

GNU/LINUX TUTORIAL

PART II: USERS, PERMISSIONS, PROCESSES, FILE TREE

§5: USERS

- Commands are executed by users.
- Each user may belong to groups.
- `root` is the most powerful user.
- `sudo` command allows to execute other commands as `root`.
- Special users and groups may exist to run software.

WHO AM I? 🤨

```
user@cosmos:~$ whoami  
user
```

```
user@cosmos:~$ id  
uid=1000(user) gid=1000(user) groups=1000(user),27(sudo)
```

```
user@cosmos:~$ id root  
uid=0(root) gid=0(root) groups=0(root)
```

- Each user has
 - numeric User ID (UID),
 - home directory, e.g. `/home/user`,
 - login shell, e.g. `/bin/bash` or `/usr/sbin/nologin` if disabled.
- `root` has UID 0.

- Each group has numeric Group ID (GID).
- `sudo` group: those who can run `sudo`.

MANAGING PASSWORDS



```
user@cosmos:~$ passwd
Changing password for user.
Current password:
New password:
Retype new password:
passwd: password updated successfully
```

CHANGING ROOT PASSWORD



```
user@cosmos:~$ sudo passwd root
New password:
Retype new password:
passwd: password updated successfully
```

```
user@cosmos:~$ su
Password:
root@cosmos:/home/user# exit
exit
user@cosmos:~$
```


MANAGING USERS



Command

Action

```
useradd --create-home --shell /bin/bash  
newuser
```

add user

```
groupadd newgroup
```

add group

```
passwd newuser
```

set password

```
sudo usermod -aG sudo newuser
```

add to `sudo`
group

```
su newuser
```

login as
`newuser`

```
userdel newuser
```

delete `newuser`

WHERE USERS AND PASSWORDS ARE?

- `/etc/passwd` : list of users.
Can be accessed by everyone.
- `/etc/group` : list of groups.
Can be accessed by everyone.
- `/etc/shadow` : list of users and hashes of their passwords.
Can be accessed by `root`.

PRACTICE

- Check `/etc/passwd`, `/etc/group`, and `/etc/shadow`.
- Create a new user with home directory and setup their password.
- Add user to the `sudo` group.
- Check `/etc/passwd`, `/etc/group`, and `/etc/shadow` again.
- Login as that user with `su`.
- Exit shell with `exit`.
- Delete the new user.

§6: PERMISSIONS



- Permissions are important
 - for security reasons,
 - not to touch data you're not supposed to.
- Managed on three levels:
 - file owner,
 - file group,
 - everyone.

MANAGING OWNER AND GROUP



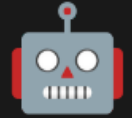
- `chown <owner> <file>`
- `chgrp <group> <file>`

PERMISSIONS

Permission	Regular files	Directories
 read	read contents	list files inside
 write	modify contents	create, rename, delete files
 execute	execute	enter with 

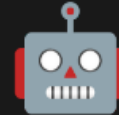
 Set  only on executable files (binaries, scripts) and directories.

REPRESENTED BY A NUMBER



Number	Binary	Permissions
0	0b000	---
1	0b001	--x
2	0b010	-w-
3	0b011	-wx
4	0b100	r--
5	0b101	r-x
6	0b110	rw-
7	0b111	rwX

FORMAT



```
user@cosmos:/usr/local/lib$ stat /usr/bin
```

```
Access: (0755/drwxr-xr-x)  Uid: (  0/  root)   Gid: (  0/  root)
```

```
  |  |  |  |  |  └─ all: r-x
```

```
  |  |  |  |  └─ group: r-x
```

```
  |  |  |  └─ owner: rwx
```

```
  |  |  └─ file type: (d)irectory
```

```
  |  └─ 755 = rwxr-xr-x
```

```
"sticky bit" = 0
```

FILE TYPES



Character	Type
-	regular file
d	directory
l	symbolic link
c	character special device
b	block special device
p	FIFO
s	socket

EXAMPLES

- `777` = `rw-rw-rw-` : everything is permitted to everyone.
- `666` = `rw-rw-rw-` : permission of the beast, anyone can read and modify the file.
- `644` = `rw-r--r--` : owner can read and modify; others can read.
- `755` = `rw-r-xr-x` : directory is read-only, except for the owner.

REAL EXAMPLES

File	Permissions	
/etc/passwd	0644	-rw-r--r--
/bin/bash	0755	-rwxr-xr-x
/tmp	1777	drwxrwxrwt
/home	0755	drwxr-xr-x
/home/user	0755	drwxr-xr-x

CHANGING PERMISSIONS



```
user@cosmos:~$ touch test.txt
```

```
user@cosmos:~$ stat test.txt
```

```
Access: (0644/-rw-r--r--)  Uid: (1000/  user)   Gid: (1000/  user)
```

```
user@cosmos:~$ chmod +x test.txt
```

```
user@cosmos:~$ stat test.txt
```

```
Access: (0755/-rwxr-xr-x)  Uid: (1000/  user)   Gid: (1000/  user)
```

```
user@cosmos:~$ chmod 666 test.txt
```

```
user@cosmos:~$ stat test.txt
```

```
Access: (0666/-rw-rw-rw-)  Uid: (1000/  user)   Gid: (1000/  user)
```

WHAT IS THE DEFAULT? 🤔

- `umask` : print file mode mask.
- `umask 022` : set file mode mask.
- Default permissions:
 - `666` - `umask` for regular files,
 - `777` - `umask` for directories.

```
user@cosmos:~$ umask
```

```
0022
```

```
user@cosmos:~$ touch test.txt
```

```
user@cosmos:~$ mkdir test
```

```
user@cosmos:~$ ls -ll
```

```
drwxr-xr-x 1 user user 0 Aug  8 20:11 test
```

```
-rw-r--r-- 1 user user 0 Aug  8 20:11 test.txt
```

Remember: `chmod`, `chown`, `chgrp`.

§7: PROCESSES

- Each process has its unique Process ID (PID).
- `pidof {name}` (e.g. `pidof bash`) finds PIDs by name.

PS: PROCESS INFO

- `sudo apt install procs` : utilities for working with processes.

Command	Action
<code>ps</code>	list current processes
<code>ps -e</code>	list all processes
<code>ps -C bash</code>	processes with command <code>bash</code>
<code>ps -U root</code>	processes run by <code>root</code> user
<code>pstree</code>	process tree

MONITORING PROCESSES



- top

```
top - 19:57:02 up 4:24, 0 users, load average: 3.86, 1.34, 0.61
Tasks: 120 total, 1 running, 118 sleeping, 1 stopped, 0 zombie
%Cpu(s): 70.3 us, 11.7 sy, 0.0 ni, 6.4 id, 0.8 wa, 0.0 hi, 0.1 si
MiB Mem : 6603.6 total, 6592.3 free, 5.1 used, 6.2 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 6598.5 available
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME-
20995	abeshen+	20	0	7894004	2.2g	297812	S	305.7	34.0	2:44.67
21501	abeshen+	20	0	2475216	55784	26788	S	23.0	0.8	0:00.69
21495	abeshen+	20	0	2475216	52100	26660	S	22.0	0.8	0:00.66
499	abeshen+	20	0	72268	28936	8140	S	1.7	0.4	2:16.57

KILL



Command	Action
<code>kill {pid}</code>	sends <code>SIGTERM</code>
<code>killall {name}</code>	sends <code>SIGTERM</code>
<code>kill -KILL {pid}</code>	sends <code>SIGKILL</code>
<code>killall -KILL {name}</code>	sends <code>SIGKILL</code>

- `SIGTERM` asks process politely to terminate.
- `SIGKILL` kills the process (e.g. when not responding)

EXAMPLE



- Run `less /etc/passwd`.
- Press `C-z`.
- Check `top`.
- Try `killall less`.
- Check `top` again.
- Try `killall -KILL less`.

§8: FILE SYSTEM








WHAT'S IN THE ROOT /? 🤔

```
user@cosmos:~$ ls -p /  
bin/    dev/    home/   lib64/  mnt/    proc/   run/  
srv/    tmp/    var/    boot/   etc/    lib/    media/  
opt/    root/   sbin/   sys/    usr/
```







- Linux distributions more or less follow the **Filesystem Hierarchy Standard**.

Dir	Contents	Dir	Contents
/bin	basic bins	/proc	process info
/boot	bootloader	/root	root home
/dev	devices	/run	run-time data
/etc	configuration	/sbin	system binaries
/home	home dirs	/srv	data served
/lib(64)	libs	/sys	system info
/media	removable media	/tmp	temporary files
/mnt	mounted fs	/usr	user shareable, read-only data
/opt	addon pkgs	/var	variable data




BINARIES

```
|—  bin    ← essential binaries available to all users
|   |—  bash    ← some basic commands
|   |—  date
|   |—  mv
|   |—  rm
|   |—  tar
|   |— ...
|
|—  sbin  ← system binaries available to root
      (boot, recovery, etc.)
```

LIBRARIES









```
|—  lib      ← libraries: shared objects = SO,  
|—  lib64      kernel modules, etc.  
|                                     (like DLLs in W*ndows)  
└—  usr  
    |—  lib      ← user libraries  
    |  
    └—  local  
        |  
        └—  lib
```

HOME DIRECTORIES

```
|—  home    ← user home directories
|   |—  user      (user's data, installations, configurations)
|   |— ...
|
|—  root    ← root's home
```











DEVICES

DEVFS MANAGED BY KERNEL

```
└─  dev    ← device files
    └─  null    ← null-device (discards data)
    └─  pts
        └─  0      ← pseudoterminal connected to standard input
        └─ ...
    └─  random  ← (pseudo)random byte source, blocks when out of data
    └─  tty     ← current console (TeleTYpewriter)
    └─  urandom ← (pseudo)random byte source, non-blocking (unlike random)
    └─  zero    ← provides zero-characters 0x00
    └─ ...
```



INFO MANAGED BY KERNEL













```
|—  proc    ← processes
|
|   |—  cpuinfo    ← CPU info
|   |—  meminfo    ← memory info
|   |—  version    ← kernel version
|   |
|   |—  {pid}      ← directory attached to process ID
|       |—  cmdline    ← command that started the process
|       |—  cwd        ← symlink to the current working dir
|       |—  environ    ← environment variables
|       |—  exe        ← symbolic link to the executable
|       |—  fd         ← directory with file descriptors
```

TEMPORARY DATA











- |  run ← run-time variable data; cleared on boot
- |
- |  tmp ← temporary files,
may be created by everyone,
don't persist

/usr

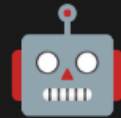
```
└─  usr    ← data shared between users
    │
    └─  bin
    │
    └─  include ← C header files
    │
    └─  lib
    │
    └─  local  ← locally installed software
        │
        └─  bin
        │
        └─  include
        │
        └─  lib
        │
        └─  share
        │
        └─  src
        │
        └─ ...
```


OTHER STUFF



- |—  etc ← 'etcetera'; host-specific configuration files
- |
- |—  opt ← additional software packages
- |
- |—  srv ← data served by the system
- |
- |—  var ← variable data
 - |—  cache
 - |—  lock
 - |—  log
 - |—  mail

BOOT AND FILE SYSTEMS



- |— boot ← boot loader files
- |
- |— media ← mount points for removable devices
- |
- |— mnt ← mount points for other file systems

EXPLORE IT YOURSELF! 💪

- Read `cpuinfo`, `meminfo`, `version` inside `/proc` as regular files.
- For a process with some ID, explore `/proc/{pid}`.
- What's the difference between `/bin` and `/usr/bin`?
- Check out `/var/log/apt/history.log`.

NEXT TIME

- IO redirection and pipelines.
- Pattern matching, aliases, variables, `$PATH`.
- Anatomy of scripts: shebang.