# Linux Lab 5 Files and Permissions

## Disk Management

Often, a hard disk is broken into partitions by the operating system. This has advantages:

* if a program uses too much disk space, it will fill its partition but will not affect the other partitions. If it were allowed to fill the entire disk the computer could crash;
* -it allows different rights to be assigned to different partitions. A partition that contained system binaries that don't change could be marked read-only, so malware couldn't affect it.

Partitions have the disadvantage that they carve disk space into pieces. It may be difficult to resize partitions if you find you need more space in one partition.

You can see the partitions on the disk by using the command, **df**. It shows you how much space is available on the disk file system, and where the partitions are mounted.

The hard disk is called sda. "s" stands for SCSI, "d" for disc, and "a" means it is the first drive. If you had several hard drives, they would be sda, sdb, etc. (For modern Linux, all block devices get the "s" even when they aren't SCSI. Older disks on older Linux may be hda, hdb, etc.) Partitions on the disk are labeled sda1, sda2, etc. The df command shows the logical volumes (LVM) it has created in sda2 as /dev/mapper/… (The other partitions, tmpfs, reside in a portion of RAM.) Why would you want to protect sda1 so that something doesn't use up all its space? What is mounted there?

You need to protect sda1, because it contains /boot. That is where the Linux kernel files are. Without them, your computer won’t boot.

You should see where the partitions mount on the file system tree. One starts at root /, and another starts at /boot. Which one is the very top of the file system tree?

The top of the file system tree is /.

Run the command, fdisk -l , where lower case “L” means list (you have to run it as root or sudo.) DO NOT use any other options with fdisk, as they allow you to format your disk--not good when your system is already installed. This shows disk information at a more physical level. Here you see the sda2 that df did not show you, and that it is used for the LVM which appears in df as /dev/mapper/… Read man fdisk to see what the information means.

Programs that use disk space are assigned often assigned space in either /usr or /var. Often /var is its own disk partition so that programs or logs installed there cannot overflow the disk and crash the machine.

## Exercises—hand in your commands and answers to questions

1) If there isn't an extra user on your computer, create one now. I'll call mine user1.

[john@john ~]$ sudo useradd user1

[sudo] password for john:

[john@john ~]$ sudo passwd user1

Changing password for user user1.

New password:

BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word

Retype new password:

passwd: all authentication tokens updated successfully.

[john@john ~]$

2) As root, create a directory in /var called test.

[john@john var]$ su -

Password:

Last login: Thu Oct 19 08:25:28 EDT 2017 on pts/0

[root@john ~]# cd /var

[root@john var]# ls

account cache db games kerberos local log nis preserve spool yp

adm crash empty gopher lib lock mail opt run tmp

[root@john var]# mkdir test

3) As root, create a group called test.

[root@john var]# groupadd test

4) Use ls -l on /var and look at the file rights are on /var/test. Pay special attention to:

[root@john var]# ls -l /var

total 12

drwxr-xr-x. 2 root root 19 Oct 3 11:29 account

drwxr-xr-x. 2 root root 6 Nov 5 2016 adm

drwxr-xr-x. 13 root root 159 Oct 3 11:36 cache

drwxr-xr-x. 2 root root 6 Aug 7 10:33 crash

drwxr-xr-x. 3 root root 34 Oct 3 11:30 db

drwxr-xr-x. 3 root root 18 Oct 3 11:29 empty

drwxr-xr-x. 2 root root 6 Nov 5 2016 games

drwxr-xr-x. 2 root root 6 Nov 5 2016 gopher

drwxr-xr-x. 3 root root 18 Oct 3 11:26 kerberos

drwxr-xr-x. 54 root root 4096 Oct 3 11:36 lib

drwxr-xr-x. 2 root root 6 Nov 5 2016 local

lrwxrwxrwx. 1 root root 11 Oct 3 11:25 lock -> ../run/lock

drwxr-xr-x. 17 root root 4096 Oct 20 10:01 log

lrwxrwxrwx. 1 root root 10 Oct 3 11:25 mail -> spool/mail

drwxr-xr-x. 2 root root 6 Nov 5 2016 nis

drwxr-xr-x. 2 root root 6 Nov 5 2016 opt

drwxr-xr-x. 2 root root 6 Nov 5 2016 preserve

lrwxrwxrwx. 1 root root 6 Oct 3 11:25 run -> ../run

drwxr-xr-x. 12 root root 140 Oct 3 11:29 spool

drwxr-xr-x. 2 root root 6 Oct 20 14:53 test

drwxrwxrwt. 10 root root 4096 Oct 20 14:50 tmp

drwxr-xr-x. 2 root root 6 Nov 5 2016 yp

[root@john var]#

What is the user (owner) and what permissions does it have?

User root, rwx

What is the file's group, and what permissions does the group have?

Group root, r x

What permissions does the world (other or everyone) have?

Other has r x

5) Use commands to change the file rights on /var/test so that members of the group test have read/write/x access. (hint: you will need to create the test group, use chgrp (or chown with the : option) to make test the group that owns /var/test instead of root, then chmod if the permissions need to change.)

[root@john var]# chgrp test test

--or--

[root@john var]# chown :test test

[root@john var]# ll

total 12

drwxr-xr-x. 2 root root 19 Oct 3 11:29 account

drwxr-xr-x. 2 root root 6 Nov 5 2016 adm

<snip>

drwxr-xr-x. 2 root test 6 Oct 20 14:53 test

drwxrwxrwt. 10 root root 4096 Oct 20 14:50 tmp

drwxr-xr-x. 2 root root 6 Nov 5 2016 yp

[root@john var]# chmod g+w test

[root@john var]# ll

total 12

drwxr-xr-x. 2 root root 19 Oct 3 11:29 account

drwxr-xr-x. 2 root root 6 Nov 5 2016 adm

<snip>

drwxrwxr-x. 2 root test 6 Oct 20 14:53 test

drwxrwxrwt. 10 root root 4096 Oct 20 14:50 tmp

drwxr-xr-x. 2 root root 6 Nov 5 2016 yp

6) Make user1 a member of the group test. (hint: either gpasswd, or usermod)

[root@john var]# gpasswd -a user1 test

Adding user user1 to group test

7) Use the su command to switch user to user1. Create files in the /var/test directory. Check the file rights with ls -l.

[root@john var]# exit

logout

[john@john var]$ su user1

Password:

[user1@john var]$

[user1@john var]$ touch test/file1

[user1@john var]$ echo 'a bunch of stuff' > test/otherfile

[user1@john var]$ ll test

total 4

-rw-rw-r--. 1 user1 user1 0 Oct 20 15:14 file1

-rw-rw-r--. 1 user1 user1 17 Oct 20 15:15 otherfilewh

8) Exit from su to get back to your regular user. Can your regular user delete files in /var/test? Can it read them? Why? (hint: ls -l, and the command whoami may help. Is your user the user/owner, group, or other?)

[john@john var]$ ls -l test

total 4

-rw-rw-r--. 1 user1 user1 0 Oct 20 15:14 file1

-rw-rw-r--. 1 user1 user1 17 Oct 20 15:15 otherfile

[john@john var]$ rm test/otherfile

rm: remove write-protected regular file ‘test/otherfile’? yes

rm: cannot remove ‘test/otherfile’: Permission denied

[john@john var]$ whoami

john

[john@john var]$ cat test/otherfile

a bunch of stuff

The user john only has read rights

9) Change the rights for the /var/test directory, and also the files within it, so that only the user/owner and the group have read and write access. Everyone else/other should have no access at all. (hint: chmod ### is fastest. To change all files within test, you’ll need to use the recursive flag for chmod.)

[john@john var]$ su -

Password:

Last login: Fri Oct 20 15:08:54 EDT 2017 on pts/0

[root@john ~]# chmod -R 770 /var/test

[root@john ~]# ls -l /var

total 12

drwxr-xr-x. 2 root root 19 Oct 3 11:29 account

drwxr-xr-x. 2 root root 6 Nov 5 2016 adm

<snip>

drwxrwx---. 2 root test 36 Oct 20 15:15 test

drwxrwxrwt. 10 root root 4096 Oct 20 14:50 tmp

drwxr-xr-x. 2 root root 6 Nov 5 2016 yp

[root@john ~]# ls -l /var/test

total 4

-rwxrwx---. 1 user1 user1 0 Oct 20 15:14 file1

-rwxrwx---. 1 user1 user1 17 Oct 20 15:15 otherfile

10) Test your changes by trying to read files in /var/test with your regular user. Also test with user1.

[root@john ~]# exit

logout

[john@john var]$ cat /var/test/otherfile

cat: /var/test/otherfile: Permission denied

It works, my regular user john now has no access.

[john@john var]$ su user1 -

Password:

[user1@john var]$ cat /var/test/otherfile

a bunch of stuff

Since user1 is in the test group, it still has access.