User accounts and file permission

Content

- User and group accounts
- Manage user and group accounts
- Access permission
- File permission
- Directory permission
- Manage the access permission

User account

- Normal users
- Administrators
- Group users
- A user account needs
  - Username, password, and home directory (/home/username)
  - Group (each user need to belong to only one primary group through a user can belong to many groups)
  - All user information is stored in the file: /etc/passwd

/etc/passwd

- **Username:password:UID:GID:Info:Home:Shell**
- **Username**: It is used when user logs in. It should be between 1 and 32 characters in length.
- **Password**: An x character indicates that encrypted password is stored in /etc/shadow file.
- **User ID (UID)**: Each user must be assigned a user ID (UID). UID 0 (zero) is reserved for root and UIDs 1-99 are reserved for other predefined accounts. Further UID 100-999 are reserved by system for administrative and system accounts/groups.
- **Group ID (GID)**: The primary group ID (stored in /etc/group file)
- **User ID Info**: The comment field. It allow you to add extra information about the users such as user's full name, phone number etc. This field use by finger command.
- **Home directory**: The absolute path to the directory the user will be in when they log in. If this directory does not exists then users directory becomes /
- **Command/shell**: The absolute path of a command or shell (/bin/bash). Typically, this is a shell. Please note that it does not have to be a shell.

/etc/shadow

- **User:Pwd:Last pwd change :Minimum:Maximum:Warn:Inactive :Expire**
- User name : It is your login name
- Password: It your encrypted password. The password should be minimum 6-8 characters long including special characters/digits
- Last password change (lastchanged): Days since Jan 1, 1970 that password was last changed
- Minimum: The minimum number of days required between password changes i.e. the number of days left before the user is allowed to change his/her password
- Maximum: The maximum number of days the password is valid (after that user is forced to change his/her password)
- Warn : The number of days before password is to expire that user is warned that his/her password must be changed
- Inactive : The number of days after password expires that account is disabled
- Expire : days since Jan 1, 1970 that account is disabled i.e. an absolute date specifying when the login may no longer be used

Group users

- Each user can belong to one or many groups
  - One group = group name + member list
  - Can share files among members of a group.
  - The list of groups is stored in /etc/group
  - root can create additional groups, aside default groups that the Linux OS created

/etc/group

- **group_name:Password:Group ID (GID): Group List**
- group_name: It is the name of group. If you run ls -l command, you will see this name printed in the group field.
- Password: Generally password is not used, hence it is empty/blank. It can store encrypted password. This is useful to implement privileged groups. X means passwd is stored in /etc/gshadow
- Group ID (GID): Each user must be assigned a group ID. You can see this number in your /etc/passwd file.
- Group List: It is a list of user names of users who are members of the group. The user names, must be separated by commas.

/etc/gshadow

- *Group name* — The name of the group. Used by various utility programs as a human-readable identifier for the group.
- *Encrypted password* — The encrypted password for the group. If set, non-members of the group can join the group by typing the password for that group using the newgrp command. If the value of this field is !, then no user is allowed to access the group using the newgrp command. A value of !! is treated the same as a value of ! — however, it also indicates that a password has never been set before. If the value is null, only group members can log into the group.
- *Group administrators* — Group members listed here (in a comma delimited list) can add or remove group members using the gpasswd command.
- *Group members* — Group members listed here (in a comma delimited list) are regular, non-administrative members of the group.

Tools to manage user/group

- useradd/mod/del
- passwd
- groupadd/mod/del
- gpasswd
- sg/newgrp
- su
- users/groups
- id

Ownership

- Each file belongs to a user and a group
  - The file/directory creator will be the owner and the primary group of the owner will be the group owner.
- The ownership allows to determine the permission of a user to files or directories.

Access permission

- r : read
  - Allow to read the content of a file or directory
- w : write
  - Allow to change the content of a file
  - Allow to add or remove files inside a directory
- x : execute
  - Allow to execute a binary file
  - Allow to change to a directory

Permission groups

- There are three user based permission groups each file/directory :
  - u (owner) : the owner of the file
  - g (group) : users belong to the primary group of the owner
  - o (others) : other users.
- Each permission group has a set of permission (r, w, x).

Examples

```
$ ls -l
----rw-rw- 1 tuananh  user1   16 Feb 10 19:12 test1.txt
-rw-rw-rw- 1 tuananh  user1   16 Feb 10 19:12 test2.txt
drw-r--r-- 2 tuananh  user1  512 Feb 10 19:14 vanban
$ whoami
tuananh
$ cat test1.txt
cat: test1.txt: Permission denied
$ cat test2.txt
Un fichier de test
$ cp test2.txt vanban
cp: vanban: Permission denied
```

Some notes

- To add more files to a directory, you need « w » permission
- To delete or move a file, a user needs « w » permission of that file
  - The file deletion depends on the permission of the directory that that file belongs to
- To secure data, the owner can even remove the « r » permission to other users.
- To limit access to file system, user can remove « x » permission to the root of the file system (/).

Change permission (1)

$chmod <mode> <files>

```
set_uid set-gid sticky user group other
                       rwx   --x   --x
   1       1      0    111   001   001
           6            7     1     1

$ chmod  6711  test
$ ls -l test
-rws--s--x  1  tuananh  user1  Mar 10 10:20  test
$ chmod  711  test
$ ls -l test
-rwx--x--x  1  tuananh  user1  Mar 10 10:20  test
```

Change permission (2)

```
$chmod <ugoa><+-=><rwsx> <files>
```

- u | g | o | a (all)
- Operation
  - + (add one or some permission)
  - - (remove one or some permission)
  - = (assign one or some permission)
- Permission = r | w | x | s

Examples

```
$ ls -l test.txt
-rw-rw-r-- 1 tuananh user1  150 Mar 19 19:12 test.txt
$ chmod o+w test.txt
$ ls -l test.txt
-rw-rw-rw- 1 tuananh user1  150 Mar 19 19:12 test.txt
$ chmod a-rw test.txt
$ ls -l test.txt
---------- 1 tuananh user1  150 Mar 19 19:12 test.txt
$ cat test.txt
cat: test.txt: Permission denied
```

Default permission while creating a file

- You can view or change the default permission of a file while creating by using command umask

```
$umask
022
```

  - 0 means no limitation (rwx)
  - 2 mean no writing permission (w) but can read or execute (r-w).

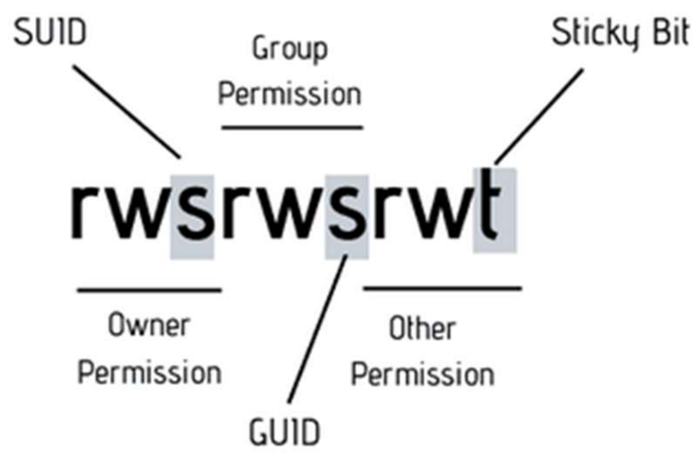
```
$umask 022
```

Change the owner and group

- $chown [-R] <utilisateur> <files>
  - Change the ownber
- $chgrp <group> <files>
  - Change the group
  - Use option –R to change the ownership of child files/folders of a folder
- Those commands can only be executed by the root user

Some special permission with binary files

- set-uid: -rw**s** --- ---
  - Program will be run under the permission of the owner
- set-gid: - --- rw**s** ---
  - Program will be run under the permission of the group owner
- bit sticky
  - The sticky bit works on the directory. With sticky bit set on a directory, all the files in the directory can only be deleted or renamed by the file owners only or the root

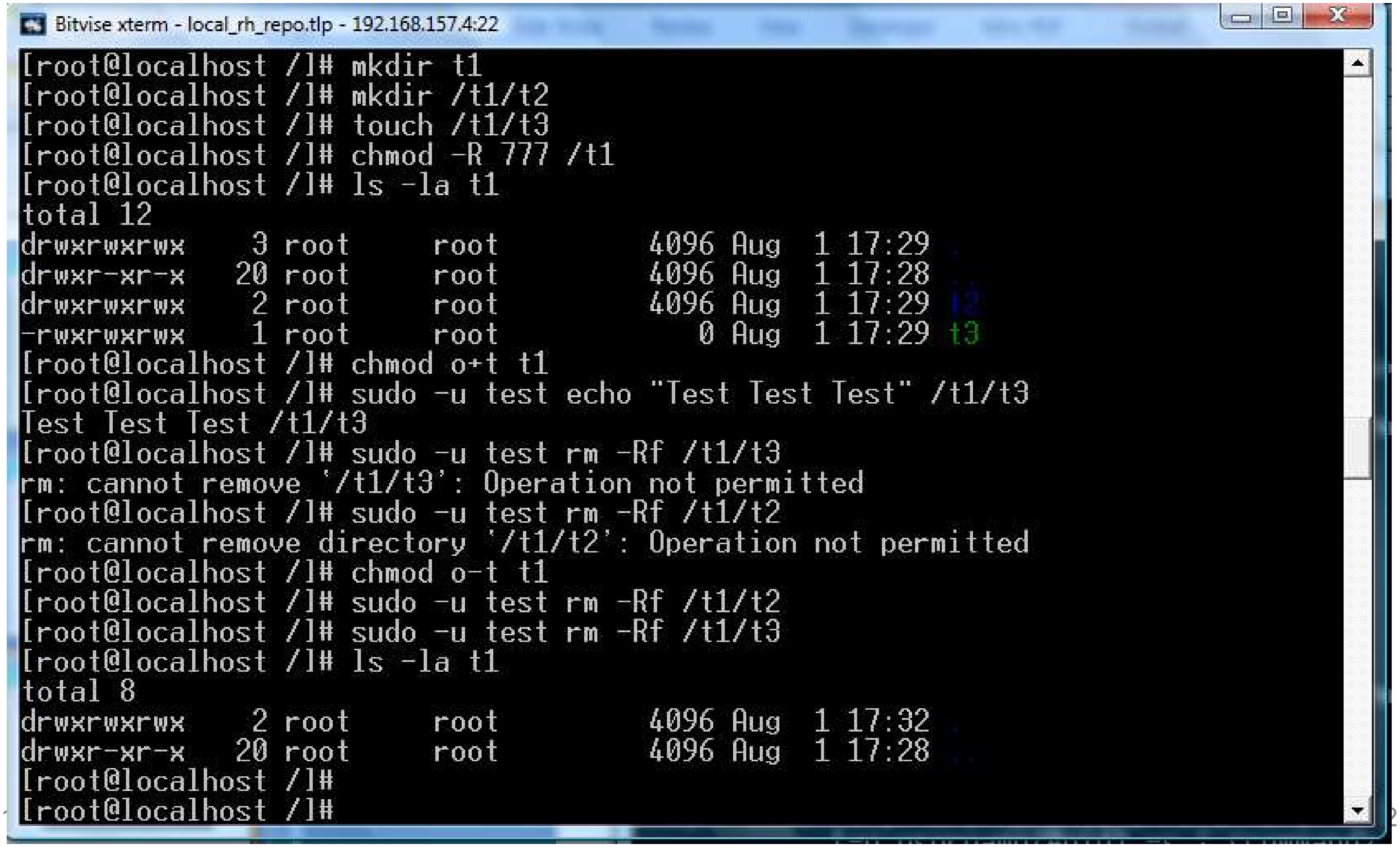


Examples

```
$ ls -l /etc/passwd
-rw-rw---- 1 root  root  568 Feb 10 19:12 passwd
$ ls -l /bin/passwd
-rwsrws--x 1 root  root 3634 Feb 10 19:12 passwd
```

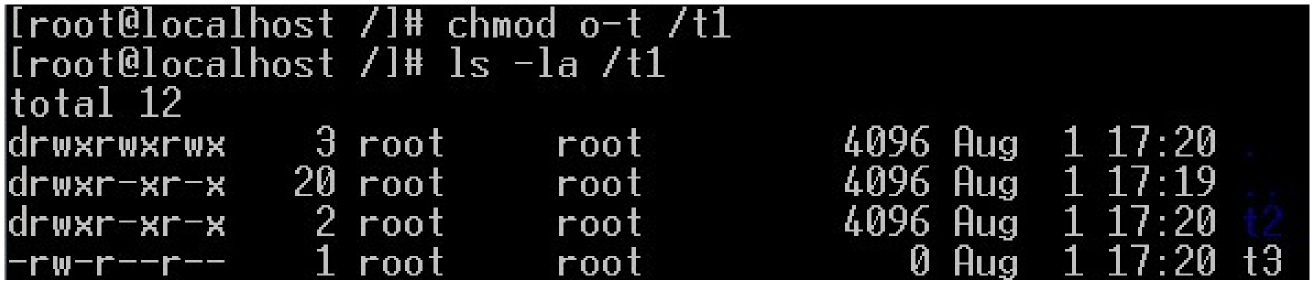
- When a user change their password (/bin/passwd), that user « borrow » the root permission to change the password of their account

Sticky bit example

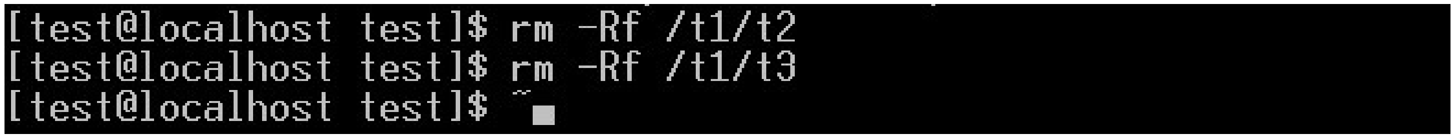

Bitvise xterm - local_rh_repo.tlp - 192.168.157.4:22

```
[root@localhost /]# mkdir t1
[root@localhost /]# mkdir /t1/t2
[root@localhost /]# touch /t1/t3
[root@localhost /]# chmod -R 777 /t1
[root@localhost /]# ls -la t1
total 12
drwxrwxrwx    3 root     root         4096 Aug  1 17:29 .
drwxr-xr-x   20 root     root         4096 Aug  1 17:28 ..
drwxrwxrwx    2 root     root         4096 Aug  1 17:29 t2
-rwxrwxrwx    1 root     root            0 Aug  1 17:29 t3
[root@localhost /]# chmod o+t t1
[root@localhost /]# sudo -u test echo "Test Test Test" /t1/t3
Test Test Test /t1/t3
[root@localhost /]# sudo -u test rm -Rf /t1/t3
rm: cannot remove '/t1/t3': Operation not permitted
[root@localhost /]# sudo -u test rm -Rf /t1/t2
rm: cannot remove directory '/t1/t2': Operation not permitted
[root@localhost /]# chmod o-t t1
[root@localhost /]# sudo -u test rm -Rf /t1/t2
[root@localhost /]# sudo -u test rm -Rf /t1/t3
[root@localhost /]# ls -la t1
total 8
drwxrwxrwx    2 root     root         4096 Aug  1 17:32 .
drwxr-xr-x   20 root     root         4096 Aug  1 17:28 ..
[root@localhost /]#
[root@localhost /]#
```

Sticky bit example

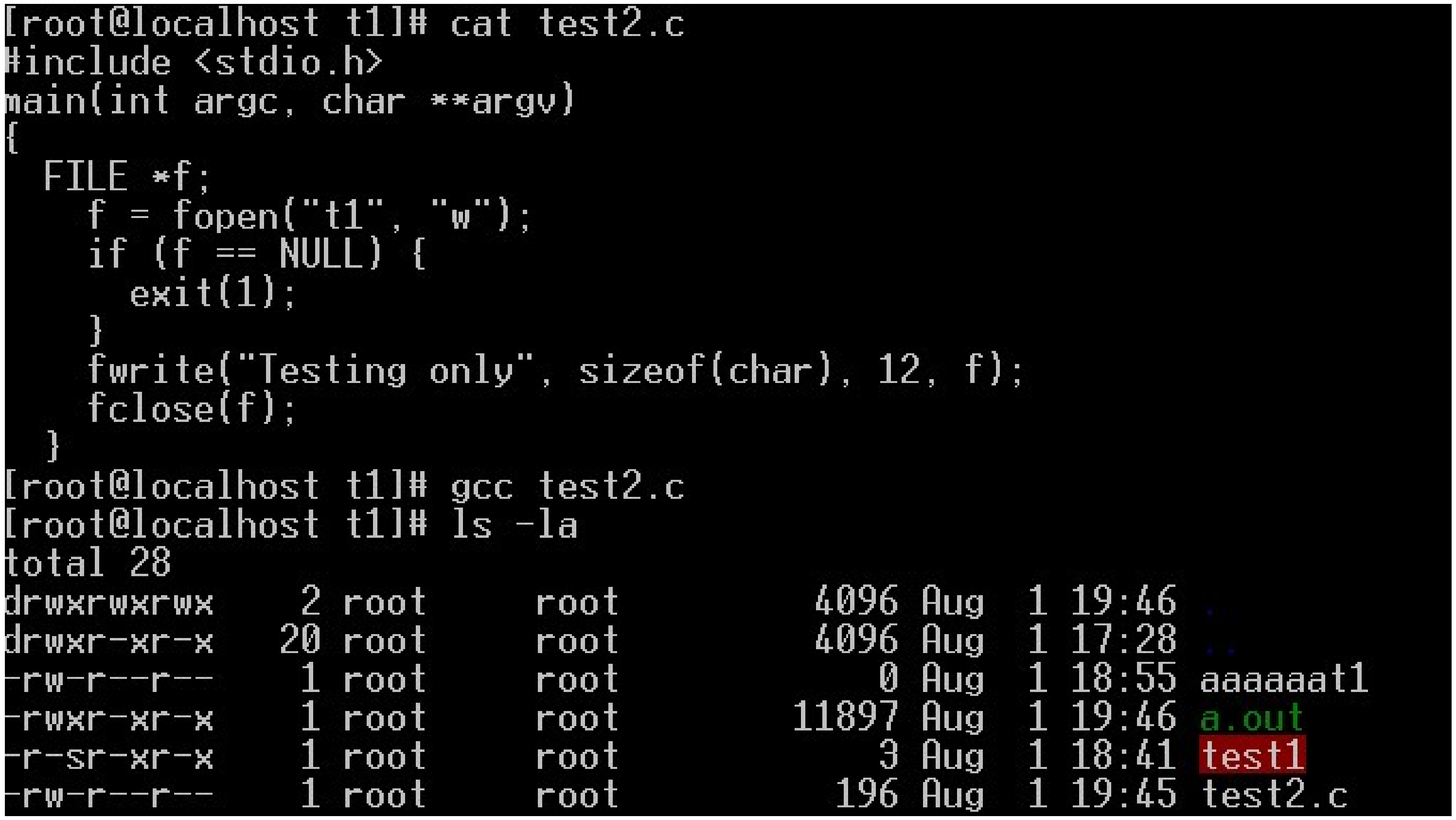


```
[root@localhost /]# chmod o-t /t1
[root@localhost /]# ls -la /t1
total 12
drwxrwxrwx    3 root     root         4096 Aug  1 17:20 .
drwxr-xr-x   20 root     root         4096 Aug  1 17:19 ..
drwxr-xr-x    2 root     root         4096 Aug  1 17:20 t2
-rw-r--r--    1 root     root            0 Aug  1 17:20 t3
```



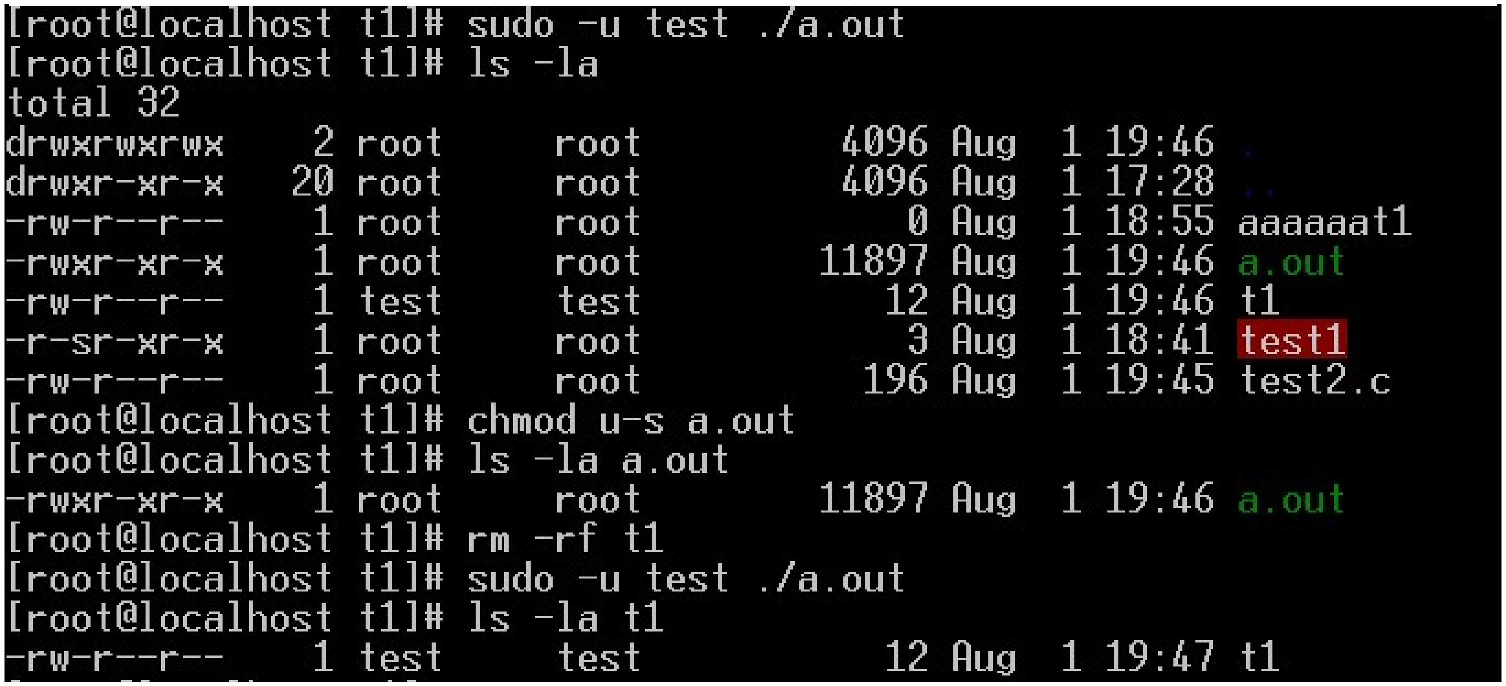
```
[test@localhost test]$ rm -Rf /t1/t2
[test@localhost test]$ rm -Rf /t1/t3
[test@localhost test]$ ~
```

Suid bit-example



```
[root@localhost t1]# cat test2.c
#include <stdio.h>
main(int argc, char **argv)
{
  FILE *f;
    f = fopen("t1", "w");
    if (f == NULL) {
      exit(1);
    }
    fwrite("Testing only", sizeof(char), 12, f);
    fclose(f);
  }
[root@localhost t1]# gcc test2.c
[root@localhost t1]# ls -la
total 28
drwxrwxrwx    2 root     root         4096 Aug  1 19:46 .
drwxr-xr-x   20 root     root         4096 Aug  1 17:28 ..
-rw-r--r--    1 root     root            0 Aug  1 18:55 aaaaaat1
-rwxr-xr-x    1 root     root        11897 Aug  1 19:46 a.out
-r-sr-xr-x    1 root     root            3 Aug  1 18:41 test1
-rw-r--r--    1 root     root          196 Aug  1 19:45 test2.c
```

Suid bit-example



```
[root@localhost t1]# sudo -u test ./a.out
[root@localhost t1]# ls -la
total 32
drwxrwxrwx    2 root     root          4096 Aug  1 19:46 .
drwxr-xr-x   20 root     root          4096 Aug  1 17:28 ..
-rw-r--r--    1 root     root             0 Aug  1 18:55 aaaaaat1
-rwxr-xr-x    1 root     root         11897 Aug  1 19:46 a.out
-rw-r--r--    1 test     test            12 Aug  1 19:46 t1
-r-sr-xr-x    1 root     root             3 Aug  1 18:41 test1
-rw-r--r--    1 root     root           196 Aug  1 19:45 test2.c
[root@localhost t1]# chmod u-s a.out
[root@localhost t1]# ls -la a.out
-rwxr-xr-x    1 root     root         11897 Aug  1 19:46 a.out
[root@localhost t1]# rm -rf t1
[root@localhost t1]# sudo -u test ./a.out
[root@localhost t1]# ls -la t1
-rw-r--r--    1 test     test            12 Aug  1 19:47 t1
```