Users, Groups and Permissions, text processing commands

Users

Every user is assigned a unique User ID number (UID)
UID 0 identifies root
Users' names and UIDs are stored in /etc/ passwd
Users are assigned a home directory and a program that is run when they log in (usually a shell)
Users cannot read, write or execute each others' files without permission

Groups

Users are assigned to groups
Each group is assigned a unique Group ID number (gid)
GIDs are stored in /etc/group
Each user is given their own private group
Can be added to other groups for additional access
All users in a group can share files that belong to the group

Linux File Security

Every file is owned by a UID and a GID
Every process runs as a UID and one or more GIDs
Usually determined by who runs the process
Three access categories:

- Processes running with the same UID as the file (user)
- Processes running with the same GID as the file (group)
- All other processes (other)

Permission Precedence

If UID matches, user permissions apply
Otherwise, if GID matches, group permissions apply
If neither match, other permissions apply

Permission Types

Four symbols are used when displaying
permissions:
r: permission to read a file or list a directory's contents
w: permission to write to a file or create and remove files from a directory
x: permission to execute a program or change into a directory and do a long listing of the directory
-: no permission (in place of the r, w, or x)

Examining Permissions

File permissions may be viewed using **ls -l**

```
$ ls -l /bin/login
-rwxr-xr-x 1 root root 19080 Apr 1 18:26 /bin/login
```

File type and permissions represented by a 10 character string

Interpreting Permissions

```
-rwxr-x--- 1 hari itg 2948 Oct 11 14:07 testscript
```

- Read, Write and Execute for the owner, hari
- Read and Execute for members of the itg group
- No access for all others

Changing File Ownership

Only root can change a file's owner

Only root or the owner can change a file's group

Ownership is changed with chown:

```
chown [-R] user_name file|directory
```

Group-Ownership is changed with chgrp:

```
chgrp [-R] group_name file|directory
```

Changing Permissions – Symbolic Method

To change access modes:

chmod [-R] mode file

Where **mode** is:

- u,g or o for user, group and other
- + or - for grant or deny
- r, w or x for read, write and execute

Examples:

- ugo+r: Grant read access to all
- o-wx: Deny write and execute to others

Changing Permissions – Numeric Method

Uses a three-digit mode number

first digit specifies owner's permissions

second digit specifies group permissions

third digit represents others' permissions

Permissions are calculated by adding:

- 2 (for write)
- 4 (for read)
- 1 (for execute)

Example:

```
chmod 640 myfile
```

Changing Permissions - Nautilus

Nautilus can be used to set the permissions and group membership of files and directories.

- In a Nautilus window, right-click on a file
- Select Properties from the context menu
- Select the Permissions tab

Text Processing

head - -n
tail - -n, -f
wc - -c, -w, -l
sort - -r, -n , -f (ignore case), -u (unique), -t, -k POS1, -k POS1, POS2

eg. `sort /etc/passwd -t : -k 3 -nr`

uniq

```
cut -d: -f7 /etc/passwd | sort | uniq
```

cut

```
cut -c2-6 /usr/share/dict/words
```

paste

```
paste -d: a.txt b.txt > combined.txt
```

tr

```
cat lower.txt | tr 'a-z' 'A-Z' > upper.txt
```

diff

```
diff a.c b.c
```

grep

```
grep gecuser /etc/passwd
grep ^h.l.$ /usr/share/dict/words
grep ^a.*c$ /usr/share/dict/words
grep -v 'nologin' /etc/passwd
```

sed

```
sed -e 's/cat/dog/g' pets.txt
```

less

awk

syntax: awk pattern { action }

```
awk  '/bash/ { print }' /etc/passwd
awk '3[45]+' { print }' abc.txt
awk -F: '$3 > 500 { print $1}' /etc/passwd
```