1 Unix History

- 1965: Researchers from Bell Labs and other organizations begin work on Multics, a state-of-the-art interactive, multi-user operating system.
- 1969: Bell Labs researchers, losing hope for the viability of Multics due to performance issues, withdraw from the project.

  One of the researchers, Ken Thompson, finds a little-used PDP-7, and in a month implements a simple operating system comprising a kernel, a command interpreter, an editor, and an assembler.

  Other Bell researchers, most notably Dennis Ritchie, are attracted to Thompson's system and contribute to it.
- 1970: Peter Neumann suggests the name "Unics" for Thompson's operating system, a pun on "Multics". A DEC PDP-11 is acquired for further development of UNIX.
- 1971: Work begins on the C programming language.
- 1973: UNIX is rewritten in C.
- 1991: Linus Torvalds released Linux kernal, the core of the Linux operating system that was completely compliant with the original UNIX.
- 1993: There were 12000 Linux users.
- Nowadays, Linux keeps getting bigger and better.

2 Commands

2.1 Basic Commands

Command	**Meaning**
ls	Displays a list of files in the current working directory
cd *directory*	change directories
passwd	change the password for the current user
file *filename*	display file type of file with name filename
cat *textfile*	throws content of *textfile* on the screen
pwd	display present working directory
exit or **logout**	leave this session
man *command*	read man pages on ***command***
info *command*	read info pages on ***command***

2.2 Getting Help

- The **man** pages
  - Browse to the next page using the space bar.
  - Go back to the previous page using the b-key.
  - When you reach the end, man will usually quit and you get the prompt back.
  - Type q if you want to leave the man page before reaching the end, or if the viewer does not quit automatically at the end of the page.

  - Some commands have multiple man pages. For instance, the passwd command has a man page in section 1 and another in section 5. By default, the man page with the lowest number is shown. If you want to see another section than the default, specify it after the man command:
    man 5 passwd
  - If you want to see all man pages about a command, one after the other, use the -a to man:
    man -a passwd

    This way, when you reach the end of the first man page and press SPACE again, the man page from the next section will be displayed.

- More info: you can read the Info pages about a command, using the **info** command.
  - Use the arrow keys to browse through the text and move the cursor on a line starting with an asterisk, containing the keyword about which you want info, then hit **Enter**.
  - Use the **P** and **N** keys to go to the previous or next subject.
  - The space bar will move you one page further, no matter whether this starts a new subject or an Info page for another command. Use **Q** to quit.
- The **whatis** command: a short index of explanations for commands.
- The **--help** option: gives a short explanation about how to use the command and a list of available options. Eg., **man --help**.

3 Linux File System

"On a UNIX system, everything is a file; if something is not a file, it is a process." A directory is just a file containing names of other files. Programs, services, texts, images, and so forth, are all files. Input and output devices, and generally all devices, are considered to be files, according to the system.

3.1 Sorts of Files

- *Regular files (-)*: contain normal data, for example text files, executable files or programs.
- *Directories (d)*: files that are lists of other files.
- *Special files (c)*: the mechanism used for input and output. Most special files are in /dev.
- *Links (l)*: a system to make a file or directory visible in multiple parts of the system's file tree.
- *Sockets (s)*: a special file type, similar to TCP/IP sockets, providing inter-process networking protected by the file system's access control.
- *Named pipes (p)*: act more or less like sockets and form a way for processes to communicate with each other, without using network socket semantics.

The -l option to **ls** displays the file type, using the first character of each input line:

```
polaris:~ jtao$ ls -l
total 0
drwx----w-  13 jtao  faculty   442 Jan 27 18:39 Desktop
```

```
drwx----w-  20 jtao  faculty   680 Jan 23 19:28 Documents
drwx----w-  89 jtao  faculty  3026 Jan 28 13:06 Downloads
drwx------@ 18 jtao  faculty   612 Jan 23 19:28 Dropbox
drwx----w-@ 39 jtao  faculty  1326 Oct 26 18:14 Library
drwx----w-   3 jtao  faculty   102 Oct 26 17:39 Movies
drwx----w-   5 jtao  faculty   170 Oct 26 17:39 Music
drwx----w-   6 jtao  faculty   204 Oct 26 17:39 Pictures
drwxr-xrwx   5 jtao  faculty   170 Oct 26 17:39 Public
```

3.2 The File System

In a file system, a file is represented by an *inode*, a kind of serial number containing information about the actual data that makes up the file: to whom this file belongs, and where is it located on the hard disk.

At the time a new file is created, it gets a free inode, containing the following information:

- File type (regular, directory, ...).
- Owner and group owner of the file.
- Permissions on the file
- Date and time of creation, last read and change.
- Date and time this information has been changed in the inode.
- Number of links to this file
- File size
- An address defining the actual location of the file data.

3.2.1 Orientation in the File System

When you type a command, you almost never have to give the full path to that command. For example, we know that the ls command is in the /bin directory:

```
polaris:~ jtao$ which -a ls
/bin/ls
```

yet we don't have to enter the command /bin/ls for the computer to list the content of the current directory. This is taken care of by the PATH environment variable. To display the content ("$") of the variable PATH, we use **echo** command:

```
polaris:~ jtao$ echo $PATH
/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin:/usr/local/munki:/usr/texbin
```

A path is the way you need to follow in the tree structure to reach a given file. A path that starts with a slash (/) is called an *absolute path* since it starts from the root directory. The Unix file system groups all files together in a hierarchical tree structure. The top of the hierarchy is traditionally called the root. A path that starts with ∼ is in the user's home directory. Paths that don't start with a slash are always *relative*. In relative paths we also use the . and .. indications for the *current* and the *parent directory.*

3.2.2 The Most Important Files and Directories

- The **kernel** is the heart of Unix, it allocates time and memory, handles file operations and system calls.
- The **shell** acts as an interface between the user and the kernel. The shell is what you are typing to at the Unix prompt after you log in. There are different types of shells:
  - **sh** or Bourne Shell: the original shell still used on UNIX systems and in UNIX related environments.
  - **bash** or Bourne Again SHell: the standard GNU shell, intuitive and flexible. On Linux, **bash** is the standard shell for common users. This shell is a so-called *superset* of the Bourne shell, a set of add-ons and plug-ins. This means that the Bourne Again SHell is compatible with the Bourne shell: commands that work in sh, also work in bash. However, the reverse is not always the case.
  - **csh** or C Shell: the syntax of this shell resembles that of the C programming language.
  - **tcsh** or Turbo C Shell: a superset of the common C Shell, enhancing user-friendliness and speed.
  - **ksh** or the Korn shell: sometimes appreciated by people with a UNIX background. A superset of the Bourne shell; with standard configuration a nightmare for beginning users.
  - **zsh** or the Z shell: a Unix shell that can be used as an interactive login shell and as a powerful command interpreter for shell scripting. Zsh can be thought of as an extended Bourne shell with a large number of improvements, including some features of bash, ksh, and tcsh.

  The file /etc/shells gives an overview of known shells on a Linux system:

  ```
  polaris:~ jtao$ cat /etc/shells
  /bin/bash
  /bin/csh
  /bin/ksh
  /bin/sh
  /bin/tcsh
  /bin/zsh
  ```

  To switch from one shell to another, just enter the name of the new shell in the active terminal.

  ```
  polaris:~ jtao$ tcsh
  [polaris:~] jtao%
  ```

  Which shell am I using?

  ```
  [polaris:~] jtao% echo $SHELL
  /bin/bash
  ```

- Your home directory is stored in the HOME environment variable and indicated by a tilde (∼). To display the content of this variable:

  ```
  polaris:~ jtao$ echo $HOME
  /home/jtao
  ```

3.2.3 Manipulating Files

- **ls**: displays all the files in the directory.

  -l displays additional information for each file

  -a displays all files, files that begin with "." are not displayed automatically

  -F displays "/" after directories, and "*" after executables

  -t sort by timestamp instead of alphabetically

- **mkdir**: creates a directory. Use command "mkdir CS246" to create a directory to store your homework files and stuff; create subdirectories for each of the homeworks and projects.

- **cd**: change directory.

  cd CS246: switch to the directory CS246. All file names that you enter are relative to the current working directory.

  cd ..

  cd ∼

- **pwd**: displays the current working directory.

- **mv**: moves or renames files.

```
polaris:CS246 jtao$ ls
test.txt
polaris:CS246 jtao$ ls -l test.txt
-rw-r--r--  1 jtao  faculty  5 Jan 29 18:44 test.txt
polaris:CS246 jtao$ mv test.txt hi.txt
polaris:CS246 jtao$ ls
hi.txt
polaris:CS246 jtao$ ls -l hi.txt
-rw-r--r--  1 jtao  faculty  5 Jan 29 18:44 hi.txt
```

- **cp**: copies files and directories. A useful option is recursive copy (copy all underlying files and subdirectories), using the **-R** option to **cp**. The general syntax is **cp [-R] fromfile tofile** .

- **rm**: removes single files.

- **rmdir**: removes empty directories.

3.3 File Security

3.3.1 Access Right

On a Linux system, every file is owned by a user and a group user. There is also a third category of users, who are not the user owner and don't belong to the group owning the file. For each category of users, read, write(change/move/delete) and execute permissions can be granted or denied. Access to files is controlled by several things:

- The user's id (in /etc/passwd that keeps user account and password information. This file holds the majority of information about accounts on the Unix system.)

- The users group membership (/etc/passwd and /etc/group)
- File ownership
- File permission bits

```
Jias-MacBook-Pro:CS246 jiatao$ ls -l
total 1024
drwxr-xr-x  22 jiatao  staff     748 Jan 29 23:31 Exercise
```

Above command shows that Exercise is owned by the user *jiatao* and the group *staff* and it is a directory (d). The next nine characters are three triples specifying permissions for user (owner), group, and "other".

user (owner)	**group**	other
rwx	r-x	r-x
read, write, execute	read, execute	read, execute

3.3.2 File Permissions

The **chmod** command is used to change the mode (permissions) of files. One way to specify the mode is to use octal (base 8)each digit corresponds to an rwx triple:

```
Jias-MacBook-Pro:Exercise jiatao$ ls -l x
-rwxr-xr-x   1 jiatao  staff  8496 Jan 20 15:24 x
Jias-MacBook-Pro:Exercise jiatao$ chmod 742 x
Jias-MacBook-Pro:Exercise jiatao$ ls -l x
-rwxr---w-  1 jiatao  staff  8496 Jan 20 15:24 x
```

Note that r/w/x is used to improve readability. rwxr-xr-x could be expressed as 111101101 instead. For example, the interpretation of 742 is:

7	=	111	rwx (user)
4	=	100	r– (group)
2	=	010	-w- (other)

Note that the leading dash is the file type.

```
Jias-MacBook-Pro:Exercise jiatao$ ls -l test
-r--r--r--  1 jiatao  staff  8496 Jan 20 15:24 test
Jias-MacBook-Pro:Exercise jiatao$ chmod ug+w test
Jias-MacBook-Pro:Exercise jiatao$ ls -l test
-rw-rw-r--  1 jiatao  staff  8496 Jan 20 15:24 test
Jias-MacBook-Pro:Exercise jiatao$ chmod +x test
Jias-MacBook-Pro:Exercise jiatao$ ls -l test
-rwxrwxr-x  1 jiatao  staff  8496 Jan 20 15:24 test
```

Note that "chmod o+rw test" changes the mode for OTHERS, not owner.

Many files may be specified at once:

```
chmod +x dat1 *.sh test.*
```

The -R option recursively applies a change to every file (and directory) in a tree:

```
chmod -R -w src
```

In addition to adjustments like g+rw and o-rwx, permissions can be set directly with a symbolic specification:

```
chmod u=rw,g=wx,o=r
```

3.3.3 Directory Permissions

Permissions apply to directories, too, but with a different interpretation:

- Read permission allows the directory entries to be listed.
- Write permission allows entries to be added or deleted.
- Execute permission allows a directory to be traversed and/or cd'd into.

4 I/O Redirection

Most Linux commands read input, such as a file or another attribute for the command, and write output. By default, input is being given with the keyboard, and output is displayed on your screen. The keyboard is the *standard input* (stdin) device, and the screen or a particular terminal window is the *standard output* (stdout) device.

4.1 Output redirection with > and |

If you want to put output of a command in a file, or issue another command on the output of one command, you need to *redirect output.* Redirection is done using either the ">" (greater-than symbol), or using the "|" (pipe) operator which sends the standard output of one command to another command as standard input.

- The **cat** command concatenates files and puts them all together to the standard output. By redirecting this output to a file, this file name will be created or overwritten if it already exists.

  ```
  Jias-MacBook-Pro:Exercise jiatao$ cat test1.txt
  Hihi, this is test1.
  Jias-MacBook-Pro:Exercise jiatao$ cat test2.txt
  Hihi, this is test2.
  Jias-MacBook-Pro:Exercise jiatao$ cat test3.txt
  Hi, this is test3.
  Jias-MacBook-Pro:Exercise jiatao$ cat test1.txt test2.txt > test3.txt
  Jias-MacBook-Pro:Exercise jiatao$ cat test3.txt
  Hihi, this is test1.
  Hihi, this is test2.
  ```

  When redirecting output, be careful not to overwrite existing files!

- Redirecting "nothing" to an existing file will empty the file.

- Redirection to an nonexistent file will create a new empty file with the given name.
- To find a word within some text, display all lines matching "pattern1", and exclude lines also matching "pattern2" from being displayed:

  grep *pattern1* file | grep -v *pattern2*
- To display output of a directory listing one page at a time:

  ls -la | less
- To find a file in a directory:

  ls -l | grep *part_of_file_name*
- You can save output of any command to a file. For example,

```
Jias-MacBook-Pro:~ jiatao$ ls -l | less > less_file
Jias-MacBook-Pro:~ jiatao$ cat less_file
total 472
drwx------+ 16 jiatao  staff     544 Jan 28 13:28 Desktop
drwx------+ 20 jiatao  staff     680 Jan 16 06:54 Documents
drwx------+ 27 jiatao  staff     918 Feb  1 21:49 Downloads
drwx------@ 19 jiatao  staff     646 Jan 31 13:29 Dropbox
drwx------@ 20 jiatao  staff     680 Sep 18 21:19 Dropbox_jtao510
drwx------@ 55 jiatao  staff    1870 Sep 11 12:23 Library
drwx------+  5 jiatao  staff     170 May 11  2013 Movies
drwx------+  4 jiatao  staff     136 Nov 10  2012 Music
drwx------+ 17 jiatao  staff     578 Aug  1  2013 Pictures
drwxr-xr-x+  4 jiatao  staff     136 Nov  8  2012 Public
-rw-r--r--   1 jiatao  staff  239424 Jan 28 15:53 hs_err_pid45616.log
-rw-r--r--   1 jiatao  staff       0 Feb  2 00:42 less_file
```

4.2 The >> Operator

Instead of overwriting file data, you can also append text to an existing file using two subsequent greater-than signs:

```
Jias-MacBook-Pro:~ jiatao$ date
Sun Feb  2 00:47:00 EST 2014
Jias-MacBook-Pro:~ jiatao$ date >> less_file
Jias-MacBook-Pro:~ jiatao$ cat less_file
total 472
drwx------+ 16 jiatao  staff     544 Jan 28 13:28 Desktop
drwx------+ 20 jiatao  staff     680 Jan 16 06:54 Documents
drwx------+ 27 jiatao  staff     918 Feb  1 21:49 Downloads
drwx------@ 19 jiatao  staff     646 Jan 31 13:29 Dropbox
drwx------@ 20 jiatao  staff     680 Sep 18 21:19 Dropbox_jtao510
drwx------@ 55 jiatao  staff    1870 Sep 11 12:23 Library
drwx------+  5 jiatao  staff     170 May 11  2013 Movies
drwx------+  4 jiatao  staff     136 Nov 10  2012 Music
drwx------+ 17 jiatao  staff     578 Aug  1  2013 Pictures
```

```
drwxr-xr-x+  4 jiatao  staff     136 Nov  8  2012 Public
-rw-r--r--   1 jiatao  staff  239424 Jan 28 15:53 hs_err_pid45616.log
-rw-r--r--   1 jiatao  staff       0 Feb  2 00:42 less_file
Sun Feb  2 00:45:58 EST 2014
```

The **date** command would normally put the last line on the screen; now it is appended to the file less_file.