LPI 110.1 - Perform security administration tasks

Curs 2021 - 2022

ASIX M01-ISO 110 Security

Perform security administration tasks

Description

Key concepts:

- ❏ Audit a system to find files with the suid/sgid bit set.
- ❏ Set or change user passwords and password aging information.
- ❏ Being able to use nmap and netstat to discover open ports on a system.
- ❏ Set up limits on user logins, processes and memory usage.
- ❏ Determine which users have logged in to the system or are currently logged in.
- ❏ Basic sudo configuration and usage.

Commands and files:

- ❏ find
- ❏ passwd
- ❏ fuser
- ❏ lsof
- ❏ nmap
- ❏ chage
- ❏ netstat
- ❏ sudo
- ❏ /etc/sudoers
- ❏ su
- ❏ usermod
- ❏ ulimit
- ❏ who, w, last

sudo Command

The superuser do sudo utility allows a user to execute a single program or command as the root or another user without knowing their password or remaining logged in as that user, thus improving security. This utility is commonly-used to execute programs that require root privileges.

When sudo asks for a password, it needs the current user's password, and not the root account password.

- /etc/sudoers
- visudo

- sudo command

There are several key advantages of using sudo instead of logging in as root:

- The root password is not exposed to users.
- The amount of time that users spend with root privileges is strictly restricted to the command execution only.
- Users that are executing the commands as root are logged.

The /etc/sudoers file should be edited using the visudo command as root or by using sudo and not a standard text editor. visudo is a special editor that validates the syntax of the file before saving the changes.

Type of /etc/sudoers entries:

Aliases

User_Alias

Specifies groups of users. Usernames, system groups (prefixed by a percent % sign), and netgroups (prefixed by a plus + sign) can be specified. You can exclude particular users with !.

Host_Alias

Specifies a list of hostname, IP addresses, networks, and netgroups (prefixed with a plus + sign). Netmasks can be also specified. You can exclude particular hosts with !.

Runas_Alias

Similar to user aliases but accepts UIDs instead of username. Better for matching multiple user names and groups having different names but the same UID. Uid users are in the form #UID (using the # character)

Cmnd_Alias

Specifies a list of commands and directories. Specifying a directory will include all files within that directory but no subdirectories. You can exclude particularcommands with !.

Aliases examples

```
User_Alias     OPERATORS = user1, user2, user3
Host_Alias     DBNET = 172.16.0.0/255.255.224.0
Runas_Alias    OP = root, operator, #1001, #2005
Cmnd_Alias     EDITORS = /usr/bin/vim, /usr/bin/nano
```

Specification

Specifications define which users can execute which programs

host

machines where the rules are efective

as user

run the command as this user

command

command or commands to run.

/etc/sudoers example

```
OPERATORS ALL=ALL
testuser1 DBNET=(ALL) ALL
testuser2 ALL= EDITORS
```

Administrator groups:

In the Red Hat family of distributions the wheel group is the counterpart to the special administrative sudo group of Debian systems. Usually the sudo configuration (sudoers file) includes a rule for the group sudo/wheel. So the user's members od this groups can use sudo without the need to modify the file.

Sudoers configuration examples:

User_Alias	OPERATORS = user1, user2, user3
Host_Alias	DBNET = 172.16.0.0/255.255.224.0
Runas_Alias	OP = root, operator, #1001, #2005
Cmnd_Alias	EDITORS = /usr/bin/vim, /usr/bin/nano

OPERATORS ALL=ALL

Users who are part of the OPERATORS groups can execute any command. The first ALL indicates which machines the rule applies to, and the second ALL indicates which commands can be executed.

testuser1 DBNET=(ALL) ALL

testuser1 can run any command as any user on any host that is in the DBNET network.

testuser2 ALL= EDITORS

testuser2 can run the vim and nano editors as either the root user or any other user on the system.

The sudo command executes the commands allowed by the sudoers file. Options:

-b	Execute the command in background
-u user_name	Execute the command as the specified user instead of as the root user
-n	Do not prompt the user for their password

```
$ sudo fdisk -l
$ sudo -u pere ls ~
```

More sudoers configuration examples:

```
# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
```

```
%sudo   ALL=(ALL:ALL) ALL


# user carol be able to check apache2 status from any host as any user or group,
carol   ALL=(ALL:ALL) /usr/bin/systemctl status apache2


# save carol the inconvenience of having to provide her password to run the systemctl
# status apache2 command
carol   ALL=(ALL:ALL) NOPASSWD: /usr/bin/systemctl status apache2


# restrict your hosts to 192.168.1.7 and enable carol to run systemctl status apache2
# as user mimi
carol   192.168.1.7=(mimi) /usr/bin/systemctl status apache2
```

```
# The SERVERS host alias includes an IP address and two hostnames
Host_Alias SERVERS = 192.168.1.7, server1, server2

# he ADMINS user alias — for example — includes user carol, the members of the sudo group
# and those members of the PRIVILEGE_USERS user alias that do not belong
# in the REGULAR_USERS user alias
User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# The SERVICES command alias includes a single command with all its subcommands
# — as specified by the asterisk (*)
Cmnd_Alias SERVICES = /usr/bin/systemctl *
```

Example-1

```
# visudo
pere ALL=(ALL:ALL) /sbin/useradd
```

```
# visudo
pere ALL=(ALL:ALL) /sbin/useradd
marta ALL /sbin/usermod
esc:wq

etc/sudoers:122:24: syntax error
marta ALL /sbin/usermod
                      ^
What now? e [edit]

What now? q [exit]
Options are:
  (e)dit sudoers file again
  e(x)it without saving changes to sudoers file
  (Q)uit and save changes to sudoers file (DANGER!)
```

```
[pere@sudo ~]$ useradd marta
useradd: Permission denied.
useradd: cannot lock /etc/passwd; try again later.

[pere@sudo ~]$ sudo useradd marta
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 pere:

[pere@sudo ~]$ tail -1 /etc/passwd
marta:x:1001:1001::/home/marta:/bin/bash
```

```
[pere@sudo ~]$ sudo userdel -r marta
Sorry, user pere is not allowed to execute '/usr/sbin/userdel -r marta' as root on
sudo.edt.org.
```

Example-2

```
[pere@sudo ~]$ sudo useradd  -s /bin/sh marta
[sudo] password for pere:
Sorry, user pere is not allowed to execute '/usr/sbin/useradd -s /bin/sh marta' as root
on sudo.edt.org.
```

```
pere ALL=(ALL:ALL) /sbin/useradd -s /bin/bash *
```

```
[pere@sudo ~]$ sudo useradd  -s /bin/bash marta
[sudo] password for pere:
useradd: user 'marta' already exists
```

Example-3

```
User_Alias ADMINS = pere, marta, @operators
Host_Alias SERVERS = 172.17.0.0/16, server1, server2
Cmnd_Alias CMDUSERS = /sbin/user*
ADMINS  SERVERS=CMDUSERS
```

```
[pere@sudo ~]$ sudo userdel -r marta
[sudo] password for pere:

[pere@sudo ~]$ sudo useradd marta

[pere@sudo ~]$ sudo usermod -g 100 marta

[pere@sudo ~]$ id marta
uid=1001(marta) gid=100(users) groups=100(users)
```

Example-4

```
User_Alias ADMINS = pere, marta, %operators
Host_Alias SERVERS = 172.17.0.0/16, server1, server2
Cmnd_Alias CMDUSERS = /sbin/user*
ADMINS  SERVERS=(ALL:ALL) NOPASSWD: CMDUSERS
```

```
[pere@sudo ~]$ sudo  useradd anna

[pere@sudo ~]$ tail -1 /etc/passwd
anna:x:1002:1002::/home/anna:/bin/bash
```

su command

The superuser su command is used to execute a shell with a different user identity. This command is typically used by a regular user to execute a command which otherwise needs root privileges or when the root user wants to execute a command as a regular user. For a regular user to use this command, the password for the other account must be entered.

- su -
- su -l
- su (no usar!)
- su -l -c command

Su options:

- or -l
  Start the new user's login shell and execute the initialization (.rc) files providing an environment (i.e., variables, aliases, home directory, etc.) similar to what the user would expect had the user logged in directly.

-c command
  Pass a single command to the shell. As a result, after the su command has completed, the user will revert back to their original shell.

-m
  Do not reset the values of environment variables.

If someone with knowledge of the root password needs to execute several commands with root privileges, they would use the su - command to switch identities to the root user and acquire the root account environment settings (the - option tells the shell to read the user's initialization files) by executing the following command and providing the root user's password.

The main differences between su and sudo are that su switches the current user (possibly to root) and remains that user until the account is exited, whereas sudo runs a single command with root privileges when provided the current user's password.

```
$ su -

$ su -  pere

$ su - -c fdisk -l
```

Monitor SetUID / SetGID file

There are different types of UIDs supported by Linux to facilitate user management:

Real User ID
  The ID assigned by the system when a user logs in. All processes which are started by the user account will inherit the user's real user ID. The real user ID can be displayed with the id -u -r command.

Effective User ID
  The ID used by the system to determine the level of access the current process has. The setuid permission and su command, both discussed below, change the effective user ID of a program, providing a means for a user to run a program or access a file as another user without having to log off and log in to another user account. The effective user ID is displayed with the id command.

Normally, when a user accesses or executes a file, the user's real UID and GID are used to determine the level of access when executing the procedure.

Execute permissions:

SetUID

When an executable file (a program) has the SUID (Set User ID, setuid) permission, then the owner of the executable file becomes the Effective User ID to determine access and execute the procedure.

SetGID

When a file has the SGID (Set Group ID, setgid) permission, the group owner identity of the file is used as the Effective Group ID to determine file access and execute the procedure.

Files that have the SUID or SGID permission bits set can be used to create backdoors, which provide unauthorized access to the system. Diligent monitoring of the files that have these bits set is an important activity for the administrator to perform in order to keep the system secure.

```
$ ls -l /usr/bin/passwd /usr/bin/write
-rwsr-xr-x. 1 root root 37600 Jan 29  2020 /usr/bin/passwd
-rwxr-sr-x. 1 root tty  25248 May 20  2020 /usr/bin/write
```

```
$ ls -l /usr/bin | grep rws
-rwsr-xr-x. 1 root root       62832 Jan 28  2020 at
-rwsr-xr-x. 1 root root       83760 Nov 23  2020 chage
-rws--x--x. 1 root root       37760 May 20  2020 chfn
-rws--x--x. 1 root root       29504 May 20  2020 chsh
-rwsr-xr-x. 1 root root       67352 Jan 28  2020 crontab
-rwsr-xr-x. 1 root root       41752 Jan 28  2020 fusermount
-rwsr-xr-x. 1 root root       92296 Nov 23  2020 gpasswd
-rwsr-xr-x. 1 root root       58560 May 20  2020 mount
-rwsr-xr-x. 1 root root       47584 Nov 23  2020 newgrp
-rwsr-xr-x. 1 root root       37600 Jan 29  2020 passwd
-rwsr-xr-x. 1 root root       32888 Jan 30  2020 pkexec
-rwsr-xr-x. 1 root root       75736 May 20  2020 su
-rwsr-xr-x. 1 root root       41760 May 20  2020 umount

$ ls -l /usr/bin/ | grep r-s
-rwxr-sr-x. 1 root root       37920 May 20  2020 wall
-rwxr-sr-x. 1 root tty        25248 May 20  2020 write
```

```
$ find /usr/bin/ -perm -4000 -ls
  2658925        44 -rwsr-xr-x   1 root     root     41752 Jan 28  2020 /usr/bin/fusermount
  2635512        76 -rwsr-xr-x   1 root     root     75736 May 20  2020 /usr/bin/su
  2635497        60 -rwsr-xr-x   1 root     root     58560 May 20  2020 /usr/bin/mount
  2639018        40 -rws--x--x   1 root     root     37760 May 20  2020 /usr/bin/chfn
  2630604        92 -rwsr-xr-x   1 root     root     92296 Nov 23  2020 /usr/bin/gpasswd
  2639019        32 -rws--x--x   1 root     root     29504 May 20  2020 /usr/bin/chsh
  2630894        20 -rwsr-xr-x   1 root     root     17120 Apr  6  2021 /usr/bin/vmware-user-suid-wrapper
  2635515        44 -rwsr-xr-x   1 root     root     41760 May 20  2020 /usr/bin/umount
  2650960       188 ---s--x--x    1 root    root     191048 Jan 26  2021 /usr/bin/sudo
  2637420        40 -rwsr-xr-x   1 root     root     37608 Dec  1  2020 /usr/bin/fusermount-glusterfs
  2636136        36 -rwsr-xr-x   1 root     root     32888 Jan 30  2020 /usr/bin/pkexec
  2630607        48 -rwsr-xr-x   1 root     root     47584 Nov 23  2020 /usr/bin/newgrp
  2639062        64 -rwsr-xr-x   1 root     root     62832 Jan 28  2020 /usr/bin/at
  2659156        68 -rwsr-xr-x   1 root     root     67352 Jan 28  2020 /usr/bin/crontab
  2637181        44 -rwsr-xr-x   1 root     root     41760 Mar 19  2020 /usr/bin/fusermount3
  2657166        40 -rwsr-xr-x   1 root     root     37600 Jan 29  2020 /usr/bin/passwd
  2658237        84 -rwsr-xr-x   1 root     root     83760 Nov 23  2020 /usr/bin/chage

$ find /usr/bin/ -perm -2000 -ls
  2635523        28 -rwxr-sr-x   1 root     tty      25248 May 20  2020 /usr/bin/write
  2658295        40 -rwxr-sr-x   1 root     root     37920 May 20  2020 /usr/bin/wall
  2650849        48 -rwx--s--x   1 root     slocate  46240 Nov 27  2020 /usr/bin/locate
```

```
# find / -perm -4000 2> /dev/null -ls > check-perm.log

# find / -perm -type -f -4000 -o -perm -2000  2> /dev/null -ls > check-perm.log

# diff check-perm.log check-perm.old.log
```

```
 find /usr/bin -perm -u+s
/usr/bin/fusermount
/usr/bin/su
/usr/bin/mount
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/vmware-user-suid-wrapper
...

# sudo find /usr/bin -perm /6000
/usr/bin/fusermount
/usr/bin/su
/usr/bin/mount
/usr/bin/chfn
/usr/bin/gpasswd
...
```

Password Policy

Password policy:

- length
- password rules
- change password in the next login
- password aging
- /etc/login.defs

Commands:

```
$ passwd -l user
$ passwd -u user
$ passwd -e user
$ passwd [-x -n -w -i ] user
$ passwd -S user

$ usermod -L user
$ usermod -U user

$ chage -d 0 user
$ chage [-W -I -M -m -E -d ] user
$ chage -l user
```

Password aging

/etc/login.defs

PASS_MAX_DAYS

Maximum number of days a password is valid. A value of 99999 means “no maximum password age”.

PASS_MIN_DAYS

Minimum number of days a password is valid. A value of 0 means “no minimum password age”.

PASS_WARN_AGE

Number of days before password expiry that a warning message is given

/etc/login.defs

```
#15
$ cat /etc/login.defs
…
# Password aging controls:
#
#       PASS_MAX_DAYS  Maximum number of days a password may be used.
#       PASS_MIN_DAYS  Minimum number of days allowed between password changes.
#       PASS_MIN_LEN   Minimum acceptable password length.
#       PASS_WARN_AGE  Number of days warning given before a password expires.
#
PASS_MAX_DAYS  99999
PASS_MIN_DAYS  0
PASS_WARN_AGE  7
...
```

The chage command is used to update information related to password expiration. Using this command, the administrator can enforce a password changing and expiry policy for specific user accounts.

- Min days. Min days required to change to password (prevent change)
- Max days. Max days password is vàlid (force change periodically)
- Warning period. Warning message n days before change password date.
- Inactivity period. Set the inactivity period after passwd maxdays.
- Expiry date. Set the expiry date of the account (not password)

- Force the user to change password in the next login.
- Force the user to change password periodically.
- Prevent the user to revert to old passwd (no allow change password immediately).
- Set an expiration date.

```
# 16
# passwd -S pere
pere NP 2021-10-31 0 99999 7 -1 (Empty password.)

# chage -l pere
Last password change                                    : Oct 31, 2021
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

# chage -W 3 -m 5 -M 90  pere

# chage -l pere
Last password change                                    : Oct 31, 2021
Password expires                                        : Jan 29, 2022
Password inactive                                       : never
Account expires                                                : never
```

```
Minimum number of days between password change     : 5
Maximum number of days between password change     : 90
Number of days of warning before password expires  : 3

# chage -E 2022-01-29 pere

# chage -E $(date -d +180days +%Y-%m-%d) pere

# chage -l pere
Last password change                                : Oct 31, 2021
Password expires                                    : Jan 29, 2022
Password inactive                                   : never
Account expires                                            : Apr 29, 2022
Minimum number of days between password change     : 5
Maximum number of days between password change     : 90
Number of days of warning before password expires  : 3
```

Monitor users and logins

Monitor the current users in the system and the last logins

- who
- w
- last (/var/log/wtmp)

The following describes the output of the who command:

- Username
- terminal
- date
- Host

```
$ who
ecanet   :0           2021-11-17 16:42 (:0)
root    tty3          2021-11-17 18:03
guest   tty4          2021-11-17 18:03
ecanet   tty5         2021-11-17 18:03
```

```
$ w
 18:04:10 up  1:22,  4 users,  load average: 0.51, 0.69, 0.61
USER    TTY     LOGIN@   IDLE   JCPU   PCPU WHAT
ecanet   :0           16:42   ?xdm?  34:25   0.00s /usr/libexec/gdm-x-session
--run-script /usr/bin/gnome-session
root    tty3    18:03   17.00s  0.01s  0.01s -bash
guest   tty4    18:03   33.00s  0.01s  0.01s -bash
ecanet   tty5         18:03   26.00s  0.01s  0.01s -bash
```

```
$ last | head -n 15
guest   pts/3          ::1              Wed Nov 17 18:06 - 18:06  (00:00)
ecanet   tty5                           Wed Nov 17 18:03   still logged in
guest   tty4                            Wed Nov 17 18:03   still logged in
root    tty3                            Wed Nov 17 18:03   still logged in
ecanet   :0           :0               Wed Nov 17 16:42   still logged in
reboot   system boot  5.11.22-100.fc32 Wed Nov 17 16:41   still running
guest   pts/2          ::1              Tue Nov 16 17:36 - 17:39  (00:03)
ecanet   :0           :0               Mon Nov 15 19:02 - down  (1+01:13)
reboot   system boot  5.11.22-100.fc32 Mon Nov 15 19:02 - 20:16 (1+01:14)
ecanet   :0           :0               Sun Nov 14 12:49 - down   (08:58)
```

```
reboot   system boot  5.11.22-100.fc32 Sun Nov 14 12:47 - 21:47  (08:59)
ecanet   :0           :0               Sat Nov 13 08:33 - down   (12:47)
reboot   system boot  5.11.22-100.fc32 Sat Nov 13 08:32 - 21:20  (12:48)
ecanet   :0           :0               Fri Nov 12 15:35 - down   (05:32)
reboot   system boot  5.11.22-100.fc32 Fri Nov 12 15:35 - 21:08  (05:32)
```

```
# last guest
guest   pts/3          ::1              Wed Nov 17 18:42   still logged in
guest   pts/3          ::1              Wed Nov 17 18:06 - 18:06  (00:00)
guest   tty4                            Wed Nov 17 18:03   still logged in
guest   pts/2          ::1              Tue Nov 16 17:36 - 17:39  (00:03)
guest   tty2           :1               Wed Apr  7 17:20 - 17:28  (00:08)
guest   tty2           :1               Wed Apr  7 17:03 - down   (00:06)
wtmp begins Wed Apr  7 17:02:51 2021
```

Monitor network resources

Check the system for open network resources:

- nmap
- ss / netstat
- lsof | lsof -u user | lsof -i
- fuser

The nmap (network mapper) command is an open source tool used by system administrators for auditing networks, security scanning, and finding open ports on host machines. It is capable of scanning a host or the entire subnet to find open TCP and UDP ports. This tool is also used by attackers to find vulnerable ports.

Nmap port status

open
: Application on the target host is listening for incoming packets on this port

closed
: No applications are listening on this port

filtered
: The nmap command cannot identify if the port is open or closed because a network-level firewall or similar filter is not allowing probes to this port

unfiltered
: The nmap command can probe this port but does not have adequate information to conclude if it is open or closed

Nmap options

-sT and -sU	To scan for both TCP and UDP ports that may be open
-sP	To check which hosts are available on a network

```
$ nmap localhost
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-17 18:12 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00014s latency).
Other addresses for localhost (not scanned): ::1
```

```
Not shown: 996 closed ports
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
80/tcp  open  http
631/tcp open  ipp

# nmap -sT -sU example.com

# nmap -sP 192.168.1.3/24

$ nmap -p ssh,80 localhost

$ nmap -p 22-80 localhost

$ nmap 192.168.1.3-20

$ nmap 192.168.1.*

$ nmap 192.168.1.0/24 --exclude 192.168.1.7

$ nmap lms.pue.es
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-17 18:13 CET
Nmap scan report for lms.pue.es (51.15.184.105)
Host is up (0.67s latency).
rDNS record for 51.15.184.105: siurana.pue.es
Not shown: 994 filtered ports
PORT    STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
443/tcp  open  https
2522/tcp open  windb
8443/tcp open  https-alt
8888/tcp open  sun-answerbook
```

List open files

- lsof
- lsof -u user
- lsof -i

```
$ lsof  | wc -l
200552

# lsof | head
COMMAND PID   TID TASKCMD       USER    FD       TYPE             DEVICE   SIZE/OFF       NODE
NAME
systemd       1                 root  cwd      DIR          253,0         4096          2 /
systemd       1                 root  rtd      DIR          253,0         4096          2 /
systemd       1                 root  txt      REG          253,0  1730664 2757767 /usr/lib/systemd/systemd
systemd       1                 root  mem      REG          253,0       575147         263579 /etc/selinux/targeted/contexts/files/file_contexts.bin
systemd       1                 root  mem      REG          253,0  1913680 2626607 /usr/lib64/libm-2.31.so
systemd       1                 root  mem      REG          253,0       171160         2627542 /usr/lib64/libudev.so.1.6.17
systemd       1                 root  mem      REG          253,0        41096         2627262 /usr/lib64/libffi.so.6.0.2
systemd       1                 root  mem      REG          253,0       315872         2635339 /usr/lib64/libpcap.so.1.10.0
systemd       1                 root  mem      REG          253,0  1592736 2627433 /usr/lib64/libunistring.so.2.1.0


# lsof | wc -l
245712

# lsof -u pere

# lsof -u ramon | wc -l
10421
```

```
# lsof -i
COMMAND PID    USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
avahi-dae  819   avahi   15u  IPv4  32989   0t0  UDP *:mdns
avahi-dae  819   avahi   16u  IPv6  32990   0t0  UDP *:mdns
avahi-dae  819   avahi   17u  IPv4  32991   0t0  UDP *:47479
avahi-dae  819   avahi   18u  IPv6  32992   0t0  UDP *:38971
```

```
chronyd 858  chrony    6u  IPv4  32801       0t0  UDP localhost:323
chronyd 858  chrony    7u  IPv6  32802       0t0  UDP localhost:323
cupsd   952     root    9u  IPv6  36888       0t0  TCP localhost:ipp (LISTEN)
cupsd   952     root   10u  IPv4  36889       0t0  TCP localhost:ipp (LISTEN)
vsftpd  959     root    3u  IPv6  36085       0t0  TCP *:ftp (LISTEN)
httpd   1090    root    4u  IPv6  34303       0t0  TCP *:http (LISTEN)
dnsmasq    1239 dnsmasq       3u  IPv4  40977       0t0  UDP *:bootps
dnsmasq    1239 dnsmasq       5u  IPv4  40980       0t0  UDP mylaptop.edt.org:domain
dnsmasq    1239 dnsmasq       6u  IPv4  40981       0t0  TCP mylaptop.edt.org:domain
(LISTEN)

# lsof -i TCP
COMMAND  PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
cupsd   952     root    9u  IPv6  36888       0t0  TCP localhost:ipp (LISTEN)
cupsd   952     root   10u  IPv4  36889       0t0  TCP localhost:ipp (LISTEN)
vsftpd   959   root    3u  IPv6  36085       0t0  TCP *:ftp (LISTEN)
httpd   1090   root    4u  IPv6  34303       0t0  TCP *:http (LISTEN)
dnsmasq 1239 dnsmasq  6u  IPv4  40981       0t0  TCP mylaptop.edt.org:domain (LISTEN)
...

# lsof -i UDP
COMMAND PID     USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
avahi-dae  819   avahi   15u  IPv4  32989   0t0  UDP *:mdns
avahi-dae  819   avahi   16u  IPv6  32990   0t0  UDP *:mdns
avahi-dae  819   avahi   17u  IPv4  32991   0t0  UDP *:47479
avahi-dae  819   avahi   18u  IPv6  32992   0t0  UDP *:38971
chronyd 858  chrony    6u  IPv4  32801       0t0  UDP localhost:323
...
```

```
# lsof -i TCP:22
COMMAND  PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd    7955   root    5u  IPv4 135027       0t0  TCP *:ssh (LISTEN)
sshd    7955   root    7u  IPv6 135029       0t0  TCP *:ssh (LISTEN)
ssh     9528 ecanet     5u  IPv6 278408   0t0  TCP localhost:39208->localhost:ssh (ESTABLISHED)
sshd    9529   root     5u  IPv6 279265   0t0  TCP localhost:ssh->localhost:39208 (ESTABLISHED)
sshd    9540  guest     5u  IPv6 279265   0t0  TCP localhost:ssh->localhost:39208 (ESTABLISHED)

# lsof -i @127.0.0.1
COMMAND PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
chronyd 858 chrony    6u  IPv4  32801       0t0  UDP localhost:323
cupsd   952   root   10u  IPv4  36889       0t0  TCP localhost:ipp (LISTEN)
```

fuser command

The file user fuser command can also be used to display information about open files and sockets being accessed by processes.

Access codes that might be reported by the fuser command are:

- c The process is using the mount point or a subdirectory as its current directory.
- e The process is an executable file that resides in the mount point structure.
- f The process has an open file from the mount point structure.
- F The process has an open file from the mount point structure that it is writing to.
- r The process is using the mount point as the root directory.
- m The process is a memory-mapped (mmap) file or shared library.

```
# fuser -av .
                     USER           PID ACCESS COMMAND
/root:               root    6488 ..c.. bash
                     root    7480 ..c.. bash
                     root    7505 ..c.. dbus-broker-lau
                     root    7506 ..c.. dbus-broker
                     root    7508 ..c.. gvfsd

# fuser -av /home/guest
```

```
                     USER           PID ACCESS COMMAND
/home/guest:         guest   7561 ..c.. bash
                     guest   7588 ..c.. dbus-broker-lau
                     guest   7589 ..c.. dbus-broker
                     guest   7591 ..c.. gvfsd
                     guest   9542 ..c.. bash

$ fuser .
/home/guest:         7561c  7588c  7589c  7591c  9542c
```

Manage user limits

The user limit ulimit command is used to control resources that can be assigned by a user's login shell and child processes spawned from the shell. The system administrator may need to regulate the use of shared resources to prevent one process from using too much of a resource, preventing another process or user from having sufficient access to that resource.

- ulimit
- /etc/security/limits.conf
- /etc/security/limits.d

There can be two types of limits:

hard limits
: are set by the root user

soft limits
: can be set by either the root user or a regular user can set their own soft limit. The main constraint is that soft limits cannot exceed hard limits.

```
$ ulimit -a
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 29720
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files                      (-n) 1024
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 8192
cpu time                (seconds, -t) unlimited
max user processes              (-u) 29720
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

```
# ulimit -Ha
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 29720
```

```
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files                      (-n) 524288
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) unlimited
cpu time                (seconds, -t) unlimited
max user processes              (-u) 29720
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

In the list the short option indicates the option to change the limit:

-S soft limit

-H hard limit

-c Maximum size of core files created

-d Maximum size of the process's data segment

-s Maximum stack size

-u Maximum number of processes available to a single user

-v Maximum virtual memory available to the user's process

-l Maximum size that may be locked into memory

```
[root@mylaptop ~]# ulimit -a | grep core
core file size          (blocks, -c) unlimited

[root@mylaptop ~]# ulimit -c 0

[root@mylaptop ~]# ulimit -a | grep core
core file size          (blocks, -c) 0

[root@mylaptop ~]# ulimit -c unlimited

[root@mylaptop ~]# ulimit -a | grep core
core file size          (blocks, -c) unlimited
```

```
# ulimit -Hr 10

# ulimit -Ha
real-time priority              (-r) 10
...

# ulimit -Hr 0
```

/etc/security/limits.conf and /etc/security/limits.d directory

- domain
- type
- item
- value

```
#Each line describes a limit for a user in the form:
#<domain>      <type>  <item>  <value>
```

```
#<domain> can be:
#       - a user name
#       - a group name, with @group syntax
#       - the wildcard *, for default entry
#       - the wildcard %, can be also used with %group syntax,
```

```
#              for maxlogin limit
```

```
#<type> can have the two values:
#       - "soft" for enforcing the soft limits
#       - "hard" for enforcing hard limits
```

```
#<item> can be one of the following:
#       - core - limits the core file size (KB)
#       - data - max data size (KB)
#       - fsize - maximum filesize (KB)
#       - memlock - max locked-in-memory address space (KB)
#       - nofile - max number of open file descriptors
#       - rss - max resident set size (KB)
#       - stack - max stack size (KB)
#       - cpu - max CPU time (MIN)
#       - nproc - max number of processes
#       - as - address space limit (KB)
#       - maxlogins - max number of logins for this user
#       - maxsyslogins - max number of logins on the system
#       - priority - the priority to run user process with
#       - locks - max number of file locks the user can hold
#       - sigpending - max number of pending signals
#       - msgqueue - max memory used by POSIX message queues (bytes)
#       - nice - max nice priority allowed to raise to values: [-20, 19]
#       - rtprio - max realtime priority
```

```
# cat /etc/security/limits.conf
#<domain>      <type>  <item>         <value>
#*             soft    core           0
#*             hard    rss            10000
#@student      hard    nproc          20
#@faculty      soft    nproc          20
#@faculty      hard    nproc          50
#ftp           hard    nproc          0
#@student      -       maxlogins      4
```

Example Exercises

[SUID/SGID]

1. Find all files with the SUID (and other permissions) set in /usr/bin.
2. Find all files with either the SUID or the SGID set in /usr/bin.

[Password Policy]

3. Using chage make the password be valid for 365 days.
4. Using chage make user change password on next login.
5. Set warning period to 7 days and account expiration date to August, 20th 2050.
6. Print user's current password expiry information.

[network monitor]

7. List all listening *udp* sockets on your machine using netstat.
8. Scan ports 80 through 443 on host A.B.C.D using nmap.
9. Show network files for localhost on port 22 using lsof.
10. Show network files for A.B.C.D on port 22 using lsof.

[limits]

11. Display soft limits on the maximum real-time priority:
12. Display all hard limits
13. Set the soft limits on the real-time priority to 15. Show it and return the value to 0.

[sudoers]

```
Host_Alias SERVERS = 192.168.1.7, server1, server2
User_Alias REGULAR_USERS = john, mary, alex
User_Alias PRIVILEGED_USERS = mimi, alex
User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS
Cmnd_Alias WEB_SERVER_STATUS = /usr/bin/systemctl status apache2

root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=WEB_SERVER_STATUS
%sudo   ALL=(ALL:ALL) ALL
```

14. Can alex check the status of the Apache Web Server on any host? Why?
15. Can Carol?

16. Realitza els exercicis indicats a: 110.1 Perform security administration tasks
17. Realitza els exercicis del Question-Topics 110.1