Linux Security

LOCAL USER MANAGEMENT

Table of Contents

List of Tables

1. Introduction to users

This little chapter will teach you how to identify your user account on a Linux 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.

1.1. whoami

The whoami command tells you your username.

```
yves@linsrv1:~> whoami
yves
yves@linsrv1:~>
```

1.2. who

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

```
yves@linsrv1:~> who
root     :0           2016-12-13 18:43 (console)
root     console      2016-12-13 18:43 (:0)
root     pts/0        2016-12-13 20:10 (192.168.0.22)
npladm   pts/1        2016-12-13 19:46 (192.168.0.22)
sybnpl   pts/2        2016-12-13 19:58 (192.168.0.22)
yves     pts/3        2016-12-13 20:14 (192.168.0.22)
yves@linsrv1:~>
```

1.3. who am i

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

```
yves@linsrv1:~> who am i
yves     pts/3        2016-12-13 20:14 (192.168.0.22)
yves@linsrv1:~>
```

1.4. w

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

```
yves@linsrv1:~> w
 20:18:43 up  1:35,  6 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
root     :0       console          18:43   ?xdm?  59.13s  0.01s /usr/lib/gdm/gd
root     console  :0               18:43    1:35m  0.00s  0.01s /usr/lib/gdm/gd
root     pts/0    192.168.0.22     20:10    4:11   0.02s  0.02s -bash
npladm   pts/1    192.168.0.22     19:46   21:44   0.13s  0.13s -csh
sybnpl   pts/2    192.168.0.22     19:58   20:02   0.08s  0.08s -csh
yves     pts/3    192.168.0.22     20:14    0.00s  0.02s  0.00s w
yves@linsrv1:~>
```

1.5. id

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

```
yves@linsrv1:~> id
uid=1003(yves) gid=100(users) groups=100(users)
yves@linsrv1:~>
```

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
```

1.6. su to another user

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

```
yves@linsrv1:/root> su npladm
Password:
npladm@linsrv1:/root>
```

1.7. su to root

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

```
yves@linsrv1:~> su root
Password:
linsrv1:/home/yves #
```

1.8. su as root

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

```
linsrv1:~ # id
uid=0(root) gid=0(root) groups=0(root),1001(sapinst)
linsrv1:~ # su - yves
yves@linsrv1:~>
```

1.9. 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.

```
linsrv1:~ # su yves
yves@linsrv1:/root> exit
exit
linsrv1:~ # su - yves
yves@linsrv1:~> pwd
/home/yves
yves@linsrv1:~>
```

1.10. su -

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

```
yves@linsrv1:~> su -
Password:
Directory: /root
Tue Dec 13 20:52:55 CET 2016
linsrv1:~ #
```

1.11. 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 yves received the right to run useradd with the credentials of root. This allows yves to create new users on the system without becoming root and without knowing the root password.
First the command fails for yves.

```
yves@linsrv1:~> /usr/sbin/useradd -m tanguy
useradd: Permission denied.
useradd: cannot lock /etc/passwd; try again later.
```

But with sudo it works.

```
yves@linsrv1:~> sudo /usr/sbin/useradd -m tanguy
Creating mailbox file: File exists
yves@linsrv1:~>
```

1.12. visudo

Check the man page of visudo before playing with the /etc/sudoers file. Editing the sudoers is out of scope.

```
yves@linsrv1:~> apropos visudo
visudo (8)           - edit the sudoers file
yves@linsrv1:~>
```

1.13. 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:~#
```

1.14. sudo logging

Using sudo without authorization will result in a severe warning:

```
tanguy@linsrv1:~> 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.

root's password:
```

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

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

```
linsrv1:/home/yves # tail /var/log/messages | grep sudo | tr -s ' '
2016-12-13T22:07:29.237438+01:00 linsrv1 sudo: tanguy : TTY=pts/3 ; PWD=/home/tanguy ; USER=root ; COMMAND=/usr/bin/su -
2016-12-13T22:07:29.237735+01:00 linsrv1 sudo: pam_unix(sudo:session): session opened for user root by tanguy(uid=0)
2016-12-13T22:07:29.238872+01:00 linsrv1 sudo: pam_systemd(sudo:session): Cannot create session: Already running in a session
linsrv1:/home/yves #
```

1.15. Practice: introduction to users

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

```
yves@linsrv1:~> whoami
yves
yves@linsrv1:~> echo $USER
yves
yves@linsrv1:~>
yves@linsrv1:~> who am i
yves     pts/0        2016-12-13 22:17 (192.168.0.22)
yves@linsrv1:~>
```

2. Display a list of all logged on users.

```
yves@linsrv1:~> who
root     :0           2016-12-13 18:43 (console)
root     console      2016-12-13 18:43 (:0)
yves     pts/0        2016-12-13 22:17 (192.168.0.22)
sybnpl   pts/2        2016-12-13 19:58 (192.168.0.22)
tanguy   pts/3        2016-12-13 22:20 (192.168.0.22)
yves@linsrv1:~>
```

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

```
yves@linsrv1:~> w
 22:21:35 up  3:38,  5 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
root     :0       console          18:43   ?xdm?   1:39   0.02s /usr/lib/gdm/gd
root     console  :0               18:43    3:38m  0.00s  0.02s /usr/lib/gdm/gd
yves     pts/0    192.168.0.22     22:17    0.00s  0.03s  0.00s w
sybnpl   pts/2    192.168.0.22     19:58    1:40m  0.09s  0.09s -csh
tanguy   pts/3    192.168.0.22     22:20    1:33   0.02s  0.02s -bash
yves@linsrv1:~>
```

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

```
yves@linsrv1:~> id
uid=1003(yves) gid=100(users) groups=100(users)
yves@linsrv1:~>
```

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.

```
yves@linsrv1:~> su tanguy
Password:
tanguy@linsrv1:/home/yves> id
uid=1004(tanguy) gid=100(users) groups=100(users)
tanguy@linsrv1:/home/yves> exit
exit
yves@linsrv1:~>
```

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

```
yves@linsrv1:~> su - tanguy
Password:
tanguy@linsrv1:~> pwd
/home/tanguy
tanguy@linsrv1:~> logout
yves@linsrv1:~>
```

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

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.)

```
yves@linsrv1:~> /usr/sbin/useradd -m agathe
useradd: Permission denied.
useradd: cannot lock /etc/passwd; try again later.
```

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

```
yves@linsrv1:~> sudo /usr/sbin/useradd -m agathe
root's password:
Creating mailbox file: File exists
yves@linsrv1:~>
```

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

2. 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.

2.1. 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!

2.2. /etc/passwd

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

```
linsrv1:~ # tail /etc/passwd
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
uuidd:x:490:489:User for uuidd:/var/run/uuidd:/bin/bash
vnc:x:484:483:user for VNC:/var/lib/empty:/sbin/nologin
wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false
npladm:x:1001:478:SAP System Administrator:/home/npladm:/bin/bash
saprouter:x:1000:100::/var/lib/saprouter:/bin/false
sybnpl:x:1002:478:SAP Database Administrator:/sybase/NPL:/bin/csh
tanguy:x:1004:100:Tanguy Walter:/home/tanguy:/bin/bash
yves:x:1003:100:Yves Walter:/home/yves:/bin/bash
agathe:x:1005:100::/home/agathe:/bin/bash
```

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.

```
linsrv1:~ # man 5 passwd
```

2.3. 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).

```
linsrv1:~ # cat /etc/passwd | grep root
root:x:0:0:root:/root:/bin/bash
```

2.4. useradd

You can add users with the useradd command. The example below shows how to add a user named agathe (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).

```
linsrv1:~ # useradd -m -d /home/agathe -c "Agathe Thepower" agathe
linsrv1:~ # tail -1 /etc/passwd
agathe:x:1005:100:Agathe Thepower:/home/agathe:/bin/bash
```

The user named agathe received userid 1005 and primary group id 100.

2.5. /etc/default/useradd

Red Hat Enterprise Linux, Debian/Ubuntu, SLES 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.

```
linsrv1:~ # useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

2.6. userdel

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

```
linsrv1:~ # userdel -r agathe
no crontab for agathe
```

2.7. usermod

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

```
linsrv1:~ # cat /etc/passwd | grep tanguy
tanguy:x:1004:100:Tanguy Walter:/home/tanguy:/bin/bash
linsrv1:~ # usermod -c 'Tanguy Thepower' tanguy
linsrv1:~ # cat /etc/passwd | grep tanguy
tanguy:x:1004:100:Tanguy Thepower:/home/tanguy:/bin/bash
```

2.8. 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).

```
linsrv1:/ # mkdir /home/agathe
linsrv1:/ # useradd -c "Agathe Thepower" agathe
linsrv1:/ # cat /etc/passwd | grep "agathe"
agathe:x:1005:100:Agathe Thepower:/home/agathe:/bin/bash
linsrv1:/ # chown agathe:users /home/agathe
linsrv1:/ # chmod 700 /home/agathe
linsrv1:/ # ls -ld /home/agathe
drwx------ 1 agathe users 0 Dec 14 08:30 /home/agathe
```

2.9. /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.

```
linsrv1:/ # ls -la /etc/skel
total 56
drwxr-xr-x 1 root root   254 Nov 25 21:33 .
drwxr-xr-x 1 root root  5520 Dec 14 08:31 ..
-rw------- 1 root root     0 May 18  1996 .bash_history
-rw-r--r-- 1 root root  1177 Sep 30 13:44 .bashrc
drwx------ 1 root root     0 Sep 21  2014 .config
-rw-r--r-- 1 root root  1637 Sep 11  2014 .emacs
drwxr-xr-x 1 root root     0 Sep 21  2014 .fonts
-rw-r--r-- 1 root root 18517 Jul 30 01:31 .gnu-emacs
-rw-r--r-- 1 root root   305 Aug 21  2015 .i18n
-rw-r--r-- 1 root root   861 Sep 11  2014 .inputrc
drwx------ 1 root root     0 Sep 21  2014 .local
-rw-r--r-- 1 root root  6043 Aug 19 16:15 .muttrc
-rw-r--r-- 1 root root  1028 Sep 30 13:44 .profile
-rw-r--r-- 1 root root  1952 Aug 21  2015 .xim.template
-rwxr-xr-x 1 root root  1112 Jun  3  2016 .xinitrc.template
drwxr-xr-x 1 root root     0 Sep 21  2014 bin
drwxr-xr-x 1 root root    20 Nov 25 21:31 public_html
```

2.10. Deleting home directories

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

```
linsrv1:/ # ls -ld /home/agathe
drwx------ 1 agathe users 0 Dec 14 08:30 /home/agathe
linsrv1:/ # userdel -r agathe
no crontab for agathe
linsrv1:/ # ls -ld /home/agathe
ls: cannot access '/home/agathe': No such file or directory
```

2.11. Login shell

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

```
linsrv1:/ # tail -2 /etc/passwd
tanguy:x:1004:100:Tanguy Thepower:/home/tanguy:/bin/ksh
yves:x:1003:100:Yves Walter:/home/yves:/bin/bash
```

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

```
linsrv1:/ # usermod -s /bin/bash tanguy
linsrv1:/ # tail -2 /etc/passwd
tanguy:x:1004:100:Tanguy Thepower:/home/tanguy:/bin/bash
yves:x:1003:100:Yves Walter:/home/yves:/bin/bash
```

2.12. chsh

Users can change their login shell with the chsh command. First, user tanguy 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, tanguy will default into ksh instead of bash.
The screenshot below shows how tanguy can change his default shell (active on next login).

```
tanguy@linsrv1:~> echo $SHELL
/bin/bash
tanguy@linsrv1:~> cat /etc/passwd | grep 'tanguy'
tanguy:x:1004:100:Tanguy Thepower:/home/tanguy:/bin/bash
tanguy@linsrv1:~> chsh
Password:
Changing the login shell for tanguy
Enter the new value, or press ENTER for the default
        Login Shell [/bin/bash]: /bin/ksh
tanguy@linsrv1:~> echo $SHELL
/bin/bash
tanguy@linsrv1:~> exit
logout
linsrv1:/ # su - tanguy
$ echo $SHELL
/bin/ksh
$
```

2.13. 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.

```
linsrv1:/ # useradd -m -c 'Serena Williams' serena
```

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

```
linsrv1:/ # useradd -m -c 'Venus Williams' -s /bin/bash venus
```

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

```
linsrv1:/ # tail -2 /etc/passwd
serena:x:1005:100:Serena Williams:/home/serena:/bin/bash
venus:x:1006:100:Venus Williams:/home/venus:/bin/bash
```

```
linsrv1:/ # tail -2 /etc/shadow
serena:!:17149:0:99999:7:::
venus:!:17149:0:99999:7:::
```

```
linsrv1:/etc # cat /etc/group | grep 'users'
users:x:100:
```

6. Verify that their home directory was created.

```
linsrv1:/etc # ls -lrt /home | tail -2
drwxr-xr-x 1 serena users   254 Dec 14 10:50 serena
drwxr-xr-x 1 venus  users   254 Dec 14 10:53 venus
```

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

```
linsrv1:/etc # useradd -s /bin/date einstime
```

```
linsrv1:/etc # useradd -s $(which date) einstime
```

8. 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?

```
linsrv1:/etc # su - einstime
su: warning: cannot change directory to /home/einstime: No such file or directory
Wed Dec 14 11:10:56 CET 2016
```

7. Create a file named welcome.txt and make sure every new user will see this file in them home directory.

```
linsrv1:/etc # echo Hello > /etc/skel/welcome.txt
```

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

```
linsrv1:/etc # useradd -m -d /home/tanguy -c "Tanguy Thepower" tanguy
```

```
tanguy@linsrv1:~> ls -la | grep 'welcome'
-rw-r--r-- 1 tanguy users     6 Dec 14 11:17 welcome.txt
```

```
linsrv1:/etc # userdel -r tanguy
no crontab for tanguy
```

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

```
linsrv1:/etc # grep serena /etc/passwd
serena:x:1005:100:Serena Williams:/home/serena:/bin/bash
linsrv1:/etc # usermod -s /bin/sh serena
linsrv1:/etc # grep serena /etc/passwd
serena:x:1005:100:Serena Williams:/home/serena:/bin/sh
```

3. User passwords

This chapter will tell you more about passwords for local users. Three methods for setting passwords are explained; using the passwd command, using openssel passwd, and using the crypt function in a C program.
The chapter will also discuss password settings and disabling, suspending or locking accounts.

3.1. passwd

Passwords of users can be set with the passwd command. Users will have to provide their old password before twice entering the new one.

```
linsrv1:~ # passwd tanguy
New password:
BAD PASSWORD: it is based on a dictionary word
BAD PASSWORD: is too simple
Retype new password:
passwd: password updated successfully
linsrv1:~ # su - tanguy
tanguy@linsrv1:~> passwd tanguy
Changing password for tanguy.
(current) UNIX password:
New password:
BAD PASSWORD: it is based on your username
passwd: Authentication token manipulation error
passwd: password unchanged
```

As you can see, the passwd tool will do some basic verification to prevent users from using too simple passwords. The root user does not have to follow these rules (there will be
a warning though). The root user also does not have to provide the old password before entering the new password twice.

```
linsrv1:~ # passwd tanguy
New password:
BAD PASSWORD: it is based on a dictionary word
BAD PASSWORD: is too simple
Retype new password:
passwd: password updated successfully
```

3.2. shadow file

User passwords are encrypted and kept in /etc/shadow. The /etc/shadow file is read only and can only be read by root. We will see in the file permissions section how it is possible for users to change their password. For now, you will have to know that users can change their password with the /usr/bin/passwd command.

```
linsrv1:~ # tail -4 /etc/shadow
serena:!:17149:0:99999:7:::
venus:!:17149:0:99999:7:::
einstime:!:17149:0:99999:7:::
tanguy:$6$0Q/jBiyP$soZDV4DfhduJVc2ojDtsONyXnaA/z1EgltD6Gxjhqan4sgc0KCNgA35p8Lig8stPT/p.0H/dc9Hj44F1vmhdL1:17149:0:99999:7:::
```

The /etc/shadow file contains nine colon separated columns. The nine fields contain (from left to right) the user name, the encrypted password (note that only tanguy has an encrypted password), the day the password was last changed (day 1 is January 1, 1970), number of days the password must be left unchanged, password expiry day, warning number of days before password expiry, number of days after expiry before disabling the account, and the day the account was disabled (again, since 1970). The last field has no meaning yet.
All the passwords in the screenshot above are hashes of hunter2.

3.3. Encryption with passwd

Passwords are stored in an encrypted format. This encryption is done by the crypt function. The easiest (and recommended) way to add a user with a password to the system is to add the user with the useradd -m user command, and then set the user's password with passwd.

```
linsrv1:~ # useradd -m venus
linsrv1:~ # passwd venus
New password:
BAD PASSWORD: it is too short
BAD PASSWORD: is too simple
Retype new password:
passwd: password updated successfully
linsrv1:~ # tail -1 /etc/shadow
venus:$6$7ycHxB.1$7O2qsIS196VqdVDZ4HKtqU.tiLN5R5QWDb9tIt68pzJRqYYdsSgqZ53argl/W/uE5jd/r/KDvVrp.qXJh/Vbo.:17149:0:99999:7:::
```

3.4. Encryption with openssl

Another way to create users with a password is to use the -p option of useradd, but that option requires an encrypted password. You can generate this encrypted password with the openssl passwd command.
The openssl passwd command will generate several distinct hashes for the same password, for this it uses a salt.

```
yves@linsrv1:~> openssl passwd hunter2
YmZVZ1Z0zO2XQ
yves@linsrv1:~> openssl passwd hunter2
wevz7xbKe9/2A
yves@linsrv1:~> openssl passwd hunter2
QUlWncSegLDEA
```

This salt can be chosen and is visible as the first two characters of the hash.

```
yves@linsrv1:~> openssl passwd -salt 42 hunter2
42ZrbtP1Ze8G.
yves@linsrv1:~> openssl passwd -salt 42 hunter2
42ZrbtP1Ze8G.
yves@linsrv1:~> openssl passwd -salt 42 hunter2
42ZrbtP1Ze8G.
```

This example shows how to create a user with password.

```
yves@linsrv1:~> sudo useradd -m -p $(openssl passwd hunter2) agathe
```

Note that this command puts the password in your command history!

```
linsrv1:~ # tail -2 /etc/shadow
venus:$6$7ycHxB.l$7O2qsISl96VqdVDZ4HKtqU.tiLN5R5QWDb9tIt68pzJRqYYdsSgqZ53argl/W/
uE5jd/r/KDvVrp.qXJh/Vbo.:17149:0:99999:7:::
agathe:.78UoSq0lQON.:17149:0:99999:7:::
```

3.5. Encryption with crypt

A third option is to create your own C program using the crypt function, and compile this into a command.

```
paul@rhel65:~$ cat MyCrypt.c
#include <stdio.h>
#define __USE_XOPEN
#include <unistd.h>
int main(int argc, char** argv)
{
if(argc==3)
{
printf("%s\n", crypt(argv[1],argv[2]));
}
else
{
printf("Usage: MyCrypt $password $salt\n" );
}
return 0;
}
```

This little program can be compiled with gcc like this.

```
paul@rhel65:~$ gcc MyCrypt.c -o MyCrypt -lcrypt
```

To use it, we need to give two parameters to MyCrypt. The first is the unencrypted password, the second is the salt. The salt is used to perturb the encryption algorithm in one of 4096 different ways. This variation prevents two users with the same password from having the same entry in /etc/shadow.

```
paul@rhel65:~$ ./MyCrypt hunter2 42
42ZrbtP1Ze8G.
paul@rhel65:~$ ./MyCrypt hunter2 33
33d6taYSiEUXI
```

Did you notice that the first two characters of the password are the salt?
The standard output of the crypt function is using the DES algorithm which is old and can be cracked in minutes. A better method is to use md5 passwords which can be recognized by a salt starting with 1.

```
paul@rhel65:~$ ./MyCrypt hunter2 '$1$42'
$1$42$7l6Y3xT5282XmZrtDOF9f0
paul@rhel65:~$ ./MyCrypt hunter2 '$6$42'
$6$42$OqFFAVnI3gTSYG0yI9TZWX9cpyQzwIop7HwpG1LLEsNBiMr4w6OvLX1KDa./UpwXfrFk1i...
```

The md5 salt can be up to eight characters long. The salt is displayed in /etc/shadow between the second and third $, so never use the password as the salt!

```
paul@rhel65:~$ ./MyCrypt hunter2 '$1$hunter2'
$1$hunter2$YVxrxDmidq7Xf8Gdt6qM2.
```

3.6. /etc/login.defs

The /etc/login.defs file contains some default settings for user passwords like password aging and length settings. (You will also find the numerical limits of user ids and group ids and whether or not a home directory should be created by default).

```
linsrv1:~ # grep ^PASS /etc/login.defs
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
```

3.7. chage

The chage command can be used to set an expiration date for a user account (-E), set a minimum (-m) and maximum (-M) password age, a password expiration date, and set the number of warning days before the password expiration date. Much of this functionality is also available from the passwd command. The -l option of chage will list these settings for a user.

```
linsrv1:~ # chage -l yves
Last password change                                    : Dec 13, 2016
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
```

3.8. Disabling a password

Passwords in /etc/shadow cannot begin with an exclamation mark. When the second field in /etc/passwd starts with an exclamation mark, then the password can not be used. Using this feature is often called locking, disabling, or suspending a user account. Besides vi (or vipw) you can also accomplish this with usermod.

The first command in the next screenshot will show the hashed password of laura in /etc/shadow. The next command disables the password of Agathe, making it impossible for Agathe to authenticate using this password.

```
linsrv1:~ # grep agathe /etc/shadow | cut -c1-70
agathe:$6$IbZM8c.8$ZglpSsfMzi3M4IDzzvhLKs7QQ01MDAXKwEdGr0SbvljumQrnCrm
linsrv1:~ # usermod -L agathe
```

As you can see below, the password hash is simply preceded with an exclamation mark.

```
linsrv1:~ # grep agathe /etc/shadow | cut -c1-70
agathe:!$6$IbZM8c.8$ZglpSsfMzi3M4IDzzvhLKs7QQ01MDAXKwEdGr0SbvljumQrnCr
```

The root user (and users with sudo rights on su) still will be able to su into the laura account (because the password is not needed here). Also, note that laura will still be able to login if she has set up passwordless ssh!

```
linsrv1:~ # su - agathe
agathe@linsrv1:~>
```

You can unlock the account again with usermod -U.

```
linsrv1:~ # usermod -U agathe
linsrv1:~ # grep agathe /etc/shadow | cut -c1-70
agathe:$6$IbZM8c.8$ZglpSsfMzi3M4IDzzvhLKs7QQ01MDAXKwEdGr0SbvljumQrnCrm
```

Watch out for tiny differences in the command line options of passwd, usermod, and useradd on different Linux distributions. Verify the local files when using features like "disabling, suspending, or locking" on user accounts and their passwords.

3.9. Editing local files

If you still want to manually edit the /etc/passwd or /etc/shadow, after knowing these commands for password management, then use vipw instead of vi(m) directly. The vipw tool will do proper locking of the file.

```
linsrv1:~ # vipw --help
Usage: vipw [options]

Options:
  -g, --group                   edit group database
  -h, --help                    display this help message and exit
  -p, --passwd                  edit passwd database
  -q, --quiet                   quiet mode
  -R, --root CHROOT_DIR         directory to chroot into
  -s, --shadow                  edit shadow or gshadow database
```

```
sshd:x:498:498:SSH daemon:/var/lib/sshd:/bin/false
statd:x:488:65534:NFS statd daemon:/var/lib/nfs:/sbin/nologin
systemd-bus-proxy:x:492:492:systemd Bus Proxy:/:/sbin/nologin
systemd-timesync:x:491:491:systemd Time Synchronization:/:/sbin/nologin
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
uuidd:x:490:489:User for uuidd:/var/run/uuidd:/bin/bash
vnc:x:484:483:user for VNC:/var/lib/empty:/sbin/nologin
wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false
einstime:x:1007:100::/home/einstime:/usr/bin/date
npladm:x:1001:478:SAP System Administrator:/home/npladm:/bin/bash
saprouter:x:1000:100::/var/lib/saprouter:/bin/false
sybnpl:x:1002:478:SAP Database Administrator:/sybase/NPL:/bin/csh
tanguy:x:1008:100::/home/tanguy:/bin/bash
yves:x:1003:100:Yves Walter:/home/yves:/bin/bash
venus:x:1009:100::/home/venus:/bin/bash
agathe:x:1010:100::/home/agathe:/bin/bash
```

3.10. Practice: user passwords

1. Set the password for serena to hunter2.
2. Also set a password for venus and then lock the venus user account with usermod. Verify the locking in /etc/shadow before and after you lock it.
3. Use passwd -d to disable the serena password. Verify the serena line in /etc/shadow before and after disabling.
4. What is the difference between locking a user account and disabling a user account's password like we just did with usermod -L and passwd -d?
5. Try changing the password of serena to serena as serena.
6. Make sure serena has to change her password in 10 days.
7. Make sure every new user needs to change their password every 10 days.
8. Take a backup as root of /etc/shadow. Use vi to copy an encrypted hunter2 hash from venus to serena. Can serena now log on with hunter2 as a password ?
9. Why use vipw instead of vi ? What could be the problem when using vi or vim ?
10. Use chsh to list all shells (only works on RHEL/CentOS/Fedora), and compare to cat /etc/shells.
11. Which useradd option allows you to name a home directory ?
12. How can you see whether the password of user serena is locked or unlocked ? Give a solution with grep and a solution with passwd.

3.11. solution: user passwords

1. Set the password for serena to hunter2.

```
root@debian7:~# passwd serena
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

2. Also set a password for venus and then lock the venus user account with usermod. Verify the locking in /etc/shadow before and after you lock it.

```
root@debian7:~# passwd venus
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@debian7:~# grep venus /etc/shadow | cut -c1-70
venus:$6$gswzXICW$uSnKFV1kFKZmTPaMVS4AvNA/KO27OxN0v5LHdV9ed0gTyXrjUeM/
root@debian7:~# usermod -L venus
root@debian7:~# grep venus /etc/shadow | cut -c1-70
venus:!$6$gswzXICW$uSnKFV1kFKZmTPaMVS4AvNA/KO27OxN0v5LHdV9ed0gTyXrjUeM
```

Note that usermod -L precedes the password hash with an exclamation mark (!).

3. Use passwd -d to disable the serena password. Verify the serena line in /etc/shadow before and after disabling.

```
root@debian7:~# grep serena /etc/shadow | cut -c1-70
serena:$6$Es/omrPE$F2Ypu8kpLrfKdW0v/UIwA5jrYyBD2nwZ/dt.i/IypRgiPZSdB/B
root@debian7:~# passwd -d serena
passwd: password expiry information changed.
root@debian7:~# grep serena /etc/shadow
serena::16358:0:99999:7:::
root@debian7:~#
```

4. What is the difference between locking a user account and disabling a user account's password like we just did with usermod -L and passwd -d?

Locking will prevent the user from logging on to the system with his password by putting a ! in front of the password in /etc/shadow.
Disabling with passwd will erase the password from /etc/shadow.

5. Try changing the password of serena to serena as serena.

```
log on as serena, then execute: passwd serena... it should fail!
```

6. Make sure serena has to change her password in 10 days.

```
chage -M 10 serena
```

7. Make sure every new user needs to change their password every 10 days.

```
vi /etc/login.defs (and change PASS_MAX_DAYS to 10)
```

8. Take a backup as root of /etc/shadow. Use vi to copy an encrypted hunter2 hash from venus to serena. Can serena now log on with hunter2 as a password ?

```
Yes.
```

9. Why use vipw instead of vi ? What could be the problem when using vi or vim ?

```
vipw will give a warning when someone else is already using that file (with vipw).
```

10. Use chsh to list all shells (only works on RHEL/CentOS/Fedora), and compare to cat /etc/shells.

```
chsh -l
cat /etc/shells
```

11. Which useradd option allows you to name a home directory ?

```
-d
```

12. How can you see whether the password of user serena is locked or unlocked ? Give a solution with grep and a solution with passwd.

```
grep serena /etc/shadow
passwd -S serena
```

4. User profiles

Logged on users have several preset (and customized) aliases, variables, and functions, but where do they come from? The shell uses several startup files that are executed (or rather sourced) whenever the shell is invoked. What follows is an overview of startup scripts.

4.1. System profile

Both the bash and the ksh shell will verify the existence of /etc/profile and source it if it exists.
When reading this script, you will notice (both on Debian and on Red Hat Enterprise Linux) that it builds the PATH environment variable (among others). The script might also change the PS1 variable, set the HOSTNAME and execute even more scripts like /etc/inputrc

This screenshot uses grep to show PATH manipulation in /etc/profile on SLES.

```
linsrv1:~ # grep PATH /etc/profile
# ulimit package instead to set up ulimits and your PATH.
    PATH=/usr/local/bin:/usr/bin:/bin
            test -d $dir && PATH=$dir:$PATH
        test -d /opt/kde3/sbin  && PATH=/opt/kde3/sbin:$PATH
        PATH=/sbin:/usr/sbin:/usr/local/sbin:$PATH
        test -d $dir && PATH=$PATH:$dir
    export PATH
    tmp="$MANPATH"
    unset MANPATH
        MANPATH="${tmp}:`test -x /usr/bin/manpath && /usr/bin/manpath -q`"
        MANPATH="`test -x /usr/bin/manpath && /usr/bin/manpath -q`"
    export MANPATH
        export XNLSPATH=/usr/share/X11/nls
        export XNLSPATH=/usr/X11R6/lib/X11/nls
    PATH=/usr/lib/restricted/bin
    export PATH
linsrv1:~ # echo $PATH
/sbin:/usr/sbin:/usr/local/sbin:/root/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/
X11:/usr/games
```

The root user can use this script to set aliases, functions, and variables for every user on the system.

4.2. ~/.bash_profile

When this file exists in the home directory, then bash will source it. On Debian Linux 5/6/7 this file does not exist by default.
RHEL7/CentOS7 uses a small ~/.bash_profile where it checks for the existence of ~/.bashrc and then sources it. It also adds $HOME/bin to the $PATH variable.

```
[root@rhel7 ~]# cat /home/paul/.bash_profile
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
. ~/.bashrc
fi
# User specific environment and startup programs
PATH=$PATH:$HOME/.local/bin:$HOME/bin
export PATH
[root@rhel7 ~]#
```

```
linsrv1:~ # cat /home/yves/.bash_profile
cat: /home/yves/.bash_profile: No such file or directory
```

4.3. ~/.bash_login

When .bash_profile does not exist, then bash will check for ~/.bash_login and source it. Neither Debian nor Red Hat have this file by default.

```
linsrv1:~ # cat /home/yves/.bash_login
cat: /home/yves/.bash_login: No such file or directory
```

4.4. ~/.profile

When neither ~/.bash_profile and ~/.bash_login exist, then bash will verify the existence of ~/.profile and execute it. This file does not exist by default on Red Hat. On Debian this script can execute ~/.bashrc and will add $HOME/bin to the $PATH variable.

```
root@debian7:~# tail -11 /home/paul/.profile
if [ -n "$BASH_VERSION" ]; then
# include .bashrc if it exists
if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
fi
fi
# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
PATH="$HOME/bin:$PATH"
fi
```

RHEL/CentOS does not have this file by default.

```
linsrv1:/home/yves # tail -11 /home/yves/.profile
#export LANG=es_ES.UTF-8        # uncomment this line for Spanish output


# Some people don't like fortune. If you uncomment the following lines,
# you will have a fortune each time you log in ;-)

#if [ -x /usr/bin/fortune ] ; then
#    echo
#    /usr/bin/fortune
#    echo
#fi
```

4.5. ~/.bashrc

The ~/.bashrc script is often sourced by other scripts. Let us look at what it does by default.
Red Hat uses a very simple ~/.bashrc, checking for /etc/bashrc and sourcing it. It also leaves room for custom aliases and functions.

```
[root@rhel7 ~]# cat /home/paul/.bashrc
# .bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
. /etc/bashrc
fi
# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=
# User specific aliases and functions
```

```
linsrv1:/home/yves # cat /home/yves/.bashrc
# Sample .bashrc for SuSE Linux
# Copyright (c) SuSE GmbH Nuernberg

# There are 3 different types of shells in bash: the login shell, normal shell
# and interactive shell. Login shells read ~/.profile and interactive shells
# read ~/.bashrc; in our setup, /etc/profile sources ~/.bashrc - thus all
# settings made here will also take effect in a login shell.
#
```

On Debian this script is quite a bit longer and configures $PS1, some history variables and a number af active and inactive aliases.

```
root@debian7:~# wc -l /home/paul/.bashrc
110 /home/paul/.bashrc
```

```
linsrv1:/home/yves # wc -l /home/yves/.bashrc
28 /home/yves/.bashrc
```

4.6. ~/.bash_logout

When exiting bash, it can execute ~/.bash_logout. Debian use this opportunity to clear the console screen.

```
serena@deb503:~$ cat .bash_logout
# ~/.bash_logout: executed by bash(1) when login shell exits.
# when leaving the console clear the screen to increase privacy
```

```
if [ "$SHLVL" = 1 ]; then
[ -x /usr/bin/clear_console ] && /usr/bin/clear_console -q
fi
```

Red Hat Enterprise Linux 5 will simple call the /usr/bin/clear command in this script.

```
[serena@rhel53 ~]$ cat .bash_logout
# ~/.bash_logout
/usr/bin/clear
```

Red Hat Enterprise Linux 6 and 7 create this file, but leave it empty (except for a comment).

```
paul@rhel65:~$ cat .bash_logout
# ~/.bash_logout
```

```
yves@linsrv1:~> cat .bash_logout
cat: .bash_logout: No such file or directory
```

4.7. Debian overview

Below is a table overview of when Debian is running any of these bash startup scripts.

Table 4.1. Debian User Environment

script	su	su -	ssh	gdm
~./bashrc	no	yes	yes	yes
~/.profile	no	yes	yes	yes
/etc/profile	no	yes	yes	yes
/etc/bash.bashrc	yes	no	no	yes

4.8. RHEL5 overview

Below is a table overview of when Red Hat Enterprise Linux 5 is running any of these bash startup scripts.

Table 4.2. Red Hat User Environment

script	su	su -	ssh	gdm
~./bashrc	yes	yes	yes	yes
~/.bash_profile	no	yes	yes	yes
/etc/profile	no	yes	yes	yes
/etc/bashrc	yes	no	no	yes

4.9. Practice: user profiles

1. Make a list of all the profile files on your system.
2. Read the contents of each of these, often they source extra scripts.
3. Put a unique variable, alias and function in each of those files.
4. Try several different ways to obtain a shell (su, su -, ssh, tmux, gnome-terminal, Ctrlalt-F1, ...) and verify which of your custom variables, aliases and function are present in your environment.
5. Do you also know the order in which they are executed?
6. When an application depends on a setting in $HOME/.profile, does it matter whether $HOME/.bash_profile exists or not ?

4.10. solution: user profiles

1. Make a list of all the profile files on your system.

```
ls -a ~ ; ls -l /etc/pro* /etc/bash*
```

2. Read the contents of each of these, often they source extra scripts.
3. Put a unique variable, alias and function in each of those files.
4. Try several different ways to obtain a shell (su, su -, ssh, tmux, gnome-terminal, Ctrlalt-F1, ...) and verify which of your custom variables, aliases and function are present in your environment.
5. Do you also know the order in which they are executed?

```
same name aliases, functions and variables will overwrite each other
```

6. When an application depends on a setting in $HOME/.profile, does it matter whether $HOME/.bash_profile exists or not ?

Yes it does matter. (man bash /INVOCATION)

5. Groups

Users can be listed in groups. Groups allow you to set permissions on the group level instead of having to set permissions for every individual user. Every Unix or Linux distribution will have a graphical tool to manage groups. Novice users are advised to use this graphical tool. More experienced users can use command line tools to manage users, but be careful: Some distributions do not allow the mixed use of GUI and CLI tools to manage groups (YaST in Novell Suse). Senior administrators can edit the relevant files directly with vi or vigr.

5.1. groupadd

Groups can be created with the groupadd command. The example below shows the creation of five (empty) groups.

```
yves@linsrv1:~> sudo groupadd tennis
yves@linsrv1:~> sudo groupadd football
yves@linsrv1:~> sudo groupadd snooker
yves@linsrv1:~> sudo groupadd formula1
yves@linsrv1:~> sudo groupadd salsa
```

5.2. group file

Users can be a member of several groups. Group membership is defined by the /etc/group file.

```
yves@linsrv1:~> tail -5 /etc/group
tennis:x:1002:
football:x:1003:
snooker:x:1004:
formula1:x:1005:
salsa:x:1006:
```

The first field is the group's name. The second field is the group's (encrypted) password (can be empty). The third field is the group identification or GID. The fourth field is the list of members, these groups have no members.

5.3. groups

A user can type the groups command to see a list of groups where the user belongs to.

```
yves@linsrv1:~> groups
users
```

5.4. usermod

Group membership can be modified with the useradd or usermod command.

```
yves@linsrv1:~> sudo usermod -a -G tennis agathe
yves@linsrv1:~> sudo usermod -a -G tennis tanguy
yves@linsrv1:~> sudo usermod -a -G snooker tanguy
yves@linsrv1:~> tail -5 /etc/group
tennis:x:1002:agathe,tanguy
football:x:1003:
snooker:x:1004:tanguy
formula1:x:1005:
salsa:x:1006:
```

Be careful when using usermod to add users to groups. By default, the usermod command will remove the user from every group of which he is a member if the group is not listed in the command! Using the -a (append) switch prevents this behaviour.

5.5. groupmod

You can change the group name with the groupmod command.

```
yves@linsrv1:~> sudo groupmod -n darts snooker
yves@linsrv1:~> tail -5 /etc/group
tennis:x:1002:agathe,tanguy
football:x:1003:
formula1:x:1005:
salsa:x:1006:
darts:x:1004:tanguy
```

5.6. groupdel

You can permanently remove a group with the groupdel command.

```
yves@linsrv1:~> sudo groupdel tennis
```

5.7. gpasswd

You can delegate control of group membership to another user with the gpasswd command.
In the example below we delegate permissions to add and remove group members to serena for the sports group. Then we su to serena and add harry to the sports group.

```
[root@RHEL4b ~]# gpasswd -A serena sports
[root@RHEL4b ~]# su - serena
[serena@RHEL4b ~]$ id harry
uid=516(harry) gid=520(harry) groups=520(harry)
[serena@RHEL4b ~]$ gpasswd -a harry sports
Adding user harry to group sports
[serena@RHEL4b ~]$ id harry
uid=516(harry) gid=520(harry) groups=520(harry),522(sports)
[serena@RHEL4b ~]$ tail -1 /etc/group
sports:x:522:serena,venus,harry
[serena@RHEL4b ~]$
```

Group administrators do not have to be a member of the group. They can remove themselves from a group, but this does not influence their ability to add or remove members.

```
[serena@RHEL4b ~]$ gpasswd -d serena sports
Removing user serena from group sports
[serena@RHEL4b ~]$ exit
```

Information about group administrators is kept in the /etc/gshadow file.

```
[root@RHEL4b ~]# tail -1 /etc/gshadow
sports:!:serena:venus,harry
[root@RHEL4b ~]#
```

To remove all group administrators from a group, use the gpasswd command to set an empty administrators list.

```
[root@RHEL4b ~]# gpasswd -A "" sports
```

5.8. newgrp

You can start a child shell with a new temporary primary group using the newgrp command.

```
linsrv1:~ # mkdir prigroup
linsrv1:~ # cd prigroup/
linsrv1:~/prigroup # touch standard.txt
linsrv1:~/prigroup # ls -l
total 0
-rw-r--r-- 1 root root 0 Dec 14 22:30 standard.txt
linsrv1:~/prigroup # echo $SHLVL
1
linsrv1:~/prigroup # newgrp tennis
newgrp: group 'tennis' does not exist
linsrv1:~/prigroup # echo $SHLVL
1
linsrv1:~/prigroup # newgrp sports
linsrv1:~/prigroup # echo $SHLVL
2
linsrv1:~/prigroup # touch newgrp.txt
linsrv1:~/prigroup # ls -l
total 0
-rw-r--r-- 1 root sports 0 Dec 14 22:33 newgrp.txt
-rw-r--r-- 1 root root   0 Dec 14 22:30 standard.txt
linsrv1:~/prigroup # exit
exit
```

5.9. vigr

Similar to vipw, the vigr command can be used to manually edit the /etc/group file, since it will do proper locking of the file. Only experienced senior administrators should use vi or vigr to manage groups.

```
linsrv1:~/prigroup # vigr -g
root:x:0:root
bin:x:1:daemon
daemon:x:2:
sys:x:3:
tty:x:5:
disk:x:6:
lp:x:7:
www:x:8:
kmem:x:9:
wheel:x:10:
mail:x:12:postfix
news:x:13:
```

```
linsrv1:~/prigroup # tail -5 /etc/group
football:x:1003:
formula1:x:1005:
salsa:x:1006:
darts:x:1004:tanguy
sports:x:1007:
```

```
yves@linsrv1:~> sudo groupdel football
yves@linsrv1:~> sudo groupdel formula1
yves@linsrv1:~> sudo groupdel salsa
yves@linsrv1:~> sudo groupdel darts
yves@linsrv1:~> sudo groupdel sports
```

5.10. Practice: groups

1. Create the groups tennis, football and sports.

```
yves@linsrv1:~> sudo groupadd tennis; sudo groupadd football; sudo groupadd sports
```

2. In one command, make venus a member of tennis and sports.

```
yves@linsrv1:~> sudo usermod -a -G tennis,sports venus
```

3. Rename the football group to foot.

```
yves@linsrv1:~> sudo groupmod -n foot football
```

4. Use vi to add serena to the tennis group.

```
yves@linsrv1:~> tail -3 /etc/group
tennis:x:1002:venus,serena
sports:x:1004:venus
foot:x:1003:
```

5. Use the id command to verify that serena is a member of tennis.

```
yves@linsrv1:~> sudo su - serena
serena@linsrv1:~> id
uid=1011(serena) gid=100(users) groups=100(users),1002(tennis)
```

6. Make someone responsible for managing group membership of foot and sports. Test that it works.

```
gpasswd -A (to make manager)
gpasswd -a (to add member)
```