Linux users and groups Accounts and groups

START HERE

`id <user>`
UID, GID and groups

`getent passwd <user>`
Look up an account

`sudo useradd -m -s /bin/bash <user>`
Create a user with home

`sudo passwd <user>`
Set their password

`sudo usermod -aG <group> <user>`
Add to a group, keep others

`sudo chage -l <user>`
Password and expiry dates

`sudo usermod -L -e 1 <user>`
Lock an account fully

`sudo userdel -r <user>`
Delete user and home

useradd -m defaults from login.defs → **writes** **/etc/passwd** name, UID, home, shell → **plus** **/etc/shadow** hash, aging, expiry → **plus** **/etc/group** own group, same name → **copies** **/home/<user>** from /etc/skel

CONCEPTS

usermod -G alone replaces all supplementary groups. Always write -aG to add one. Group changes apply from the next login.

ID ranges (typical)

Range	Used for
`0`	root
`1-999`	system users (-r)
`1000+`	people
`65534`	nobody

Only root's UID 0 matters to the kernel, not the name. Names are looked up via NSS, so LDAP or SSSD users appear in getent but not in /etc/passwd.

WHO IS WHO

`id`
Your UID, GID and groups

`id <user>`
Same for another user

`groups <user>`
List a user's groups
Shows the groups in the files, not those of your running session.

`whoami`
Print your effective user name

`getent passwd <user>`
Look up an account entry
Uses NSS, so it also finds LDAP/SSSD users.

`getent group <group>`
Show a group and its members
Lists only supplementary members, not users with it as primary group.

`who`
Users logged in now

`w`
Logged in users and what they run

`last -n 10`
Recent logins
From wtmp. Some newer distros ship it as wtmpdb last.

CREATE USERS

`useradd -m <user>`
Create a user with a home dir
Without -m many distros create no home. Set a password next.

`useradd -m -s /bin/bash <user>`
Set the login shell too
Debian's default shell for useradd is /bin/sh.

`useradd -m -G <g1>,<g2> <user>`
Add supplementary groups

CREATE USERS (cont.)

`useradd -m -u <uid> <user>`
Pick a fixed UID

`useradd -g <group> <user>`
Set the primary group

`useradd -r -s /usr/sbin/nologin <user>`
Create a system user for a service
UID from the system range; no home unless you add -m.

Debian and Ubuntu also offer adduser, an interactive wrapper that creates the home and asks for a password. Scripts should use useradd.

MODIFY USERS

`usermod -aG <group> <user>`
Add to a group, keep the rest

`usermod -G <g1>,<g2> <user>`
Replace all supplementary groups
Drops every group not listed.

`gpasswd -d <user> <group>`
Remove a user from one group

`usermod -s <shell> <user>`
Change the login shell

`usermod -L <user>`
Lock the password
SSH keys still work. Add -e 1 to block all logins.

`usermod -U <user>`
Unlock the password

`usermod -e <YYYY-MM-DD> <user>`
Set the account expiry date
-e '' removes it again.

`usermod -l <new> <old>`
Rename the login
Home dir and group keep their old names.

`usermod -d <dir> -m <user>`
Move the home directory

DELETE USERS

`userdel <user>`
Delete the account, keep files

`userdel -r <user>`
Delete account, home and mail
Files outside the home stay, owned by a bare UID.

`find / -xdev -nouser`
Find files of deleted users

GROUPS

`groupadd <group>`
Create a group

GROUPS (cont.)

`groupadd -g <gid> <group>`
Create a group with a fixed GID

`groupadd -r <group>`
Create a system group

`gpasswd -a <user> <group>`
Add a user to a group

`groupdel <group>`
Delete a group
Fails while it is someone's primary group.

`newgrp <group>`
Start a shell with a new group
Picks up a fresh membership without logging out.

PASSWORDS AND AGING

`passwd`
Change your own password

`passwd <user>`
Set another user's password

`passwd -S <user>`
Password status: P, L or NP
P usable, L locked, NP none; then last change and ages.

`passwd -l <user>`
Lock the password

`passwd -u <user>`
Unlock the password

`passwd -e <user>`
Force a change at next login

`chage -l <user>`
Show aging and expiry info

`chage -M <days> <user>`
Max days before a change

`chage -E <YYYY-MM-DD> <user>`
Expire the account on a date

`chage -E -1 <user>`
Remove the account expiry

Lock or expire?

Command	Blocks
`passwd -l`	password logins
`usermod -L`	password logins
`usermod -e 1`	all logins, incl. SSH keys
`passwd -e`	nothing; forces a new password

FILES

/etc/passwd fields

#	Field
1	login name
2	x = hash in shadow
3	UID
4	primary GID
5	comment (GECOS)
6	home directory
7	login shell

FILES (cont.)

/etc/shadow fields

#	Field
1	login name
2	hash; ! or * = locked
3	last change (days)
4-5	min / max age
6	warn days
7	inactive days
8	account expiry (days)

`/etc/group`
name:x:GID:member,member
Members are supplementary only; primary groups live in passwd.

`/etc/skel/`
Template copied into new homes

`/etc/login.defs`
UID ranges, aging and home defaults

SWITCH USER

`su - <user>`
Log in as user, full environment
Asks for that user's password.

`su -`
Become root with root's environment

`sudo -iu <user>`
Login shell as user via sudo
Asks for your own password, not theirs.

`sudo -u <user> <cmd>`
Run one command as user

su needs the target's password; sudo needs yours plus a sudoers rule. Prefer sudo, keep root's password locked.

RECIPES

Onboard a new admin

```
sudo useradd -m -s /bin/bash -c
'Ada Lovelace' ada
# group is wheel on RHEL/Fedora
sudo usermod -aG sudo ada
sudo passwd ada
sudo passwd -e ada
```

Service account for an app

```
sudo groupadd -r myapp
sudo useradd -r -g myapp -d /
opt/myapp -s /usr/sbin/nologin
myapp
sudo chown -R myapp:myapp /opt/
myapp
sudo -u myapp /opt/myapp/bin/
run --check
```

Offboard a user

```
# block all logins, then clean
up
sudo usermod -L -e 1 bob
sudo pkill -KILL -u bob
sudo tar czf /root/bob-
home.tgz /home/bob
sudo userdel -r bob
```