Manage local users and groups

Linux #redhat #users

User Account Management

Commands

- `useradd` - Create a new user
- `groupadd` - Create a new group
- `userdel` - Delete a user that is already created
- `groupdel` - Delete a group
- `usermod` - Modify a user

Whenever we create a new user, those user management are created, their record is maintained in three different files:

- `/etc/passwd`
- `/etc/group`
- `/etc/shadow`

Example:
`useradd –g superheros –s /bin/bash –c “user description” –m –d /home/spiderman spiderman`

- In this example the `useradd` command is being used to create a new user
- `–g` = Option to add a group
- `–s` = Option to give a shell environment
- `–c` = Define the user description
- `–m` & `–d` = To define the user home directory and the user itself (name of the user at the end)

In order to manage user and use user commands you have to be root

useradd

Example using `useradd` command:

```
[root@localhost ~]# useradd spiderman
```

- Creates a user named "spiderman"

How can you verify if the user has been created?

Example using `id` command:

```
[root@localhost ~]# id spiderman
```

- `id` shows the id assigned to the user "spiderman"
- The group is created and is also called "spiderman"
  - `id` also shows the group id of that user

Output:

```
uid=1001(spiderman) gid=1001(spiderman) groups=1001(spiderman)
```

An additional way of checking if the user you just created is to travel to the `/home/` directory and see if the user is inside this location. (Users are usually listed under the `/home/` directory)

Example using `cd` command:

```
[root@localhost ~]# cd /home/
```

Output from `ls -ltr` :

```
drwx------.  3 spiderman spiderman   78 Jun  9 16:08 spiderman
drwx------. 20 mmarin    mmarin    4096 Jun  6 15:29 mmarin
```

- The new user called `spiderman` is inside the `/home/` directory

---

groupadd

Example using `groupadd` command:

```
[root@localhost ~]# groupadd superheros
```

- This line adds a group called "superheros"

How can you verify if the group has been created?

Example using `cat` command:

```
[root@localhost ~]# cat /etc/group
```

- The `group` file from the `/etc` directory displays the list of groups available in your machine
- This line uses `cat` to access that list and show it to the terminal

Output:

```
...
tcpdump:x:72:
mmarin:x:1000:
spiderman:x:1001:
superheros:x:1002:
```

---

userdel

Example using `userdel` command:

```
[root@localhost ~]# userdel -r spiderman
```

- `userdel` deletes the user called "spiderman"
- `-r` specifier specifies to also remove its directory

groupdel

Example using command:

```
[root@localhost ~]# groupdel nonewgroup
```

- `groupdel` deletes a group called "nonewgroup"
- You can check if the group has been deleted by doing `cat /etc/group` and looking at the bottom of the list

usermod

To add a user to a certain group:
Example using `usermod` command:

```
[root@localhost ~]# usermod -G superheros spiderman
```

- `usermod` modifies a user
- The `-G` option specifies that we want to change the group from a user
  - Then we need to specify the name of the group, in this case "superheros"
  - Finally we specify the name of the user, in this case "spiderman"

How can you verify if the group of a user has been changed?

- Simply search for it in the `/etc/group` file.

Example using `grep` command:

```
[root@localhost ~]# grep spiderman /etc/group
```

- The `grep` command is used to search for terms inside files
  - It outputs the lines where they appear instead of having to read the whole file only for those lines

Output:

```
superheros:x:1002:spiderman
spiderman:x:1001:
```

- Note that the "superheros" group has a defined user inside called "spiderman"
- Also note that "spiderman" is also a name for the group of that specific use.

If you do `ls -ltr` inside the `/home` directory you can see that the group for "spiderman" user is still "spiderman".

- Why didn't it changed to the "superheros" group?
  - Because it's actual group is "spiderman" but always remain "spiderman", but it is also part of the other group as well, which is "superheros"
- If you want it to change, then you have to run the command `chgrp` to change the actual group of the user to a different one. Manage Files from the command line

Example using `chgrp` command:

```
[root@localhost ~]# chgrp superheros spiderman
```

- Note the order of inputs for the `chgrp` command
  - 1st specify the new group, "superheros"
  - 2nd specify the user from which you want to change its group, "spiderman"
- The name of the group has to be valid meaning that it has to reference an existing group. In other words, this command cannot create groups, only changes the group from files.

Output from `ls -ltr` inside `/home` :

```
total 4
drwx------. 20 mmarin    mmarin     4096 Jun  6 15:29 mmarin
drwx------.  3 spiderman superheros   78 Jun  9 18:14 spiderman
```

Check the `/etc/passwd` file

- You can use `cat /etc/passwd` to check the insides of this file
- Its contents identifies different fields for users

Example using `cat` command:

```
[root@localhost ~]# cat /etc/passwd
```

Output:

```
...
tcpdump:x:72:72::/:/sbin/nologin
mmarin:x:1000:1000:Maximus Marin:/home/mmarin:/bin/bash
spiderman:x:1001:1001::/home/spiderman:/bin/bash
```

- As you add a new user, a line inside the `/etc/passwd` file appears at the bottom
- We can see that fields are separated in columns by the ":"
  - 1st column: Name of the user
  - 2nd column: Tells you the password which is encrypted (it just gives you the 'x')
  - 3rd column: User id
  - 4th column: Group id
  - 5th column: Description (can sometimes be empty if not specified)
  - 6th column: The home directory of the user
  - 7th column: The shell that the user is using

Check the `/etc/group` file

Example using `cat` command:

```
[root@localhost ~]# cat /etc/group
```

Output:

```
...
tcpdump:x:72:
mmarin:x:1000:
spiderman:x:1001:
superheros:x:1002:
```

- We can see that fields are separated in columns by the ":" sign
  - 1st column: Name of the group
  - 2nd column: Tells you the group password which is encrypted (it just gives you the 'x')
  - 3rd column: Group id
  - 4th column: Tells you which users are part of that group

Check the `/etc/shadow` file

Example using `cat` command:

```
[root@localhost ~]# cat /etc/group
```

Output:

```
...
tcpdump:!!:19877::::::
```

```
mmarin:$6$xz1Oi0PAy2gUjKvi$xBxumvGCgx7nihxcll6NqW2tWjtLlNkHE0Kr8cySwiNh1xpHO3sRbSiRHtSA7KRfwfkG7jEdtAX
yTCfFTwoMo.::0:99999:7:::
spiderman:!!:19883:0:99999:7:::
```

- The `/etc/shadow` file is strictly for passwords of users that we create
- This does not mean that the password is explicitly found here
- The password is encrypted so we do not see it
- It also maintains some other parameters like password longevity and length, or restrictions.
- if you notice there is no encrypted password for "spiderman" in `/etc/shadow` file because we have not set the password for that user yet

You can also use the `grep` command to search the fields for a specific user in either `/etc/shadow` or `/etc/passwd`, or if it is a group in the `/etc/group`

Example using command:

```
[root@localhost ~]# grep spiderman /etc/passwd
```

- Looking for the term "spiderman" inside the `/etc/passwd` file

Output:

```
spiderman:x:1001:1001::/home/spiderman:/bin/bash
```

Adding users in the corporate way

Example using command:

```
[root@localhost ~]# useradd –g superheros –s /bin/bash –c “Ironman Character” –m –d /home/ironman
ironman
```

- In this example the `useradd` command is being used to create a new user
- `–g` = Option to add user to a group
  - in this case the user is being added to the "superheros" group
- `–s` = Option to give a shell environment
  - In this case it is specified to give the "/bin/bash" shell to the user
- `–c` = Define the user description
  - The description for this user is "Ironman character"
- `–m` & `–d` = To define the user home directory and the user itself (name of the user at the end).
  - The home directory given to the user is the "/home/ironman" directory
  - The user itself will be created with the name "ironman"

Creating a password for our user

Example using `passwd` command:

```
[root@localhost ~]# passwd ironman
```

- After running this command you need to type the password.

Output:

```
Changing password for user ironman.
New password:
```

- After specifying the password and hitting enter an alert of the type "BAD PASSWORD" could pop up.
  - E.g. `BAD PASSWORD: The password is a palindrome`
  - You can ignore this alerts
- Then you are asked to type the password a second time just to verify it.
- When the password is complete setting up the following line will be shown.
  - `passwd: all authentication tokens updated successfully.ca`

Enable Password Aging

The `/etc/login.def` file

The `chage` command – per user

- Use this command to specify those number of minimum days, maximum days, last day inactive, expiry date, and morning.
- You make that change to per user basis
- Example
  - `chage [-m mindays] [-M maxdays] [-d lastday] [-I inactive] [-Eexpiredate] [-W warndays] user`

What if you want to set those default values for every user that is created in your system? What if you have to create 10 users?

- For that there is a file called `/etc/login.defs` which stands for "login defaults"

File = /etc/login.defs

```
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_MIN_LEN    5
PASS_WARN_AGE   7
```

In order to check this file you have to be root

Example using `more` command:

```
[root@localhost ~]# more /etc/login.defs
```

- You can also use `cat` to check all of its contents in the terminal.
- Remember: The `more` command shows you the contents of a file one page at a time.

Output:

```
...
# Password aging controls:
#
#       PASS_MAX_DAYS   Maximum number of days a password may be used.
#       PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#       PASS_MIN_LEN    Minimum acceptable password length.
#       PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7

# Currently PASS_MIN_LEN is not supported
...

--More--(74%)
```

- You will see how everything that is set as a hash sign means it's just a comment.
- If you make changes in this part of the file and specify the default settings that is based on your corporate or your company requirement you could set that up and then next time you create a user account, we'll have that setting applied to every user account.
- There are other values that could set in this file as well.

Take as an example the following portion of the `/etc/login.defs` file

```
...
# Min/max values for automatic uid selection in useradd(8)
#
UID_MIN                  1000
UID_MAX                 60000
# System accounts
SYS_UID_MIN               201
SYS_UID_MAX               999
# Extra per user uids
SUB_UID_MIN              100000
SUB_UID_MAX             600100000
SUB_UID_COUNT            65536
...
```

- Every time you crate a user minimum max value for automatic UID selection `useradd`
- Whenever you run the `useradd` command the value or the UID is assigned by default to that user it starts with '1000'.
- UID stands for User ID
- If you wanted to make that more secure, sometimes seen in corporate environments, they start the user ID with 10000 or 20000 or a number that cannot be guessed easily.
- In this example you could have a total of 59,000 users.
- SYS_UID stands for System User ID for the systems processes. Anything that is set up as a system process user.
- There are other ones like Group User
  - `GID_MIN`
  - `GID_MAX`
  - `CREATE_HOME`
- Every time you run user ad command it'll create an ID and grop, and an encryption method that is set by default
  - `ENCRYPT_METHOD SHA512`
- `UMASK` is when you create a user, every time that user creates a file, that file is created with specific permissions and their permissions are actually assigned in this portion of the file. This is the max permission will be given to every user that is

created

This file is very important.

- It actually controls how the passwords age minimum and maximum is set for each user
- When someone is in the corporate environment and wants to tighten up the security, that person always have to start with the user level, and when you talk about creating users, the `/etc/login.defs` file is the one you really need to remember.

The `chage` Command

- Stands for "change age"
- Used to set the parameters around the password.
- It is not used to make any changes to the user account itself.
- To change or make any changes to the user account, you will use `usermod`.
- The `chage` command is per user basis.
- Use this command to specify those number of minimum days, maximum days, last day inactive, expiry date, and morning.
- Example
  - `chage [-m mindays] [-M maxdays] [-d lastday] [-I inactive] [-Eexpiredate] [-W warndays] user`
- `-d` = 3. **Last password change** (lastchanged): Days since Jan 1, 1970 that password was last changed
- `-m` = 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
- `-M` = 5. **Maximum**: The maximum number of days the password is valid (after that user is forced to change his/her password)
- `-W` = 6. **Warn**: The number of days before password is to expire that user is warned that his/her password must be changed
- `-I` = 7. **Inactive**: The number of days after password expires that account is disabled
- `-E` = 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.

The main reason to make changes to users passwords is to secure our environment.

Example using command:

```
[root@localhost ~]# chage -m 5 -M 90 -W 10 -I 3 babubutt
```

- In this line we are changing the parameters of the password for the user 'babubutt'
  - `-m` is setting the minimum (required) days to change the password again to 5 days.
  - `-M` is setting the maximum number of days before password has to be changed to 90 days.
  - `-I` is setting the number of days after password expires to disable that account to 3 days.

You can check changes done to the password of a user by looking for it in the `/tc/shadow` file

Example using `grep` command:

```
[root@localhost ~]# grep babubutt /etc/shadow
```

- Searching the user called "babubutt" in the `/etc/shadow` file

Output:

```
babubutt:$6$PmnObiclsElAntuV$dyE/n.r.iwgF87umFvKhtcQdwzFVdCRc8gwL8s80hQrpk2/6nQUdNfm9fjSveBTEGFKcw0tLS
NsCTiEUa0Sdq0:19884:5:90:10:3::
```

- All the parameters we set previously with the `chage` command appear here separated by ":" signs

Switch Users & `sudo` Access

- Sudo access is a command which allows an ordinary user to run root level commands

Commands

- `su - username` = Switch from one user to another (Replace "username" with the actual name of the user), To become a user, it will ask for its password.
- `sudo command` = If you do not have root privileges or cannot become root, then you will run `sudo` followed by the command
- `visudo` = Edits the "etc sudo" or file which is a configuration file that allows a user to add or remove the rights to run certain commands.

File

- `/etc/sudoers`
  - Remember: `/etc` is the folder or directory in Linux where all the configuration files are stored

If you are currently the root user and try to change to a different user the password will not be asked since you are the root user and have all rights. Root does not need any password to be authenticated.

The `su -` command

Example using `su -` command:

```
[mmarin@localhost ~]$ su - ironman
```

- Note the terminal is currently being used by the user called "mmarin"
- The `su -` command in this case is being used to change from the "mmarin" user to the "ironman"
- The "mmarin" user has not root access so the password for the "ironman" user will be asked.

Examples using commands with permission denied:
Example using `dmidecode` command:

```
[mmarin@localhost ~]$ dmidecode
```

Output:

```
# dmidecode 3.5
/sys/firmware/dmi/tables/smbios_entry_point: Permission denied
Scanning /dev/mem for entry point.
Can't read memory from /dev/mem
```

Example using `fdisk` command:

```
[mmarin@localhost ~]$ fdisk -l
```

- Command is used to check the disk size

Output:

```
fdisk: cannot open /dev/sda: Permission denied
fdisk: cannot open /dev/mapper/rhel-root: Permission denied
fdisk: cannot open /dev/mapper/rhel-swap: Permission denied
```

The `visudo` command

- Command to add a group to `/etc/sudo` file is `visudo`
- You have to become the root user in order to use this command or run the `sudo` command to give permission to your current user.

Example using `visudo` command:

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

File editor:

```
...
## The COMMANDS section may have other options added to it.
##
## Allow root to run any commands anywhere
root    ALL=(ALL)       ALL

## Allows members of the 'sys' group to run networking, software,
## service management apps and more.
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING, PROCESSES, LOCATE, DRIVERS

## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL

## Same thing without a password
# %wheel        ALL=(ALL)       NOPASSWD: ALL

## Allows members of the users group to mount and unmount the
## cdrom as root
# %users  ALL=/sbin/mount /mnt/cdrom, /sbin/umount /mnt/cdrom

## Allows members of the users group to shutdown this system
# %users  localhost=/sbin/shutdown -h now

## Read drop-in files from /etc/sudoers.d (the # here does not mean a comment)
#includedir /etc/sudoers.d
```

- This is only the final portion of the file
- The group named 'wheel' is the group that is automatically by default added to this file

    - It allows people in group 'wheel' to run all commands (ALL USERS)
    - So all we need to do is to add any user we want into the 'wheel' group to allow it to run all commands
- If you would like to do this at the user level, meaning you only want this user to be able to run any commands without having to add the user to any group you can input the user name and parameter below the following lines of the file:

File editor:

```
## Allow root to run any commands anywhere
root    ALL=(ALL)       ALL
mmarin  ALL=(ALL)       ALL
```

- The last line is allowing the user called "mmarin" to run any commands anywhere.
- This is done at the user level
- Remember: to save the file and quit to the terminal use `:wq!` (bang) when not in INSERT mode.

Add a user to 'wheel' group:
Example using `usermod` command:

```
[root@localhost ~]# usermod -aG wheel mmarin
```

- Remember: The `usermod` command is used to modify a user
- The `-a` option stands for "append" which adds the user to the supplementary group(s). Use only with the -G option.
- The `-G` Option is adding the user "mmarin" to the group 'wheel'
    - If the `-a` Option is not specified, the `usermod` command will remove the user from any current groups that the user is member of and only add the new group.
- After adding a user to the 'wheel' group, this user will have all rights to use the `sudo` command to run any other command anywhere.

The `sudo` command

Example using `sudo` command:

```
[mmarin@localhost ~]$ sudo dmidecode
```

- `sudo` is being used to run the `dmidecode`, which is a root only command, while still being the "mmarin" user
- The `dmidecode` command won't run if we didn't use the `sudo` command as well.

Output:

```
...
        Header and Data:
                80 08 04 00 25 42 32 00

Handle 0xFEFF, DMI type 127, 4 bytes
End Of Table
```

Example using `fdisk` command:

```
[mmarin@localhost ~]$ sudo fdisk -l
```

- `sudo` is being used to run the `fdisk -l`, which is a root only command, while still being the "mmarin" user
- `fdisk -l` Tells you the size of your disks and all the partition information
  - You could only view that information if you are root.

Output:

```
...
Disk /dev/mapper/rhel-swap: 2 GiB, 2147483648 bytes, 4194304 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```