Table of Contents

Chapter 27. introduction to users

This little chapter will teach you how to identify your user account on a Unix computer using commands like **who am i**, **id**, and more.

In a second part you will learn how to become another user with the **su** command.

And you will learn how to run a program as another user with **sudo**.

27.1. whoami

The **whoami** command tells you your username.

```
[paul@centos7 ~]$ whoami
paul
[paul@centos7 ~]$
```

27.2. who

The **who** command will give you information about who is logged on the system.

```
[paul@centos7 ~]$ who
root     pts/0        2014-10-10 23:07 (10.104.33.101)
paul     pts/1        2014-10-10 23:30 (10.104.33.101)
laura    pts/2        2014-10-10 23:34 (10.104.33.96)
tania    pts/3        2014-10-10 23:39 (10.104.33.91)
[paul@centos7 ~]$
```

27.3. who am i

With **who am i** the **who** command will display only the line pointing to your current session.

```
[paul@centos7 ~]$ who am i
paul     pts/1        2014-10-10 23:30 (10.104.33.101)
[paul@centos7 ~]$
```

27.4. w

The **w** command shows you who is logged on and what they are doing.

```
[paul@centos7 ~]$ w
 23:34:07 up 31 min,  2 users,  load average: 0.00, 0.01, 0.02
USER     TTY        LOGIN@   IDLE   JCPU   PCPU WHAT
root     pts/0     23:07   15.00s  0.01s  0.01s top
paul     pts/1     23:30    7.00s  0.00s  0.00s w
[paul@centos7 ~]$
```

27.5. id

The **id** command will give you your user id, primary group id, and a list of the groups that you belong to.

```
paul@debian7:~$ id
uid=1000(paul) gid=1000(paul) groups=1000(paul)
```

On RHEL/CentOS you will also get **SELinux** context information with this command.

```
[root@centos7 ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r\
:unconfined_t:s0-s0:c0.c1023
```

27.6. su to another user

The **su** command allows a user to run a shell as another user.

```
laura@debian7:~$ su tania
Password:
tania@debian7:/home/laura$
```

27.7. su to root

Yes you can also **su** to become **root**, when you know the **root password**.

```
laura@debian7:~$ su root
Password:
root@debian7:/home/laura#
```

27.8. su as root

You need to know the password of the user you want to substitute to, unless your are logged in as **root**. The **root** user can become any existing user without knowing that user's password.

```
root@debian7:~# id
uid=0(root) gid=0(root) groups=0(root)
root@debian7:~# su - valentina
valentina@debian7:~$
```

27.9. su - $username

By default, the **su** command maintains the same shell environment. To become another user and also get the target user's environment, issue the **su -** command followed by the target username.

```
root@debian7:~# su laura
laura@debian7:/root$ exit
exit
root@debian7:~# su - laura
laura@debian7:~$ pwd
/home/laura
```

27.10. su -

When no username is provided to **su** or **su -**, the command will assume **root** is the target.

```
tania@debian7:~$ su -
Password:
root@debian7:~#
```

27.11. run a program as another user

The sudo program allows a user to start a program with the credentials of another user. Before this works, the system administrator has to set up the **/etc/sudoers** file. This can be useful to delegate administrative tasks to another user (without giving the root password).

The screenshot below shows the usage of **sudo**. User **paul** received the right to run **useradd** with the credentials of **root**. This allows **paul** to create new users on the system without becoming **root** and without knowing the **root password**.

First the command fails for **paul**.

```
paul@debian7:~$ /usr/sbin/useradd -m valentina
useradd: Permission denied.
useradd: cannot lock /etc/passwd; try again later.
```

But with **sudo** it works.

```
paul@debian7:~$ sudo /usr/sbin/useradd -m valentina
[sudo] password for paul:
paul@debian7:~$
```

27.12. visudo

Check the man page of **visudo** before playing with the **/etc/sudoers** file. Editing the **sudoers** is out of scope for this fundamentals book.

```
paul@rhel65:~$ apropos visudo
visudo               (8)  - edit the sudoers file
paul@rhel65:~$
```

27.13. sudo su -

On some Linux systems like Ubuntu and Xubuntu, the **root** user does not have a password set. This means that it is not possible to login as **root** (extra security). To perform tasks as **root**, the first user is given all **sudo rights** via the **/etc/sudoers**. In fact all users that are members of the admin group can use sudo to run all commands as root.

```
root@laika:~# grep admin /etc/sudoers
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL
```

The end result of this is that the user can type **sudo su -** and become root without having to enter the root password. The sudo command does require you to enter your own password. Thus the password prompt in the screenshot below is for sudo, not for su.

```
paul@laika:~$ sudo su -
Password:
root@laika:~#
```

27.14. sudo logging

Using **sudo** without authorization will result in a severe warning:

```
paul@rhel65:~$ sudo su -

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for paul:
paul is not in the sudoers file.  This incident will be reported.
paul@rhel65:~$
```

The root user can see this in the **/var/log/secure** on Red Hat and in **/var/log/auth.log** on Debian).

```
root@rhel65:~# tail /var/log/secure | grep sudo | tr -s ' '
Apr 13 16:03:42 rhel65 sudo: paul : user NOT in sudoers ; TTY=pts/0 ; PWD=\
/home/paul ; USER=root ; COMMAND=/bin/su -
root@rhel65:~#
```

27.15. practice: introduction to users

1. Run a command that displays only your currently logged on user name.

2. Display a list of all logged on users.

3. Display a list of all logged on users including the command they are running at this very moment.

4. Display your user name and your unique user identification (userid).

5. Use **su** to switch to another user account (unless you are root, you will need the password of the other account). And get back to the previous account.

6. Now use **su -** to switch to another user and notice the difference.

Note that **su -** gets you into the home directory of **Tania**.

7. Try to create a new user account (when using your normal user account). this should fail. (Details on adding user accounts are explained in the next chapter.)

8. Now try the same, but with **sudo** before your command.

27.16. solution: introduction to users

1. Run a command that displays only your currently logged on user name.

```
laura@debian7:~$ whoami
laura
laura@debian7:~$ echo $USER
laura
```

2. Display a list of all logged on users.

```
laura@debian7:~$ who
laura     pts/0        2014-10-13 07:22 (10.104.33.101)
laura@debian7:~$
```

3. Display a list of all logged on users including the command they are running at this very moment.

```
laura@debian7:~$ w
 07:47:02 up 16 min,  2 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
root     pts/0    10.104.33.101    07:30    6.00s  0.04s  0.00s w
root     pts/1    10.104.33.101    07:46    6.00s  0.01s  0.00s sleep 42
laura@debian7:~$
```

4. Display your user name and your unique user identification (userid).

```
laura@debian7:~$ id
uid=1005(laura) gid=1007(laura) groups=1007(laura)
laura@debian7:~$
```

5. Use **su** to switch to another user account (unless you are root, you will need the password of the other account). And get back to the previous account.

```
laura@debian7:~$ su tania
Password:
tania@debian7:/home/laura$ id
uid=1006(tania) gid=1008(tania) groups=1008(tania)
tania@debian7:/home/laura$ exit
laura@debian7:~$
```

6. Now use **su -** to switch to another user and notice the difference.

```
laura@debian7:~$ su - tania
Password:
tania@debian7:~$ pwd
/home/tania
tania@debian7:~$ logout
laura@debian7:~$
```

Note that **su -** gets you into the home directory of **Tania**.

7. Try to create a new user account (when using your normal user account). this should fail. (Details on adding user accounts are explained in the next chapter.)

```
laura@debian7:~$ useradd valentina
-su: useradd: command not found
laura@debian7:~$ /usr/sbin/useradd valentina
useradd: Permission denied.
useradd: cannot lock /etc/passwd; try again later.
```

It is possible that **useradd** is located in **/sbin/useradd** on your computer.

8. Now try the same, but with **sudo** before your command.

```
laura@debian7:~$ sudo /usr/sbin/useradd valentina
[sudo] password for laura:
laura is not in the sudoers file.  This incident will be reported.
laura@debian7:~$
```

Notice that **laura** has no permission to use the **sudo** on this system.

Chapter 28. user management

This chapter will teach you how to use **useradd**, **usermod** and **userdel** to create, modify and remove user accounts.

You will need **root** access on a Linux computer to complete this chapter.

28.1. user management

User management on Linux can be done in three complementary ways. You can use the **graphical** tools provided by your distribution. These tools have a look and feel that depends on the distribution. If you are a novice Linux user on your home system, then use the graphical tool that is provided by your distribution. This will make sure that you do not run into problems.

Another option is to use **command line tools** like useradd, usermod, gpasswd, passwd and others. Server administrators are likely to use these tools, since they are familiar and very similar across many different distributions. This chapter will focus on these command line tools.

A third and rather extremist way is to **edit the local configuration files** directly using vi (or vipw/vigr). Do not attempt this as a novice on production systems!

28.2. /etc/passwd

The local user database on Linux (and on most Unixes) is **/etc/passwd**.

```
[root@RHEL5 ~]# tail /etc/passwd
inge:x:518:524:art dealer:/home/inge:/bin/ksh
ann:x:519:525:flute player:/home/ann:/bin/bash
frederik:x:520:526:rubius poet:/home/frederik:/bin/bash
steven:x:521:527:roman emperor:/home/steven:/bin/bash
pascale:x:522:528:artist:/home/pascale:/bin/ksh
geert:x:524:530:kernel developer:/home/geert:/bin/bash
wim:x:525:531:master damuti:/home/wim:/bin/bash
sandra:x:526:532:radish stresser:/home/sandra:/bin/bash
annelies:x:527:533:sword fighter:/home/annelies:/bin/bash
laura:x:528:534:art dealer:/home/laura:/bin/ksh
```

As you can see, this file contains seven columns separated by a colon. The columns contain the username, an x, the user id, the primary group id, a description, the name of the home directory, and the login shell.

More information can be found by typing **man 5 passwd**.

```
[root@RHEL5 ~]# man 5 passwd
```

28.3. root

The **root** user also called the **superuser** is the most powerful account on your Linux system. This user can do almost anything, including the creation of other users. The root user always has userid 0 (regardless of the name of the account).

```
[root@RHEL5 ~]# head -1 /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

28.4. useradd

You can add users with the **useradd** command. The example below shows how to add a user named yanina (last parameter) and at the same time forcing the creation of the home directory (-m), setting the name of the home directory (-d), and setting a description (-c).

```
[root@RHEL5 ~]# useradd -m -d /home/yanina -c "yanina wickmayer" yanina
[root@RHEL5 ~]# tail -1 /etc/passwd
yanina:x:529:529:yanina wickmayer:/home/yanina:/bin/bash
```

The user named yanina received userid 529 and **primary group** id 529.

28.5. /etc/default/useradd

Both Red Hat Enterprise Linux and Debian/Ubuntu have a file called **/etc/default/useradd** that contains some default user options. Besides using cat to display this file, you can also use **useradd -D**.

```
[root@RHEL4 ~]# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

28.6. userdel

You can delete the user yanina with **userdel**. The -r option of userdel will also remove the home directory.

```
[root@RHEL5 ~]# userdel -r yanina
```

28.7. usermod

You can modify the properties of a user with the **usermod** command. This example uses **usermod** to change the description of the user harry.

```
[root@RHEL4 ~]# tail -1 /etc/passwd
harry:x:516:520:harry potter:/home/harry:/bin/bash
[root@RHEL4 ~]# usermod -c 'wizard' harry
[root@RHEL4 ~]# tail -1 /etc/passwd
harry:x:516:520:wizard:/home/harry:/bin/bash
```

28.8. creating home directories

The easiest way to create a home directory is to supply the **-m** option with **useradd** (it is likely set as a default option on Linux).

A less easy way is to create a home directory manually with **mkdir** which also requires setting the owner and the permissions on the directory with **chmod** and **chown** (both commands are discussed in detail in another chapter).

```
[root@RHEL5 ~]# mkdir /home/laura
[root@RHEL5 ~]# chown laura:laura /home/laura
[root@RHEL5 ~]# chmod 700 /home/laura
[root@RHEL5 ~]# ls -ld /home/laura/
drwx------ 2 laura laura 4096 Jun 24 15:17 /home/laura/
```

28.9. /etc/skel/

When using **useradd** the **-m** option, the **/etc/skel/** directory is copied to the newly created home directory. The **/etc/skel/** directory contains some (usually hidden) files that contain profile settings and default values for applications. In this way **/etc/skel/** serves as a default home directory and as a default user profile.

```
[root@RHEL5 ~]# ls -la /etc/skel/
total 48
drwxr-xr-x  2 root root  4096 Apr  1 00:11 .
drwxr-xr-x 97 root root 12288 Jun 24 15:36 ..
-rw-r--r--  1 root root    24 Jul 12  2006 .bash_logout
-rw-r--r--  1 root root   176 Jul 12  2006 .bash_profile
-rw-r--r--  1 root root   124 Jul 12  2006 .bashrc
```

28.10. deleting home directories

The -r option of **userdel** will make sure that the home directory is deleted together with the user account.

```
[root@RHEL5 ~]# ls -ld /home/wim/
drwx------ 2 wim wim 4096 Jun 24 15:19 /home/wim/
[root@RHEL5 ~]# userdel -r wim
[root@RHEL5 ~]# ls -ld /home/wim/
ls: /home/wim/: No such file or directory
```

28.11. login shell

The **/etc/passwd** file specifies the **login shell** for the user. In the screenshot below you can see that user annelies will log in with the **/bin/bash** shell, and user laura with the **/bin/ksh** shell.

```
[root@RHEL5 ~]# tail -2 /etc/passwd
annelies:x:527:533:sword fighter:/home/annelies:/bin/bash
laura:x:528:534:art dealer:/home/laura:/bin/ksh
```

You can use the usermod command to change the shell for a user.

```
[root@RHEL5 ~]# usermod -s /bin/bash laura
[root@RHEL5 ~]# tail -1 /etc/passwd
laura:x:528:534:art dealer:/home/laura:/bin/bash
```

28.12. chsh

Users can change their login shell with the **chsh** command. First, user harry obtains a list of available shells (he could also have done a **cat /etc/shells**) and then changes his login shell to the **Korn shell** (/bin/ksh). At the next login, harry will default into ksh instead of bash.

```
[laura@centos7 ~]$ chsh -l
/bin/sh
/bin/bash
/sbin/nologin
/usr/bin/sh
/usr/bin/bash
/usr/sbin/nologin
/bin/ksh
/bin/tcsh
/bin/csh
[laura@centos7 ~]$
```

Note that the **-l** option does not exist on Debian and that the above screenshot assumes that **ksh** and **csh** shells are installed.

The screenshot below shows how **laura** can change her default shell (active on next login).

```
[laura@centos7 ~]$ chsh -s /bin/ksh
Changing shell for laura.
Password:
Shell changed.
```

28.13. practice: user management

1. Create a user account named **serena**, including a home directory and a description (or comment) that reads **Serena Williams**. Do all this in one single command.

2. Create a user named **venus**, including home directory, bash shell, a description that reads **Venus Williams** all in one single command.

3. Verify that both users have correct entries in **/etc/passwd**, **/etc/shadow** and **/etc/group**.

4. Verify that their home directory was created.

5. Create a user named **einstime** with **/bin/date** as his default logon shell.

7. What happens when you log on with the **einstime** user ? Can you think of a useful real world example for changing a user's login shell to an application ?

8. Create a file named **welcome.txt** and make sure every new user will see this file in their home directory.

9. Verify this setup by creating (and deleting) a test user account.

10. Change the default login shell for the **serena** user to **/bin/bash**. Verify before and after you make this change.