1. **User and Group Management**

Just like in the Windows world Linux supports user and group creation. Similar to user Administrator from Windows, in Linux we have **root** which can do everything on the system. The password for user root is established at system installation.

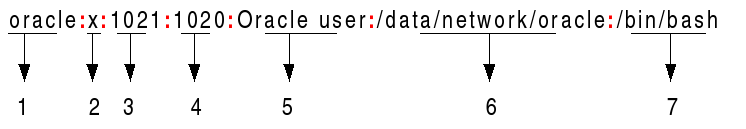
If humans use names when working with users on a Linux system, the Linux kernel refers to a user by it’s UID (user id). The user id for user root is 0, and the UID for non-privileged users start from UID 500 (on Redhat/Centos Linux distribution). To check the UID for a certain user just type **id <username>.**

Regarding groups under Linux, it works just like with users. You refer to a group name (for example: admins or dba) and kernel refers to it via GID (group ID).

In Windows, by default all user’s files are located in “Documents and Settings/Username/” folder. In Linux it is similar, all user’s files and directories are located in /home/username (/home is recommended to be the mountpoint for the partition dedicated to store user’s files.)

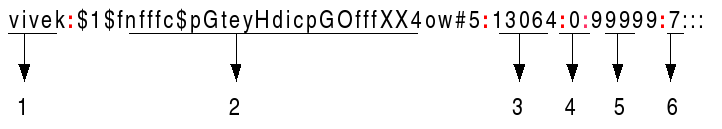
When managing users on Linux you have to get used with three files:

**/etc/passwd**

****

1. **Username**: It is used when user logs in. It should be between 1 and 32 characters in length.
2. **Password**: An x character indicates that encrypted password is stored in /etc/shadow file.
3. **User ID (UID)**: Each user must be assigned a user ID (UID). UID 0 (zero) is reserved for root and UIDs 1-99 are reserved for other predefined accounts. Further UID 100-999 are reserved by system for administrative and system accounts/groups.
4. **Group ID (GID)**: The primary group ID (stored in /etc/group file)
5. **User ID Info**: The comment field. It allow you to add extra information about the users such as user's full name, phone number etc. This field use by finger command.
6. **Home directory**: The absolute path to the directory the user will be in when they log in. If this directory does not exists then users directory becomes /
7. **Command/shell**: The absolute path of a command or shell (/bin/bash). Typically, this is a shell. Please note that it does not have to be a shell.

**/etc/shadow**

****

1. **User name**: It is your login name
2. **Password**: It your encrypted password. The password should be minimum 6-8 characters long including special characters/digits
3. **Last password change (lastchanged):** Days since Jan 1, 1970 that password was last changed
4. **Minimum**: 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
5. **Maximum**: The maximum number of days the password is valid (after that user is forced to change his/her password)
6. **Warn** : The number of days before password is to expire that user is warned that his/her password must be changed
7. **Inactive** : The number of days after password expires that account is disabled
8. **Expire** : days since Jan 1, 1970 that account is disabled i.e. an absolute date specifying when the login may no longer be used

**/etc/group**

cdrom:x:24:vivek,student13,raj

| | | |

1 2 3 4

1. **Group name**: It is the name of group. If you run ls -l command, you will see this name printed in the group field.
2. **Password**: Generally password is not used, hence it is empty/blank. It can store encrypted password. This is useful to implement privileged groups.
3. **Group ID (GID)**: Each user must be assigned a group ID. You can see this number in your /etc/passwd file.
4. **Group List**: It is a list of user names of users who are members of the group. The user names, must be separated by commas.

**Adding a user:**

[root@Oma.lan:/]# useradd -s /bin/bash -m -d /home/alex -c "Alex Istrate" alex

[root@Oma.lan:/]# passwd alex

**Deleting a user**

[root@Oma.lan:/]# userdel -r alex

**Changing a users password:**

[root@Oma.lan:/]# passwd alex

Enter new UNIX password:

**Adding group dba with GID 99 and inserting user alex in this group**

[root@Oma.lan:/]# groupadd –g 99 dba

[root@Oma.lan:/]# usermod –aG dba alex

**List password aging paramaters for a user:**

[root@Oma.lan:/]# chage -l alex

Last password change : Feb 13, 2011

Password expires : never

Password inactive : never

Account expires : never

Minimum number of days between password change : 0

Maximum number of days between password change : 99999

Number of days of warning before password expires : 7

**Modify user’s alex warn days to 3, Min days to 10, Max days to 12, expire date to 1 March 2011:**

[root@Oma.lan:/]# chage -m 10 -W 2 -M 12 -E 2011-03-01 alex

[root@Oma.lan:/]# chage -l alex

Last password change : Feb 13, 2011

Password expires : Feb 25, 2011

Password inactive : never

Account expires : Mar 01, 2011

Minimum number of days between password change : 10

Maximum number of days between password change : 12

Number of days of warning before password expires : 2

**Get information about user alex:**

[root@Oma.lan:/]# finger alex

Login: alex Name: Alex Istrate

Directory: /home/alex Shell: /bin/bash

Never logged in.

No mail.

No Plan.

[root@Oma.lan:/]#

[root@Oma.lan:/]# id alex

uid=1100(alex) gid=1100(alex) groups=1100(alex)

[root@Oma.lan:/]#

[root@Oma.lan:/]# cat /etc/passwd | grep alex

alex:x:1100:1100:Alex Istrate:/home/alex:/bin/bash

[root@Oma.lan:/]# cat /etc/shadow | grep alex

alex:$1$zmtVFLyl$vP6FhVHzvdsWIMybk0qbd.:15018:10:12:2::15034:

**Modify user’s home directory and shell:**

[root@Oma.lan:/]# mkdir /home/alex\_new

[root@Oma.lan:/]#

[root@Oma.lan:/]# usermod -s /bin/ksh -d /home/alex\_new alex

[root@Oma.lan:/]# finger alex

Login: alex Name: Alex Istrate

Directory: /home/alex\_new Shell: /bin/ksh

Never logged in.

No mail.

A cool thing present in Linux world is the possibility to switch from the current logged in user to another user by using the command **su**. This doesn’t involve knowing that user’s password if you are root(the superuser). Here is an example:

[root@Oma.lan:/]# id

uid=0(root) gid=0(root) groups=0(root)

[root@Oma.lan:/]# su - alex

[alex@Oma.lan:/home/alex\_new]$

[alex@Oma.lan:/home/alex\_new]$ id

uid=1100(alex) gid=1100(alex) groups=1100(alex)

[alex@Oma.lan:/home/alex\_new]$ su - root

Password:

[root@Oma.lan:/root]# id

uid=0(root) gid=0(root) groups=0(root)

Notice that when you are logged in with a non-privileged user, for example user alex, the shell prompt end with the character **$**. This character specifies that the user you are logged in with is a non-privileged one. When you su to user root notice that $ is being replaced with **#**, just to let you know that now you are logged in with the privileged user.

There are situations when an unprivileged user needs root rights to executer a certain script/program/binary. Linux solution for this situation is **sudo** (superuser do).