Permissions

COMS10012 Software Tools

root

UNIX model

- normal users: can do only what they are explicitly allowed to do
- **root** (administrator, superuser):
can do everything

root is user 0

```
/* How to check someone is root, in
   Linux kernel code. */

if (current_cred()->uid != 0)
     return -EPERM;
```

su and sudo

```
su [USER [COMMAND]]
```

Switches to USER (default is root) and asks for USER's password, if they have one.
If no COMMAND is given, starts their shell.

```
sudo COMMAND
```

If permitted by the system settings, runs command as root (user can be configured).
Can ask for your own password.

security

Best practice:

- root cannot log in directly at all (certainly not remotely over ssh)
- users with admin rights run sudo for individual commands as root
- if you really need a root shell, **sudo su**

capabilities

- traditional UNIX: root or user, all or nothing
- nowadays: extra capabilities system for individual tasks (open ports, shut down system, ignore file permissions etc.)

access control

inodes: mode

```
struct inode {
  // ...
  uid_t i_uid;
  gid_t i_gid;
  // ...
  umode_t i_mode;
  // ...
}
```

type:
1000 = file
0100 = directory
1010 = symlink ...

T	T	T	T	SU	SG	ST	RU
WU	XU	RG	WG	XG	RO	WO	XO

R = read	U = user
W = write	G = group
X = exec	O = others

x bit

On a file **app**, +x lets you run the file as **`./app`**
On the PATH, you can also just do **app**

If the file starts with `#!PROGRAM` then it runs as a script, e.g. `#!/usr/bin/python` (shebang).

`$ sh app` or **`$ python app`** doesn't need +x.

directories

r bit: can read contents of directory
This lets you list the files.

w bit: can write contents of directory
This lets you create, delete, rename files.

x bit: can cd to the directory
(technically, read inodes in the directory)

mode bits

```
$ ls -l /

drwxr-xr-x    2 root     root           4096 Oct  9 13:14 bin
drwx------    2 root     root           4096 Sep 12 13:48 root
drwxrwxrwt    4 root     root           4096 Oct 23 09:35 tmp
drwxr-xr-x   10 root     root           4096 Oct  9 13:13 usr
drwxrwxrwx    1 vagrant  vagrant        4096 Oct  8 15:34 vagrant
drwxr-xr-x   12 root     root           4096 Oct  9 12:44 var
...
```

mode bits

```
$ ls -l /bin
```

```
lrwxrwxrwx 1 root root     12 Sep 12 08:45 arch -> /bin/busybox
lrwxrwxrwx 1 root root     12 Sep 12 08:45 ash -> /bin/busybox
lrwxrwxrwx 1 root root     12 Sep 12 08:45 base64 -> /bin/busybox
-rwxr-xr-x 1 root root 735488 May  3  2019 bash
...
```

mode bits

```
$ ls -l ~/.ssh
```

```
drwx------  2 vagrant  vagrant   4096 Oct 16 16:04 .
drwxr-sr-x  5 vagrant  vagrant   4096 Oct 16 08:40 ..
-rw-------  1 vagrant  vagrant    798 Oct  9 12:39 authorized_keys
-rw-r--r--  1 vagrant  vagrant    142 Oct 16 16:07 config
-rw-------  1 vagrant  vagrant    411 Oct 16 15:55 id_ed25519
-rw-r--r--  1 vagrant  vagrant    430 Oct 16 16:02 known_hosts
```

chmod

```
$ chmod go-rwx id_ed25519


$ chmod 0600 id_ed25519
$ chmod  755 program
```

changing owner and group

```
$ sudo chown [-R] USER FILE
```

```
$ chgrp [-R] GROUP FILE
```

special bits

setuid

```
/usr/bin$ ls -l sudo
-rwsr-xr-x  1 root   root   120048 Feb  5  2020 sudo
```

One way to give users limited extra rights:

binary with setuid (SU) bit set runs as owner.

Examples: **`su, sudo, passwd, (shutdown) ...`**

/etc/sudoers

```
root ALL=(ALL) ALL
vagrant ALL=(ALL) NOPASSWD: ALL
```

```
SOURCE HOST=(TARGET) [OPTION:] CMD [,CMD]
```

```
%shutdown ALL=(root) /usr/sbin/shutdown
```

directories

```
$ ls -l /
```

```
drwxrwxrwt  4 root     root    4096 Oct 23 09:35 tmp
```

sticky bit (ST) = only file owner, dir owner and root can rename/delete files.

```
$ ls -l /home/vagrant
```

```
drwxr-sr-x  5 vagrant  vagrant 4096 Oct 16 08:40 vagrant
```

SGid bit on a folder: new files get directory's group.

administration

users and groups

```
$ sudo adduser Tim
$ sudo addgroup staff
$ sudo addgroup Tim staff
```

/etc/group

```
GROUP:PW:GID:USER[,USER...]

root:x:0:root
wheel:x:10:root
mail:x:12:mail
vagrant:x:1000:
vboxsf:x:101:
...
```

/etc/passwd

```
name:pw:UID:GID:GECOS:homedir:shell
```

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
...
nobody:x:65534:65534:nobody:/:/sbin/nologin
vagrant:x:1000:1000:Linux User,,,:/home/vagrant:/bin/bash
```