_name]
```

- **Modify a group definition:**

```
$ groupmod [options] [group_name]
```

Rename group name:
Below command will change the group **mygroup** to **mygroup_new** using -n option.

```
$ groupmod -n mygroup mygroup_new
```

Managing ownership

Anytime a user creates a new file or directory his account is assigned to that file or directory as “owner”.

```
[user1@centos7 ~]$ touch myfile.txt
[user1@centos7 ~]$ ls -l myfile.txt
-rw-rw-r-- 1 user1 user1 16 Sep 21 10:02 myfile.txt
[user1@centos7 ~]$
```

There is a possibility to specify a different user and/or group as the owner of a given file. To change the user who owns a file, you must be logged in as “root” or as the user with superuser privileges. To change the group that owns a file, you must be logged in as “root” or as the user with superuser privileges or as the user who currently owns the file.
It is used the following command for this purpose:

- **chown**
- **chgrp**

Managing ownership

- **chown** – is used to change the user and/or group ownership of a given file, directory or symbolic link.

```
$ chown [options] [USER][ :GROUP] FILE(s)
```

  - **[USER]** - is username of the new owner. If only the [USER] is specified, it will become the owner of the given file, the group ownership is not changed
  - **[USER:]** – when the username is followed by a colon and the group name is not given, the user will become the owner of the file and the file group ownership is changed to user's login group
  - **[USER:GROUP] –** if both user and group are specified the user ownership of the files is changed to the given user and the group ownership is changed to the given group.
  - **[:GROUP]** - If the user is omitted and the group is prefixed with a colon, only the group ownership of the files is changed to the given group

```
[root@centos7 ~]# chown user2 /home/user1/myfile.txt
[root@centos7 ~]# ls -l /home/user1/myfile.txt
-rw-rw-r-- 1 user2 user1 16 Sep 21 10:02 /home/user1/myfile.txt
[root@centos7 ~]#
```

Managing ownership

- **chgrp** – in addition to chown, it is also used chgrp to change the group ownership

```
$ chgrp [group_name] FILE(s)
```

```
[root@centos7 ~]# chgrp wheel /home/user1/myfile.txt
[root@centos7 ~]# ls -l /home/user1/myfile.txt
-rw-rw-r-- 1 user2 wheel 16 Sep 21 10:02 /home/user1/myfile.txt
[root@centos7 ~]#
```