**MANAGING LOCAL LINUX USERS AND GROUPS**

## **WHAT IS A USER ?**

* Every process on the system runs as a particular user. Every file is owned by a particular user. Access to files and directories are restricted by user. The user associated with a running process determines the files and directories accessible to that process.

## **RANGES OF USERS ID [UID]**

The root account has the awesome privilege of having UID = 0 and GID = 0

UID : 0 Root account

UID : 1-200 System users assigned to system process by redhat

UID : 201-999

* System users. These are users that do not map to actual “human” users, but are used as security identities for system daemons, to implement privilege separation and run system daemons with minimal privileges.
* Range of “system users” used by system processes that do not own files on the file system. programs run as these “unprivileged” system users in order to limit their access to just the resource they need to function.

UID : 1000+   Range available for assignment to regular users.

## **CHECK NEXT AVAILABLE UID BETWEEN 1000 & 10000**

$ awk -F: '{uid[$3]=1}END{for(x=1000; x<=10000; x++) {if(uid[x] != ""){}else{print x; exit;}}}' /etc/passwd

## **TYPES OF USER ACCOUNTS**

Accounts

* User account
* Root account
* System account

## **ROOT ACCOUNT**

The **root user**, also known as the**superuser** or **administrator**, is a special user account in Linux used for system administration. It is the most privileged user on the Linux system and it has access to all commands and files. The**root** user can do many things an ordinary user cannot, such as installing new software, changing the ownership of files, and managing other user accounts.

Any account with [**user id**](https://www.ssh.com/iam/user/id) 0 is a root account, regardless of the name.

## **SYSTEM ACCOUNT**

A system account is a user account that is created by an operating system during installation and that is used for operating system defined purposes.

## **USER ACCOUNTS**

* User accounts provide interactive access to the system for users and groups of users. General users are typically assigned to these accounts and usually have limited access to critical system files and directories.

## **USER ADMIN / CONFIG FILES**

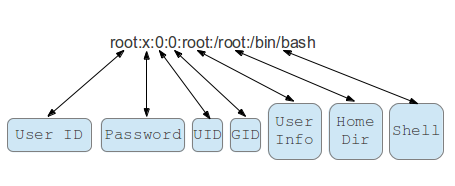
**/etc/passwd /etc/shadow /etc/group /etc/gshadow /etc/login.defs**

* **/etc/passwd**

Keeps the user account and password information. This file holds the majority of information about accounts on the Unix system.

767

Plain text-based database that contains information for all user accounts .



**Login name**: The unique name given to a user. It is case sensitive.

**Encrypted Password**: This field is for the encrypted password. However, in most modern Linux systems, the password is stored in a shadow file named /etc/shadow. The **‘x’**in the second field represents that the password is stored in a shadow file.

**UID number**: A unique ID assigned for each user. The system identifies a user by its UID number. The root user gets UID 0 which is reserved. UID 1-99 are reserved for predefined accounts. These are also called fake logins. In most Linux systems UID for new users starts from 1000.

**GID number**: GID is used to identify the group user belongs to. Information about groups is stored in /etc/group file. In Linux, a user can belong in up to 16 groups.

**GECOS**: stands for General Electric Comprehensive Operating Supervisor. This field allows the admin to store additional information about the user. This field can have multiple entries in a comma-delimited list. Information such as full name, phone number, the designation can be stored in this field.

**Home directory:**This field contains the absolute path to the user’s home directory. When a user logs in, the system puts him/her directly in his/her home directory.

**Login shell**: This field is for specifying the program that will run automatically whenever the user logs in. It’s typically a command interpreter (shell).

* **/etc/shadow**

Holds the encrypted password of the corresponding account. Not all the systems support this file.

File stores actual password in encrypted format (more like the hash of the password) for user’s account with additional properties related to user password.

mark:$6$.n.:17736:0:99999:7:::

[--] [----] [---] - [---] ----

| | | | | |||+-----------> 9. Unused

| | | | | ||+------------> 8. Expiration date

| | | | | |+-------------> 7. Inactivity period

| | | | | +--------------> 6. Warning period

| | | | +------------------> 5. Maximum password age

| | | +----------------------> 4. Minimum password age

| | +--------------------------> 3. Last password change

| +---------------------------------> 2. Encrypted Password

+----------------------------------------> 1. Username

**Username**. The string you type when you log into the system. The user account that exist on the system

**Encrypted Password**. The password is using the $type$salt$hashed format. $type is the method cryptographic hash algorithm and can have the following values:

If the password field contains an asterisk (\*) or exclamation point (!), the user will not be able to login to the system using password authentication. Other login methods like [key-based authentication](https://linuxize.com/post/how-to-setup-passwordless-ssh-login/) or [switching to the user](https://linuxize.com/post/su-command-in-linux/) are still allowed.

* $1$ – MD5
* $2a$ – Blowfish
* $2y$ – Eksblowfish
* $5$ – SHA-256
* $6$ – SHA-512

**Last password change**. This is the date when the password was last changed. The umber of days is counted since January 1, 1970 (epoch date).

**Minimum password age**. The number of days that must pass before the user password can be changed. Typically it is set to zero, which means that there is no minimum password age.

**Maximum password age**. The number of days after the user password must be changed. By default, this number is set to 99999.

**Warning period**. The number of days before the password expires during which the user is warned that the password must be changed.

**Inactivity period**. The number of days after the user password expires before the user account is disabled. Typically this field is empty.

**Expiration date**. The date when the account was disabled. It is represented as an epoch date.

Unused. This field is ignored. It is reserved for future use.

The /etc/shadow file should not be edited by hand unless you know what you are doing. Always use a command that is designed for the purpose. For example, to change a user password, use the [passwd](https://linuxize.com/post/how-to-change-user-password-in-linux/) command, and to change the password aging information, use the chage command.

* /**etc/login.defs**

The /etc/login.defs file provides default configuration information for several user account parameters. The useradd, usermod, userdel, and groupadd commands, and other user and group utilities take default values from this file. Each line consists of a directive name and associated value.

This file defines the site-specific configuration for the shadow password suite. This file is required. Absence of this file will not prevent system operation, but will probably result in undesirable operation.

* /etc/pam.d/system-auth
* UMASK
* UID | GID
* Create Home
* Encrpt method
* Password ageing control
* Mail Directory
* **/etc/group** This file contains the group information for each account.
* **/etc/gshadow** − This file contains secure group account information.

## **USERADD [USER CREATION]**

**SYNTAX: $ useradd [options] [username]**

**Options:**

* -d home directory of the new account
  + - Create a User with Different Home Directory  
      By default ‘useradd‘ command creates a user’s home directory under /home directory with username
* -p encrypted password of the new account $ useradd venkat -p password
* -s login shell of the new account
* -u UID user ID of the new account
* -g name or ID of the primary group of the new account
* -G list of supplementary groups of the new account
* -M do not create the user's home directory
* -e xpiration date of the new account [YYYY-MM-DD] By default never expire
* -c GECOS field of the new account
* -k to create custom skeleton directory
* -N do not create a group with the same name as user
* -r create a system account
* -f password inactivity period of the new account

**How to Add a New User in Linux ?**

**$ useradd username**

**Create a User with Different Home Directory**

* + $ useradd venkat -d /data/projects

**Create a User with Specific User ID**

$ useradd venkat -u 999

**Create a User with Specific Primary Group ID**

$ useradd venkat -u 999 -g 1000

**Add a User to Multiple Secondary Groups**

$ useradd venkat -G admins,webadmins,developers

**Add a User without Home Directory**

$ useradd venkat -M

**Create a User with Account Expiry Date**

**$** useradd -e 2014-03-27 venkat # [YYYY-MM-DD]

**Create a User with Password Expiry Date**

$ useradd -e 2014-04-27 -f 45 tecmint

**Add a User with Custom Comments**

$ useradd venkat -c “content”

**Change User Login Shell**

* $ user add venkat -s /sbin/nologin

## **USERMOD [USER MODIFICATION]**

* The command ‘**usermod**‘ is used to modify or change any attributes of a already created user account via command line.

**SYNTAX : $ usermod [option] username**

**-c** = We can add comment field for the useraccount.

**-d** = To modify the directory for any existing user account.

**-e** = Using this option we can make the account expiry in specific period.

**-g** = Change the primary group for a User.

**-G** = To add a supplementary groups.

**-a** = To add anyone of the group to a secondary group.

**-l** = To change the login name from tecmint to tecmint\_admin.

**-L** = To lock the user account. This will lock the password so we can’t use the account.

**-m** = moving the contents of the home directory from existing home dir to new dir.

**-p** = To Use un-encrypted password for the new password. (NOT Secured).

**-s** = Create a Specified shell for new accounts.

**-u** = Used to Assigned UID for the user account between 0 to 999.

**-U** = To unlock the user accounts. This will remove the password lock and allow us to use the user account.

**Adding Information to User Account**

$ usermod -c “This is administration” venkat

It can be viewed # grep -E –-color ‘venkat’ /etc/passwd

**Change users home directory**

$ usermod venkat -d /new/home

**Set User Account Expiry date**

$ chage -l venkat check current account expiry date

$ usermod venkat -e YYYY-MM-DD

**Change user primary group**

$ id username #check the current group for the user

$ usermod venkat -g primarygroup #make sure group is already exist

**Adding Secondary group to existing user**

$ usermod venkat -G secondary\_grp\_name

**So, always add the ‘-a‘ (append) with ‘-G‘ option to add or append new groups.**

**Appending the user to secondary group**

$ usermod venkat -a -G secondary\_grp

**User Rename**

$ usermod venkat -l NEW\_NAME

**Lock user account**

$ usermod venkat -L

After the account is locked we can’t login by using the password and you will see a **!** added before the encrypted password in **/etc/shadow**

**Check : $ cat /etc/shadow | grep username**

**Unlock user account**

Check: $ cat /etc/shadow | grep username

$ usermod -U username

**Move users home\_directory**

Check current home directory : $ cat /etc/passwd | grep username

$ usermod username -md /new/directory

**Un-encrypted password [Not recommended]**

$ usermod username -p password

Check : $ cat /etc/shadow | grep username

**Change user shell**

$ usermod username -s /bin/sh

**Change UID**

We can assign UID between **0** to **999**

$ usermod username -u 888

## **USERDEL [USER DELETION]**

**SYNTAX: $ userdel [options] username**

First lock the user account $ usermod -l username

Find and kill all running process of user

$ pgrep -u username

$ ps -f –pid $(pgrep -u username)

$ killall -9 -u username

Backup user data before deleting

* + - $ tar jcvf /user-backup/tech.tar.bz2 /home/username

**$ userdel username**

* The system will have files that are owned by an unassigned user ID number. This can also happen when files created by a deleted user exist outside that users home directory.

**To find “unowned files and directories by running**

* $ find / nouser -o -nogroup 2> /dev/null

**Delete user with home directory**

**$ userdel -r username**

**-f force**

## **MANAGING USERS PASSWORD**

The **passwd** command changes passwords for [user](https://www.computerhope.com/jargon/u/user.htm) accounts. A normal user can only change the password for their account, but the superuser can change the password for any account. **passwd** can also change or reset the account's validity period — how much time can pass before the password expires and must be changed.

SYNTAX : passwd [options] [username]

-k keep non-expired authentication tokens

-d delete the password for the named account

-l lock the password for the named account (root only)

-u unlock the password for the named account (root only)

-e expire the password for the named account (root only)

-f force operation

-x maximum password lifetime (root only)

-n minimum password lifetime (root only)

-w number of days warning users receives before password expiration

-I number of days after password expiration when an account becomes disabled

-S report password status on the named account (root only)

**Change Password of System Users**

$ passwd venkat

**Display Password Status Information.**

**$ passwd -S username**

**Removing Password of a User using -d option**

**$ passwd -d venkat**

**Set Password Expiry Immediately**

**$ passwd -e venkat**

This will force the user to change the password in the next login.

**Lock the password of System User**

**$ passwd -l venkat**

**Unlock User’s Password using -u option**

**$ passwd -u venkat**

**Setting inactive days using -i option**

**$ passwd -I venkat**

* **-i** option in passwd command is used to set inactive days for a system user. This will come into the picture when password of user ( in my case linuxtechi) expired and user didn’t change its password in ‘**n**‘ number of days ( i.e 10 days in my case)  then after that user will not able to login.

**Set Minimum Days to Change Password using -n option**

**$ passwd -n 90 venkat**

**Set Warning days before password expire using -w option**

**$ passswd -w 12 venkat**

## **CHAGE [PASSWORD AGING]**

The **chage command** is used to modify user password expiry information. It enables you to view user account aging information, change the number of days between password changes and the date of the last password change.

-d set date of last password change to LAST\_DAY

-E set account expiration date to EXPIRE\_DATE

-h display this help message and exit

-I set password inactive after expiration to INACTIVE

-l[small ell] show account aging information

-m set minimum number of days before password change to MIN\_DAYS

-M set maximum number of days during which a password is valid

-R directory to chroot into

-W set expiration warning days to WARN\_DAYS

**How to view current password expiry info?**

**$ chage -l venkat**

**How to change password expiry date?**

**$** *chage -M 1000 himanshu*

**How to change last password change date?**

**$** *chage -d 0 venkat #wil force passwd update on nxt login*

*$ change -d YYYY-MM-DD venkat*

**How to warn the user before password expires?**

*$ chage -W 10 himanshu*

**How to set Account Expire date ?**

*$ chage -E 2019-06-21 himanshu*

## **WHAT IS GROUP ?**

group is a collection of users. The main purpose of the groups is to define a set of privileges like read, write, or execute permission for a given resource that can be shared among the users within the group.

## **TYPES OF GROUP**

Every file is owned by one user and one group, no more, no less. So when you create a file it will be owned by your user and current group. When a user logs in their current group will be their primary group.

**Primary or login group**

* The **primary group** is the one that’s recorded in the **/etc/passwd** file.
* Each user can belong to exactly one primary group
* By default primary group owns new files created by the user
* The name of the primary group is the same as the name of the user. Each user must belong to exactly one primary group.

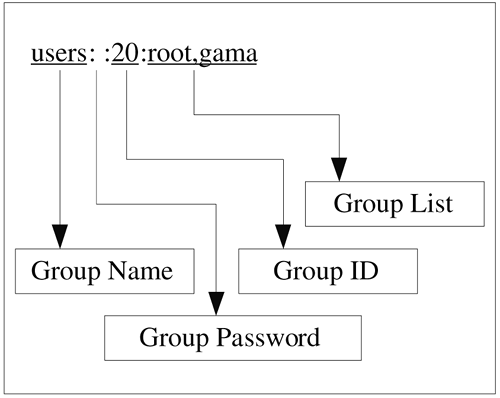
**Secondary or supplementary group**

* Secondary group memberships show up in the /etc/group file.
* Users can belong to up to 15 secondary groups.
* Used to grant certain privileges to a set of users. A user can be a member of zero or more secondary groups.

## **GROUP CONFIGURATION FILE**

* **/etc/group**

This file contains the Secondary group information for each account.



**Group name** — The name of the group. Used by various utility programs as a human-readable identifier for the group.

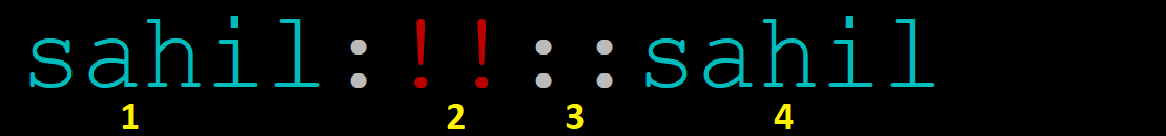
**Group password** — If set, this allows users that are not part of the group to join the group by using the newgrp command and typing the password stored here. If a lower case x is in this field, then shadow group passwords are being used.

**Group ID (GID)** — The numerical equivalent of the group name. It is used by the operating system and applications when determining access privileges.

**Member list** — A comma delimited list of the users belonging to the group.

* **/etc/gshadow**

The **/etc/gshadow** file is readable only by the root user and contains an encrypted password for each group, as well as group membership and administrator information. The information about each group’s separated by a colon.



## **GROUPADD [MANAGING GROUPS]**

*SYNTAX : $ groupadd [options[ groupname*

-f exit successfully if the group already exists and cancel -g if the GID is already used

-g use GID for the new group

-K override /etc/login.defs defaults

-o allow to create groups with duplicate

-p use this encrypted password for the new group

-r create a system account

-R directory to chroot into

-P directory prefix

Create a new Linux group

*$ groupadd group\_name*

*$ cat /etc/group | grep venkat ##verification*

Create new group with a specific groupid

*$ groupadd group\_name -g 1001*

Creating System Group [Range o to 999]

*$ groupadd groupname -r*

Create group with password

*$ groupadd groupname -p password*

Override /etc/login.defs default

*$ groupadd groupname -K GID\_MIN=8000 -K GID\_MAX=9999*

## **GROUPMOD [GROUP MODIFICATION]**

The **groupmod** command modifies the definition of the specified group by modifying the appropriate entry in the files **/etc/group**, **/etc/gshadow**, and **/etc/passwd**

*SYNTAX $ groupmod [options] group\_name*

-g change the group ID to GID

-h display this help message and exit

-n change the name to NEW\_GROUP

-o allow to use a duplicate (non-unique) GID

-p change the password to this (encrypted)

-R directory to chroot into

-P prefix directory where are located the /etc/\* files

Change the group name

*$ groupmod old\_group -n new\_group\_name*

Change groupid of a group

*$ groupmod group\_name -g 777*

Having duplicate group id

*$ groupmod group\_name1 -g 777 -o*

## **GROUPDEL [GROUP DELETION]**

groupdel command is used to delete a existing group. It will delete all entry that refers to the group, modifies the system account files, and it is handled by superuser or root user.

SYNTAX

*$ groupdel [options] groupname*

*-f force*

Delete group

*$ groupdel group\_name*

## **GPASSWD**

gpasswd command is used to administer the /etc/group and /etc/gshadow. As every group in Linux has administrators, members, and a password

SYNTAX

*$ gpasswd [options] groupname*

*-a add USER to GROUP*

*-d delete USER from GROUP*

*-h display this help message and exit*

*-Q directory to chroot into*

*-r remove the GROUP's password*

*-R restrict access to GROUP to its members*

*-M set the list of members of GROUP*

*-A set admin users*

Add user to the group

*$ gpasswd groupname -a username*

Delete user from the group

*$ gpasswd groupname -d username*

Set user as group Admin

*$ gpasswd groupname -A username*

Set password

*$ gpasswd groupname*

## **SWITCHING USER [SU]**

Utility allows you to run commands with another user’s privileges, by default the root user.

**SYNTAX**

*$ su [OPTIONS] [USER [ARGUMENT...]]*

*-m, -p, --preserve-environment do not reset environment variables*

*-g, specify the primary group*

*-G, specify a supplemental group*

*-, -l, --login make the shell a login shell*

*-c, pass a single command to the shell with -c*

*-f, --fast pass -f to the shell (for csh or tcsh)*

*-s, --shell <shell> run <shell> if /etc/shells allows it*

*-P, --pty create a new pseudo-terminal*

*--session-command <command> pass a single command to the shell with -c*

*and do not create a new session*

***SU VS SU –***

*su - invokes a login shell after switching the user. A login shell resets most environment variables, providing a clean base.*

su – *changes our path to the path of the target user, while*su *doesn’t*

*su just switches the user, providing a normal shell with an environment nearly the same as with the old user.*

*When a new terminal starts in Linux, operating system creates a new shell session. Unless we specify otherwise,*[*Bash*](https://www.man7.org/linux/man-pages/man1/bash.1.html)*is used by default for almost all of the distributions.*

*When a shell starts, it prepares an environment for itself. The environment acts like small key-value storage that holds the basic information that some of the commands need to work.*

*For example, the*[pwd](https://man7.org/linux/man-pages/man1/pwd.1.html)*command on Linux-based systems reads the current working directory from the*PWD*environment variable. The*cd – *command needs the*OLDPWD *variable to find the last visited directory*.*When we run any command on the shell, history about that command gets written to file at the path*HISTFILE.*Environment variables manage a lot of information for us in the background.*

[*The shell creates the environment variables*](https://www.baeldung.com/linux/environment-variables)*by invoking specific scripts. Not all shells have identical environment variables, different shell types might invoke different scripts or same scripts with a different order.*

***Switch to a Different User***

*$ su -l username*

*$ su - username*

***Switch to a Root user***

*$ su –*

*$ su*

***Run Specific Command as a Different User***

*$ su –c [command] [other\_user]*

*$ sudo su -l arjun -c “ls -lrt”*

***Use Diferent Shell***

*$ sudo su –l arjun -s /bin/sh*

*$ sudo su -s /bin/sh ##default root user*

## **SUDO ACCESS**

*Sudo stands for****SuperUser DO****and is used to access restricted files and operations. By default, Linux restricts access to certain parts of the system preventing sensitive files from being compromised.*

*The sudo command temporarily elevates privileges allowing users to complete sensitive tasks without logging in as the root user.*

*SUDO configuration file : /etc/sudoers*

***SYNTAX***

*$ sudo <command>*

*Benefits*

*One additional benefits to using sudo is that all commands executed using sudo are logged by default to /var/log/secure.*

## **What is VISUDO ? | SUDO CONFIG FILE**

* */etc/sudoers*
* */etc/sudoers.d*
* *The sudo command is configured through a file located at /etc/sudoers.*
* */etc/sudoers by default is the list os who can run what !*
* *Warning: Never edit this file with a normal text editor! Always use the visudo command instead!*
* *Because improper syntax in the /etc/sudoers file can leave you with a broken system where it is impossible to obtain elevated privileges, it is important to use the visudo command to edit the file.*
* *The visudo command opens a text editor like normal, but it validates the syntax of the file upon saving. This prevents configuration errors from blocking sudo operations, which may be your only way of obtaining****root****privileges.*

*$ visudo opens sudoers file safely*

*$ visudo -f /etc/sudoers.d/veracrypt*

## **/etc/sudoers**

SYNTAX

*User Host = (Runas) Command*

*User be usernames and group name*

*group names prefixed with %*

*Runas is target users and groups*

*Host may be a hostname, IP address, or a whole network (e.g., 192.0.2.0/24), but not 127.0.0.1.*

**The *sudoers* file is composed of two types of entries:**

* aliases (basically variables) and
* user specifications (which specify who may run what)

Four types of Alias

* User\_Alias Defines groups of user [Individual name | groups | other user alias]
* Runas\_Alias Defines target users [User | groups]
* Host\_Alias Defines hostname/IP
* Cmnd\_Alias Defines list of command

SYNTAX

Alias\_Type NAME = item1, item2, ...

**# User alias specification**

* User\_Alias FULLTIMERS = millert, mikef, dowdy
* User\_Alias PARTTIMERS = bostley, jwfox, crawl
* User\_Alias WEBMASTERS = will, wendy, wim

**# Runas alias specification**

* Runas\_Alias OP = root, operator
* Runas\_Alias DB = oracle, sybase

**# Host alias specification**

* Host\_Alias SPARC = bigtime, eclipse, moet, anchor:\
  + - SGI = grolsch, dandelion, black:\
    - ALPHA = widget, thalamus, foobar:\
    - HPPA = boa, nag, python
* Host\_Alias CUNETS = 128.138.0.0/255.255.0.0
* Host\_Alias CSNETS = 128.138.243.0, 128.138.204.0/24
* Host\_Alias SERVERS = master, mail, www, ns
* Host\_Alias CDROM = orion, perseus, hercules

**# Cmnd alias specification**

* Cmnd\_Alias DUMPS = /usr/sbin/dump, /usr/sbin/rdump,
* Cmnd\_Alias KILL = /usr/bin/kill
* Cmnd\_Alias PRINTING = /usr/sbin/lpc, /usr/bin/lprm
* Cmnd\_Alias SHUTDOWN = /usr/sbin/shutdown
* Cmnd\_Alias HALT = /usr/sbin/halt
* Cmnd\_Alias REBOOT = /usr/sbin/reboot
* Cmnd\_Alias SHELLS = /sbin/sh, /usr/bin/sh, /usr/bin/csh, /usr/bin/ksh,
* Cmnd\_Alias SU = /usr/bin/su
* Cmnd\_Alias VIPW = /usr/sbin/vipw, /usr/bin/passwd, /usr/bin/chsh, \

**# User specification**

**# root and users in group wheel can run anything on any machine as any user**

root ALL = (ALL) ALL

%wheel ALL = (ALL) ALL

**# full time sysadmins can run anything on any machine without a password**

FULTIMERS ALL=(ALL) NOPASSWD: ALL

**# part time sysadmins may run anything but need a password**

PARTTIMERS ALL = ALL

**# jack may run anything on machines in CSNETS**

jack CSNETS = ALL

**Defaults**

**Set a Custom log path**

**Defaults logfile=”/var/log/sudo.log”**

By default, sudo logs through syslog(3). However, to specify a custom log file, use the logfile parameter

To log hostname and the four-digit year in the custom log file, use **log\_host** and **log\_year** parameters

**Defaults log\_host, log\_year, logfile="/var/log/sudo.log"**

**Lecture sudo users**

**Defaults lecture="always" always or never or once**

Additionally, you can set a custom lecture file with the **lecture\_file** parameter, type the appropriate message in the file

**Defaults lecture\_file="/path/to/file"**

**Show custom Message when you enter wrong sudo passwd**

When a user enters a wrong password, a certain message is displayed on the command line. The default message is “sorry, try again”, you can modify the message using the badpass\_message parameter as follows

**Defaults badpass\_message="Password is wrong, please try again"**

**Increase sudo password tries limit**

The parameter passwd\_tries is used to specify the number of times a user can try to enter a password.

The default value is 3:

**Defaults passwd\_tries=5**

## **/etc/sudoers.d**

**SYNTAX**

$ visudo -f filename

$ visudo -f /etc/sudoers.d/networking

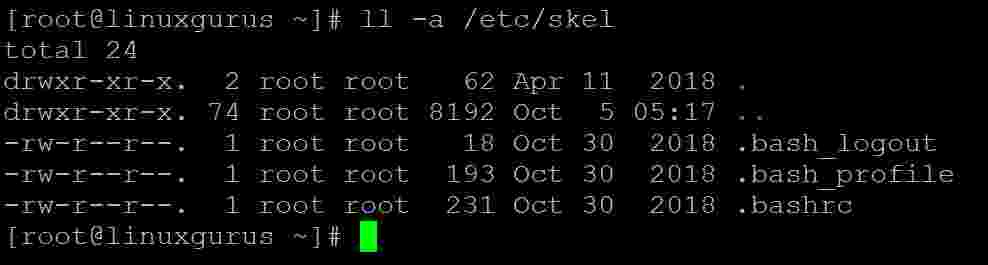
Unlike /etc/sudoers, the contents of /etc/sudoers.d survive system upgrades, so it's preferrable to create a file there than to modify /etc/sudoers

#includedir /etc/sudoers.d present in sudoers file. This ensures that the #includedir directive **is not a comment**. It has the effect of causing sudo to also read and parse any files in the /etc/sudoers.d directory (that do not end in '~' or contain a '.' character)

## **/etc/skel**

The**/etc/skel** directory contains files and directories that are automatically copied over to a new user’s [**home directory**](https://www.linuxgurus.in/linux-home-directory/) when such a user is created by the useradd program. skel is derived from the “skeleton”. Below is shown a picture.

These files are known as **skeleton files**. Typically, they’re hidden files that affect the user environment, such as**.bash\_history**, which records each command you run, and **.bashrc**, which sets the default values used with the **bash shell**.



**Note:** “**skeleton**” directory is define in **/etc/default/useradd** file

Below you can see the picture of **/etc/default/useradd** file which defines the skel directory. You can change the default location**/etc/skel** to any other location.

**cat /etc/default/useradd**



You can also change the default base **/home** directory to any other location. For Example (HOME=/home, you can change the location to any other location like **/file/home** you need to change the value **HOME=/file/home**)

The default permission of /etc/skel directory is **drwxr-xr-x**.

It is not recommended to change the permission of the skel directory or its contents. Changing the permission may break some of the programs because in the skel directory some profiles need the permission of ‘read’ and trying to permit it to execute will cause some programs/profiles to work unexpectedly.