BI296: Linux and Shell Programming

Lecture 02: Linux Command Line

Maoying,Wu

ricket.woo@gmail.com

Dept. of Bioinformatics & Biostatistics
Shanghai Jiao Tong University

Spring, 2017

Quick Recaps

- Decimal/Binary/Octal/Hexadecimal numbers and their cross-conversion
- Original digit (源码), One's complement（反码）, Two's complement（补码）
- bit, byte, kilobyte, megabyte, gigabyte, terabyte, ...
- Richard Stallman, GNU, Free and Open-source, FSF
- Linus Torvalds, Unix, Linux, Tux
- Middleware of hardware and user, utilities

Linux Commands

- Many things on a Linux system can be done by typing commands
- Note that the GUI (X-window) is not needed for running a Linux system.
- In order to type commands in X-window you need to start a terminal emulator
- Command Prompt
  - Can be configure yourself, the default is
  - $ - “logged in as a regular user”
  - # - “logged in as root (privileged user)”

Start Linux

Now start Linux and log in as administrator “root”; and then open a terminal on your desktop. See, what’s your command prompt?

Command Syntax

Syntax format

```
<command> [option(s)] [argument(s)]
```

Examples

```
# list files and directories in current directory
ls
# list files and directories lengthily
ls -l
# list files and directories in /dev
ls /dev
ls -al /dev
```

Some Basic Commands

- **System information**:
  `uname, uptime, arch, nproc, hostname, hostid`
- **User management**: `id, user, passwd, mkpasswd, useradd, userdel, groupadd, groupdel, groupmems`
- **Environment-related**: **`env`**, **`export`**
- **File system**:
  `ls, file,` **`cd`**`, mkdir, rmdir, rm, mv, cp, stat`
- **Help information**: `man, info, whatis`
- **Process control**: **`ps`**`, pstree,` **`fg`**, **`bg`**, **`jobs`**, **`nohup`**
- **File searching**: `locate, whereis, which, find`

Some Useful Commands

- `passwd`: Change your password
- `mkpasswd`: Generate a random password
- `date, cal`: Find out today's date and display a calendar
- `who, finger`: Find out who else is active on the system
- `clear`: Clear the screen
- **`echo`**: Write a message to your screen
- `write, wall, talk, mesg`: Inter-user communication
- ...

who, what, where, where to go

- `pwd`: Print Working Directory
- `uname`: Print the system information
- `whoami`: Who am I
- `cd`: Change working Directory

pwd: knowing where you are now

- **Synopsis**: `pwd`
- All the path names are started with a slash (“/”), for example
- “/root” is the home directory for root
- “/home/xxx” is the home directory for user “xxx”
- Use command “which pwd” can show you where your command locates
- Most of the user commands are stored in /bin or /usr/bin
- Commands for super user are in the directories /sbin and /usr/sbin

uname: knowing what my system is

- **Synopsis**: `uname option(s)`
- `uname -a: --all`
- `uname -i: --hardware-platform`
- `uname -m: --machine`
- `uname -n: --node-name`
- `uname -o: --operation-system`
- `uname -r: --kernel-release`
- `uname -s: --kernel-name`
- `uname -v: --kernel-version`
- `uname -p: --processor`
- You can use `uname --help` to obtain more information for this command.
- This command is very useful in compiling the system-dependent code.

whoami: knowing who you are

- **Synopsis**: `whoami`
- The command is different from “who am i”
- and also “who”
- Now guess what the command “who” can do for you?

cd: change working directory to

- **Synopsis**: `cd dir_name`
- Command “cd” without any argument will direct you to your home directory
- Command “cd ∼xxx” can direct you to the home directory of user “xxx”
- You can use either absolute pathname to visit the directory, e.g. “cd /tmp” or
- You can also use relative pathname to visit the relevant directory.
- If you are root, you can visit anywhere; but if you are a normal user, maybe you will be forbidden to visit somewhere, for example “/root”.

How to get help for a command

- "man" command
  - for example "man ls" can return the manual for the command "ls"
  - manpage is stored in the directory/usr/share/man/
- "info"
- `command --help` or `command -h`
- HOWTO Documentation
- Refer to Internet

The man command (1)

- With the man command you can get the manual page of commands
- Manual pages are stored in /usr/man or /usr/share/man
- The manual page consists of:
  - **NAME**: The name of the command and a online description
  - **SYNOPSIS**: The syntax of the command
  - **DESCRIPTION**: Explanation of how the command works and what it does
  - **Files**: The files used by the command
  - **Bugs**: Known bugs and errors
  - **See Also**: Other relevant commands

The man command (2)

- The “-k” option
  - `man -k print`
- Manual pages are divided into 8 sections:
  1. User commands
  2. System calls
  3. Libc calls
  4. Devices
  5. File formats and protocols
  6. Games
  7. Conventions, macro packages and so forth
  8. System administration
- To select the correct section, add section number:
  - `man 1 passwd`
  - `man 5 passwd`

Working with Files and Directories

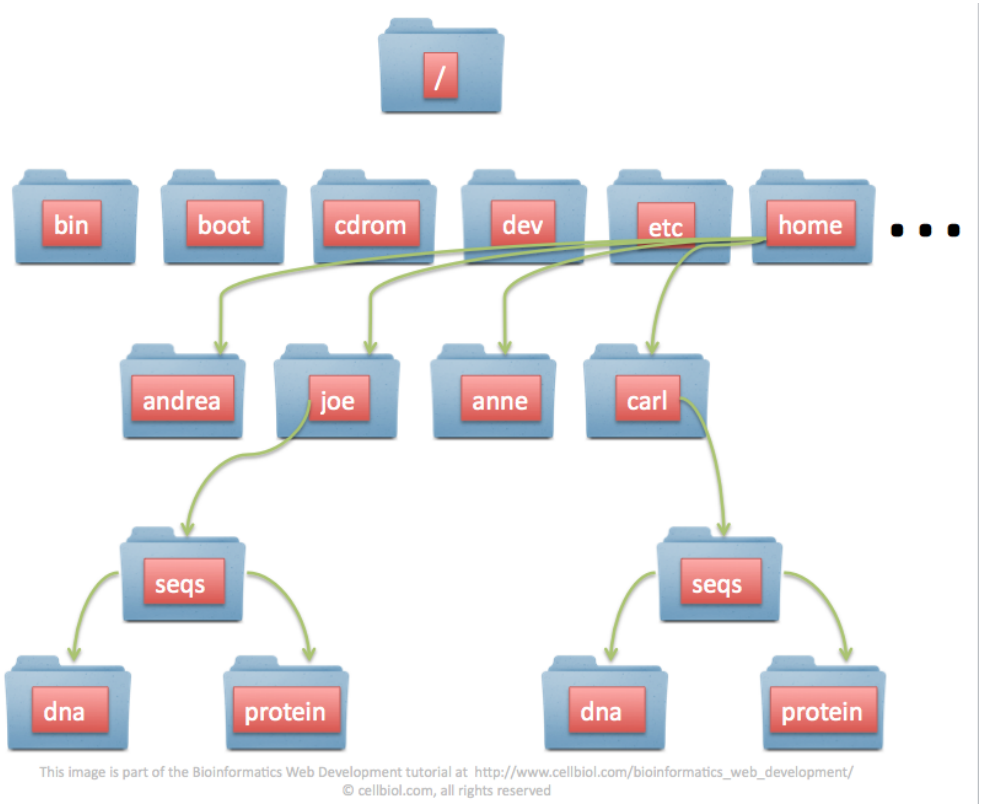
What is a file?

- A collection of data
- An object that can be read from, or written to, or both.
- *All the objects in Linux are files*
- A file has certain attributes include
  - File type
  - Access permissions

File Structure

- Generally: byte stream, record sequence, record tree
- In Linux: byte stream

Directory Structure



- Linux File System Standard: `http://www.pathname/fhs`

Print system information

Command	Description
`uname -a`	print all information
`uname -s`	print the kernel name
`uname -n`	print the network node hostname
`uname -r`	print the kernel release
`uname -m`	print the machine hardware name
`uname -p`	print the processor type or ”unknown”
`uname -i`	print the hardware platform or ”unknown”
`uname -o`	print the operating system

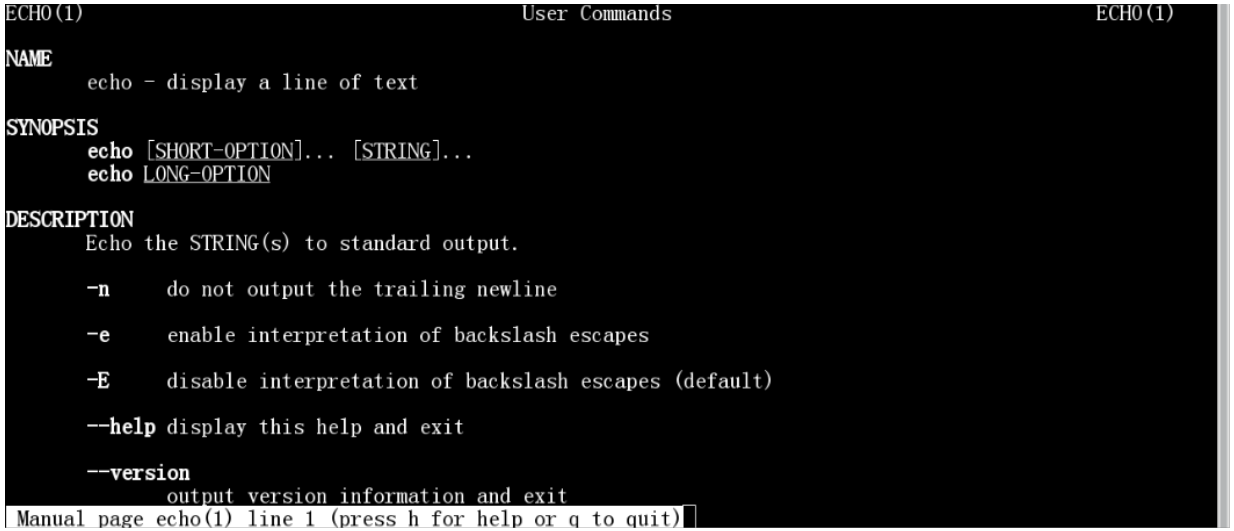
```
[bio@localhost ~]$ uname -a
Linux localhost.localdomain 3.10.0-327.el7.x86_64 #1 SMP Fri Nov 20 11:12:42
      CST 2015 x86_64 x86_64 x86_64 GNU/Linux
```

The other system-context commands

- `date`: print or set the system date and time.
- `arch`: print machine hardware name.
- `nproc`: print the number of processing units available.
- `hostname`: show or set the system's host name.
- `hostid`: print the numeric identifier for the current host.
- `uptime`: tell how long the system has been running.

Print something

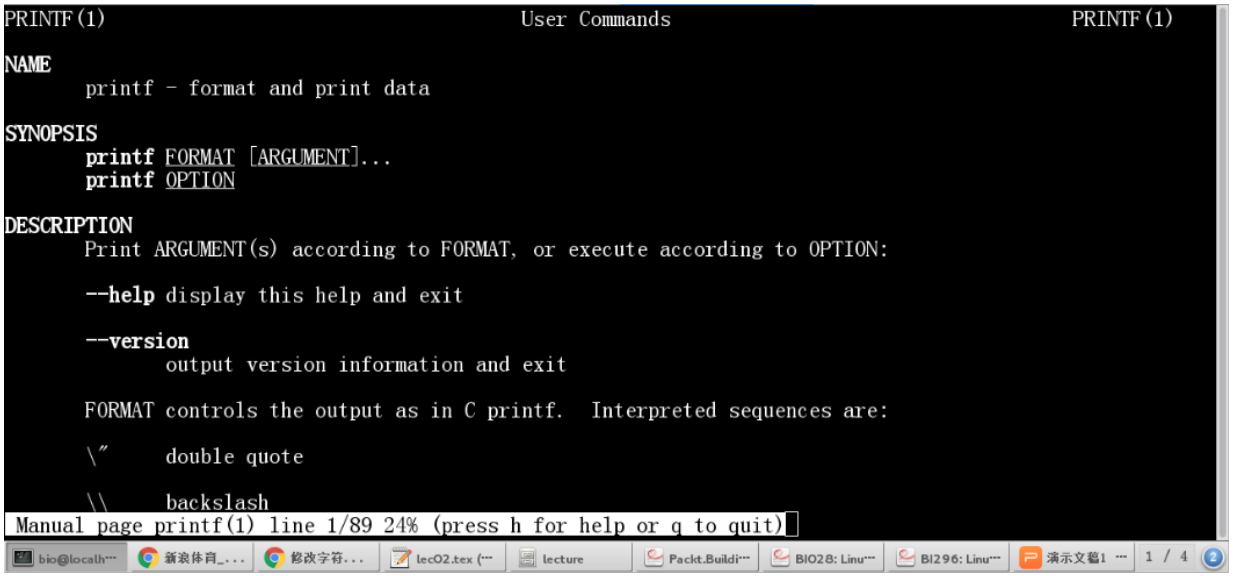
- `echo "I␣am␣genius!"#using double quotes`
- `echo 'I am genius!' #using single quotes`
- `echo I am genius! #no quotes`
- `echo $SHELL #print environment variable`
- `echo $HOSTNAME #print environment variable`
- `echo "$HOSTNAME␣is␣running␣under␣$SHELL"`



Formatting output

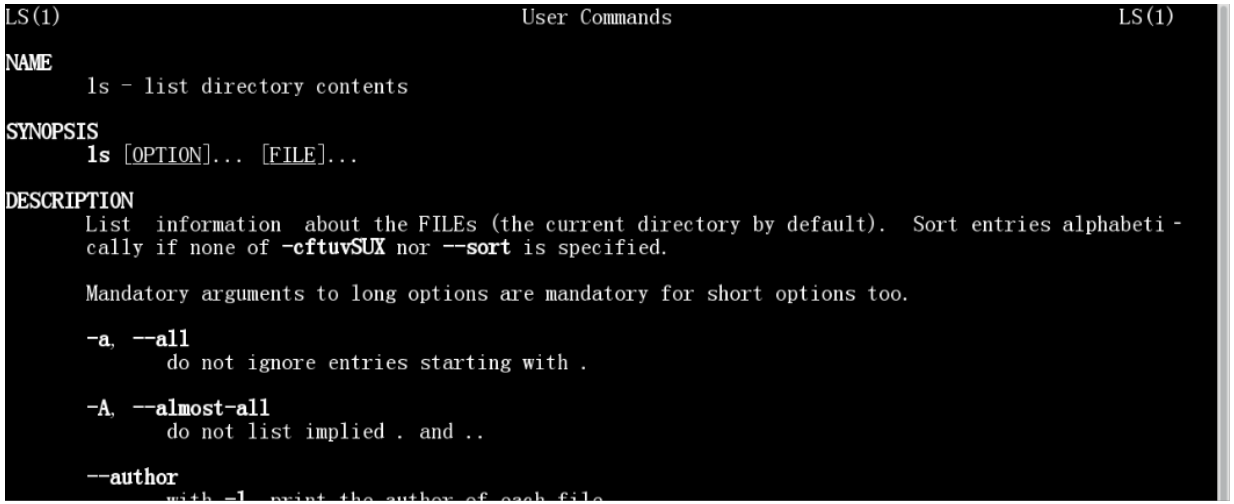
printf

```
[bio@localhost ~]$ printf "UserName\tUserID\t\tGroupID\t\tDescription\n"
UserName UserID GroupID Description
[bio@localhost ~]$ printf "The␣cost␣is␣\v%-5.2f␣dollars\n" 12.5
The cost is
           12.50 dollars
```



Getting help

Category	Example
`CMD -h,--help`	`ls -h`
`man`	`man 1 ls`
`info`	`info ls`
`whatis`	`whatis ls`



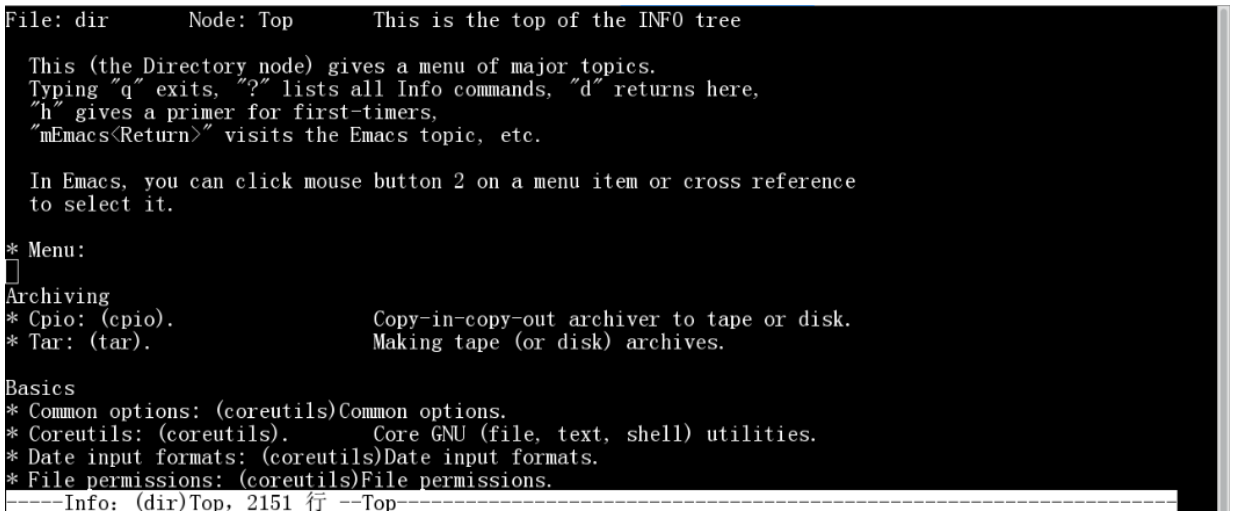
man

```
man is the system's manual pager. Each page argument given to man is
normally the name of a program, utility or function. The manual page
associated with each of these arguments is then found and displayed. A
section, if provided, will direct man to look only in that section of
the manual. The default action is to search in all of the available
sections, following a pre-defined order and to show only the first page
found, even if page exists in several sections.

The table below shows the section numbers of the manual followed by the
types of pages they contain.


1   Executable programs or shell commands
2   System calls (functions provided by the kernel)
3   Library calls (functions within program libraries)
4   Special files (usually found in /dev)
5   File formats and conventions eg /etc/passwd
6   Games
7   Miscellaneous (including macro packages and conventions), e.g.
    man(7), groff(7)
8   System administration commands (usually only for root)
```

info


File: dir Node: Top This is the top of the INFO tree
This (the Directory node) gives a menu of major topics.
Typing "q" exits, "?" lists all Info commands, "d" returns here,
"h" gives a primer for first-timers,
"mEmacs<Return>" visits the Emacs topic, etc.
In Emacs, you can click mouse button 2 on a menu item or cross reference
to select it.
* Menu:
Archiving
* Cpio: (cpio). Copy-in-copy-out archiver to tape or disk.
* Tar: (tar). Making tape (or disk) archives.
Basics
* Common options: (coreutils)Common options.
* Coreutils: (coreutils). Core GNU (file, text, shell) utilities.
* Date input formats: (coreutils)Date input formats.
* File permissions: (coreutils)File permissions.
-----Info: (dir)Top, 2151 行 --Top-----

File types

- `ls -l /dev/sda1`
- `file /dev/sda1`

File type	Description	Example
Regular file (-)	Plain data file	/etc/passwd, /bin/bash
Directory (d)	A directory entry	/usr
Block special file (b)	Block device	/dev/sda1
Character special file (c)	Character device	/dev/null
FIFO (p)	Named pipe	/run/systemd/initctl/fifo
Socket (s)	/tmp/mysqld.sock	
Symbolic link (l)	Symbolic link	/usr/lib64/libc.so.6

```
[bio@localhost ~]$ file /usr/lib64/libc-2.17.so
/usr/lib64/libc-2.17.so: ELF 64-bit LSB shared object, x86-64, version 1 (GNU
     /Linux), dynamically linked (uses shared libs), BuildID[sha1]=53
     c0918c85fa9cc08d2b57e76467631ab07554ae, for GNU/Linux 2.6.32, not
     stripped
```

File permissions

- Four access levels
  - user(u): The user that owns the file
  - group(g): The group that owns the file
  - others(o): The others except the owner and group
  - all(a): all the users
- Three permissions
  - Read(r): Read content of a file or list content of a directory
  - Write(w): Change content of a file or create/delete files in a directory
  - Execute(x): Execute files as a program or use directory as active directory
- Special file permission
  - Setuid, setgid(s): special permissions for executable files. Any user who runs that executable file assumes the user ID of the owner/group of the file.
  - Sticky-bit(t): special permission for public directories.

```
[bio@localhost lecture]$ ll /usr/bin/passwd
-rwsr-xr-x. 1 root root 27832 5月   3 2014 /usr/bin/passwd
[bio@localhost lecture]$ ll /usr/bin/locate
-rwx--s--x. 1 root slocate 40496 5月   3 2014 /usr/bin/locate
[bio@localhost lecture]$ ls -ld /tmp
drwxrwxrwt. 12 root root 4096 3月   2 15:13 /tmp
[bio@localhost lecture]$
```

Directory permissions

Symbol	Description
`r`	List files in the directory.
`w`	Add or remove files/directories/links in the directory.
`x`	Open/execute files in the directory, change to the directory and subdirectories.

Change the permission of a file

Command	Description
`ls -l filename`	list the properties of a file
`chown bio:bio filename`	change the owner/group of a file
`chgrp bio filename`	change the group of a file
`chmod 755 filename`	change the mode/permission of a file
`chmod u+w filename`	add write permission for the owner
`chattr -A filename`	change the attribute of a file
`lsattr filename`	list the attributes of a file

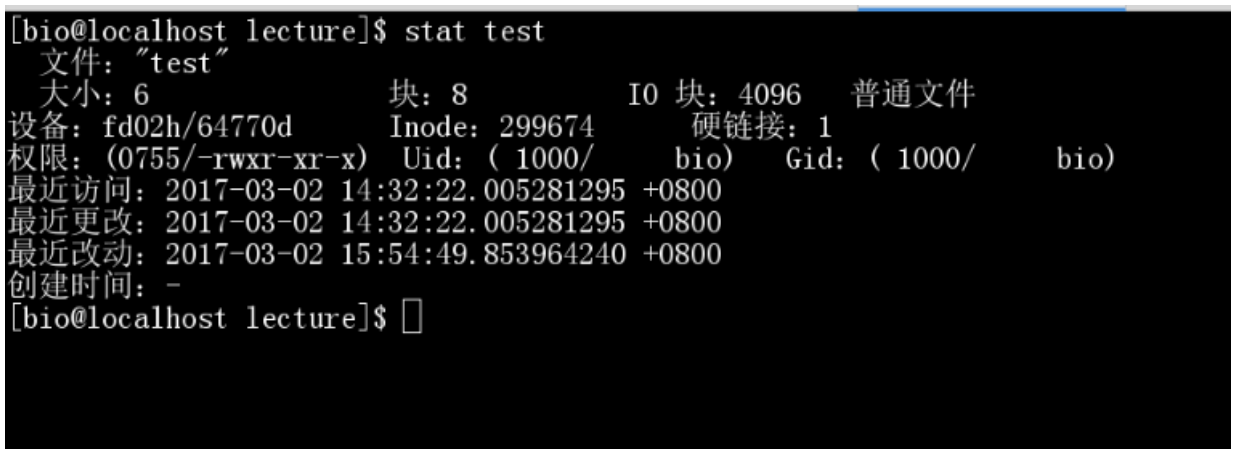
```
[bio@localhost lecture]$ ll test
-rw-rw-r-- 1 bio bio 6 3月   2 14:32 test
[bio@localhost lecture]$ chmod u+x test
[bio@localhost lecture]$ ll test
-rwxrw-r-- 1 bio bio 6 3月   2 14:32 test
[bio@localhost lecture]$ chmod 755 test
[bio@localhost lecture]$ ll test
-rwxr-xr-x 1 bio bio 6 3月   2 14:32 test
```

umask

- By default, the system sets the permission on a text file to 666, and to 777 on a directory or executable file.
- The value assigned by the **umask** command is subtracted from the default.
- **umask** `022` command denies write permission for group and others.
- Usually the `root` and a normal user has different **umask** strategies. Why?

Display file status

```
stat filename
```



- `mtime`: last time the file content was modified, `ls -l`
- `ctime`: last time of file status modification, `ls -lc`
- `atime`: last access time, `ls -lu`
- `crtime`: creation time, `stat --printf '%n: created %w\n'`

Display file contents

- Display entire file
  - `cat, tac /etc/passwd`
  - `less, more /etc/passwd`
  - `nl /etc/passwd`
  - `od /etc/passwd`
  - `base64 /etc/passwd`
- Display parts of file
  - `head /etc/passwd`
  - `tail /etc/passwd`
  - `split -d -n 2 /etc/passwd passwd.`
- Formatting file contents
  - `fmt`
  - `pr -o 5 /etc/passwd`
  - `fold -w 20 /etc/passwd`

The other commands relating to file and directory

- Directory listing: `ls, dir`
- Basic operations: `cp, dd, mv, rm`
- Directory operations: `mkdir, rmdir, cp, mv`
- Changing file attributes: `chgrp, chmod, chown, touch`
- Summarizing files: `wc, sum, md5sum`
- Sorted files: `sort, uniq, comm`
- Operating on fields: `cut, paste, join`
- Operating on characters: `tr, expand`
- Creating link file: `ln`

`ln`: creating link file

- `ln a b`: create hard link `b` for `a`
- `ln -s a b`: create symbolic link `b` for `a`
- Can you create a hard link for a directory? symbolic link?
- Can you create a hard link between different operating systems? symbolic link?

Exercise (1)

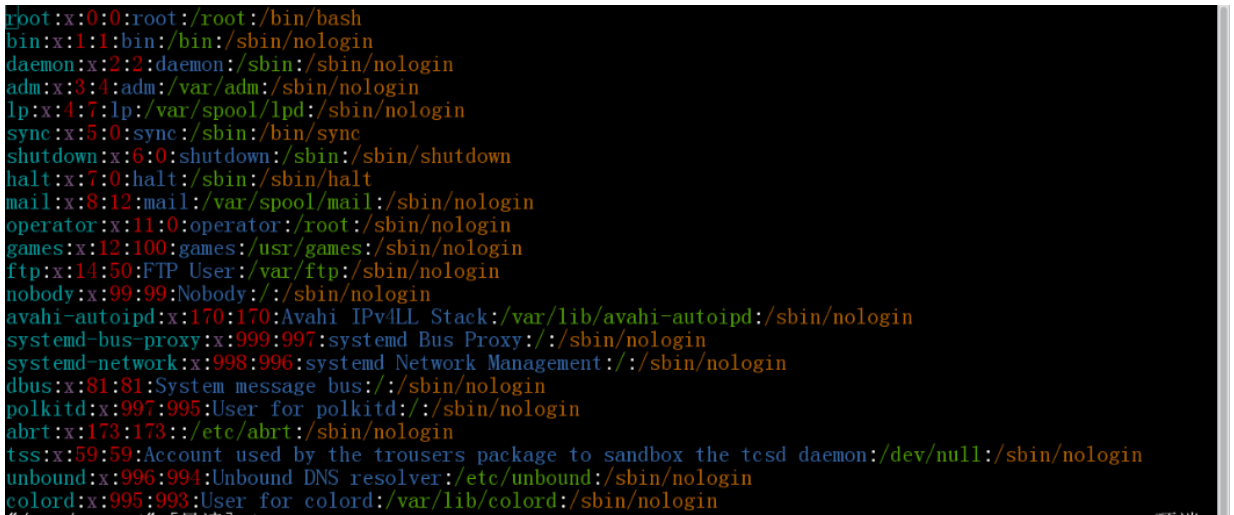
1. What is the difference between `man 1 printf` and `man 3 printf`?
2. We have known that the command `tree` can print out the directory tree. How to set the output level to be 2?
3. Compare `ls -a` and `ls -A`. Can you find what this option "-a" is able to do?
4. What is the difference among commands `cat`, `less` and `more` to view the content of a regular file?
5. What is the usage of the command `umask`? List the possible distinctions between dealing with files and with directories?
6. Tell the difference between hard link and symbolic link? If I would like to use `ln` to create a hard link file2 for file1, what should I do? If we delete file1, what will happen to file2? How about symbolic link?

Exercise (2)

1. What is the difference between `whoami` and `who am i`?
2. What does `chmod 735 file1` do? If we want to prevent a folder `new` from being accessed by all the user except the owner, what should we do?
3. How to create a multiple-level directory `/tmp/a/b/c/d` using `mkdir` in exactly one line command?
4. Tell what the following commands can do for you.
   - `chmod u+w file2`
   - `chmod a-x file2`
   - `chmod 644 file2`
   - `chmod o=rwx file2`

Users

- Each user has a unique username and userID
- A user can belong to more than one group
- All the users are stored in the file `/etc/passwd`
- The encrypted password for each user is stored in the file `/etc/shadow`
- The user can be added using `useradd` command, and modified by `usermod`, and deleted by `userdel`
- For each user, the private environment is defined by the file `/.bashrc` and the common environment is defined in the file `/etc/bashrc` and `/etc/profile`
- The default settings for `useradd` is defined in the file `/etc/default/useradd`.



Output of user information

- `id`: print real and effective user and group IDs.
- `logname`: print user's login name.
- `whoami`: print effective user id.
- `groups`: print the groups the user is in
- `users`: print the user names of users currently logged in to the current host.
- `who`: show who is logged in.
- `w`: Show who is logged on and what they are doing.

Exercise

Have a look at the manual page of the command `groupmems` and the configuration file `/etc/sudoers`.

1. Log in to the system as the root;
2. Enable the group `wheel` has the `sudo` privilege;
3. Recruit the user 'bio' into the group `wheel`;
4. Use command `id` to confirm whether the above job work;
5. Try run `sudo su - root` to check whether the `sudo` privilege work for user `bio`.

Process control (1)

Command	Description
`CTRL+z`	Stop (interrupt) a foreground process.
`CTRL+c`	Kill a foreground job.
`ps`	list the running processes.
`pstree`	print the process tree.
`fg`	run the suspended job in foreground.
`bg`	Run the suspended job in background.
`jobs`	List the background or suspended jobs.
`CMD &`	place the job in the background.
`nohup CMD &`	run in the background.
`kill`	Kill a running process or suspended jobs.

Process control (2)

1. Type in the command `yes I am genius`
2. Press `CTRL+z` to suspend the running process
3. Use **`jobs`** to look through the suspended or background processes
4. Use **`fg`** `%job-id` to bring the process to foreground, or
5. Use **`bg`** `%job-id` to run the job in background, or
6. use **`kill`** `%job-id` to kill the process

Standard streams

- Standard Input (stdin: 0)
  - Generally it is a keyboard.
- Standard Output (stdout: 1)
  - Generally it is the terminal or the printer.
- Standard Error (stderr: 2)
  - Generally it is the terminal.

Piping

- The STDOUT of a command acts as the STDIN of the next command.
- **Examples**
  - `cat /etc/paswd | wc -l`
    count the number of lines in the file.
  - `cat /etc/passwd | grep mysql`
    see if the user exists.
  - `ls -l /dev | less`
    view the content as stream
  - `gunzip file.tar.gz | tar xvf -`
    uncompress the .tar.gz file.

Redirection

- STDIN Redirection
  - `cat 0< /etc/passwd`
  - `cat < /etc/passwd`
- STDOUT Redirection
  - `ls -l > filelist.log`
    redirect the output to the file
  - `ls -l 1> filelist.log`
    redirect the stdout to the file
  - `ls -l >> filelist.log`
    append the stdout to the file
- STDERR Redirection
  - `ls -z 2> err.log`
    redirect the stderr to the file
  - `ls -z 1> run.log 2>&1`
    redirect both the stdout and stderr to the file
  - `ls -z 2>/dev/null`
    suppress the output of the stderr

Disk usage

- `fdisk -l`
  manipulate disk partition table.
- `df -h`
  report file system disk space usage.

```
Filesystem                      Size  Used Avail Use% Mounted on
/dev/sda2                        50G   35G   16G  69% /
/dev/sda3                       177G  106G   72G  60% /home
/dev/sda1                       497M  158M  340M  32% /boot
```

- `du -sh /home/bio`
  estimate file space usage.

```
102G    /home/bio
```

- `stat -f /`
  display system status.

```
  File: "/"
    ID: fd0000000000 Namelen: 255     Type: xfs
Block size: 4096       Fundamental block size: 4096
Blocks: Total: 13100800   Free: 4125851    Available: 4125851
Inodes: Total: 52428800   Free: 51628882
```

Exercises

The command `ls -i` can list the inode number of the requested file.

1. Use `echo` and redirection to create a new file;
2. Use `cp` to create a new copy of the above file;
3. Use `mv` to create a new file of the above file;
4. Use `ln` to create a hard link of the above file;
5. Use `ln -s` to create a symbolic link of the above file;
6. Use `ls -i` to check whether the files have the same index node.
7. Tell the difference among `cp, mv, ln, ln -s`.

Bring-home Exercises

1. A project requires that a set of users should share a directory so that all of them can create/modify/remove the files/subdirectories beneath the directory.

   For the sake of security, the directory cannot be accessed by the other users.

   Can you figure out a solution? You should provide both your idea and a practical example.

2. The command `man hier` will output the directory tree of the system as well as what they will host. Write a short essay to illustrate the directory tree of the Linux system.

3. `vim` is a very popular text editor under Linux system. Please have a look at the tutorial we provided in the course webpage.