

Linux File System

IT 4423

Unix/Linux Administration

J.G. Zheng
Spring 2012



Overview

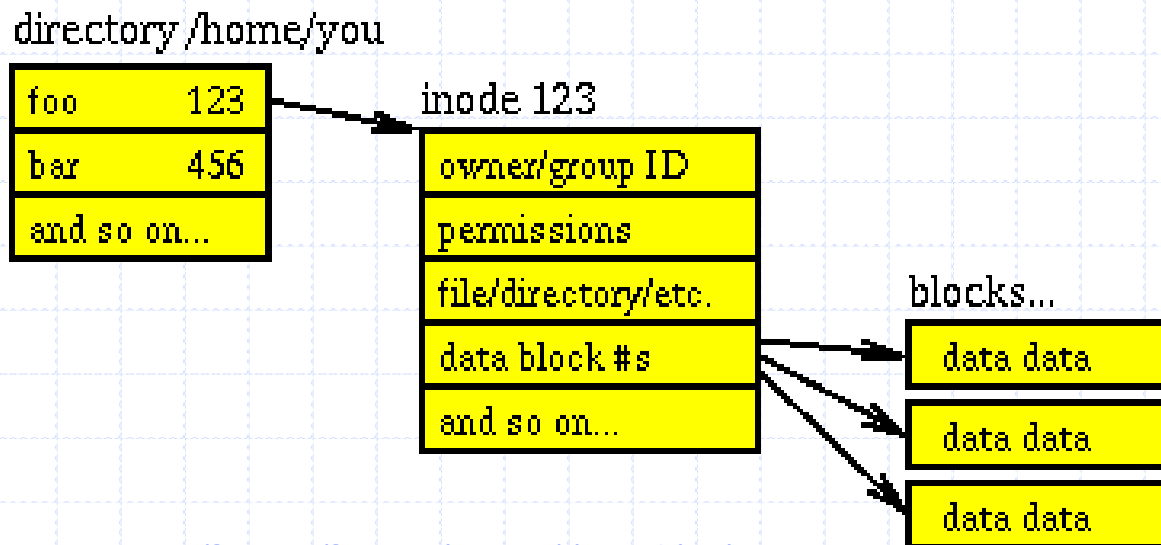
- ◆ Linux file system components
- ◆ Directory hierarchy
- ◆ File types
- ◆ File permissions
- ◆ Common file operation commands

File System

- ◆ A file system is a set of data structures that represent and organize the system's storage resources
- ◆ Common types of disk file systems
 - NTFS: used for Windows NT, and more recent Windows
 - FAT, FAT32: used for DOS, earlier Windows, and many removable storage devices
 - ext2, ext3, ext4: used for Linux
 - ISO 9660, UDF: used for optical disks

Linux File System Structure

- ◆ Basic components of a Linux file system
 - inode block: containing file metadata including
 - ◆ File types, timestamp, permissions, etc.
 - ◆ Pointers to data (content) blocks
 - Data block: actual file content



Inode

- ◆ The inode block contains file metadata
 - Inode number: a unique number assigned to a file
 - File type
 - Timestamp: access, content change, metadata change
 - File size
 - Number of links
 - File permissions
 - Owner and group id
 - Data block pointers
 - Other extended information
 - Note there is no file name
- ◆ Use "ls" or "stat" command to view most of these data

File Types

◆ The Unix way

- Every object is treated as a file and mapped to the file system, including directories, devices, etc.

◆ File types

- **Regular file:** files that contain content
- **Directory:** a holder for other files
- **Symbolic link:** a pointer to other files
- Character device file
- Block device file
- Local domain socket
- Named pipe

◆ Use “file” command and “ls” command to find out file types

Directory Hierarchy

- ◆ Directory is a special file that contains entries of other files/directories
 - For each entry, the file name and inode are recorded
 - . and .. are always in a directory
- ◆ The complete structure of directories is a tree structure
- ◆ Special directory
 - Home directory
 - Working (current) directory .
 - Parent directory ..

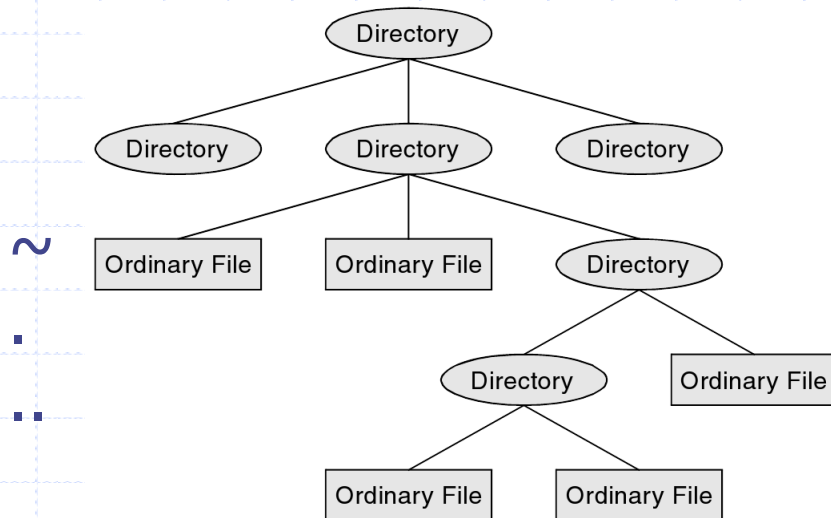


Figure 6-3 Directories and ordinary files

Path

- ◆ Path defines the complete reference name to a file
 - Unique for each file

Top level, or root

Different levels in the path are separate by "/"

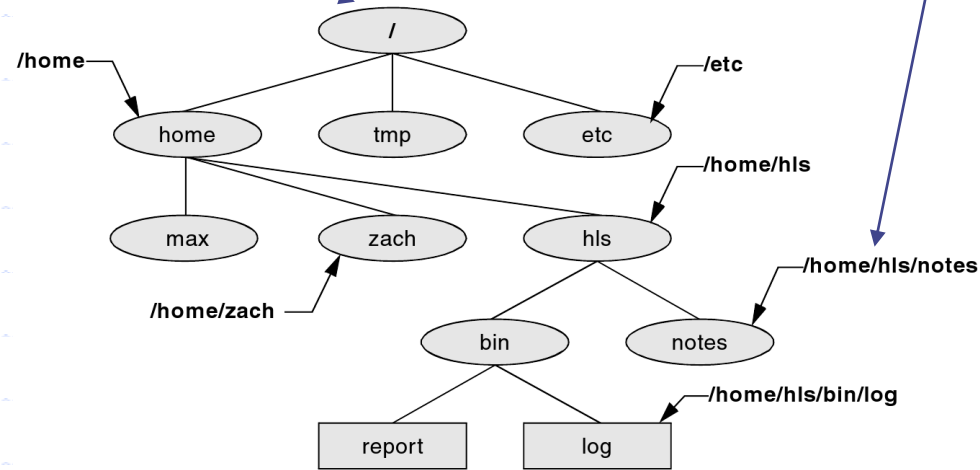


Figure 6-5 Absolute pathnames

Method

- Absolute: starting from the root "/"
- Relative: starting from the working directory

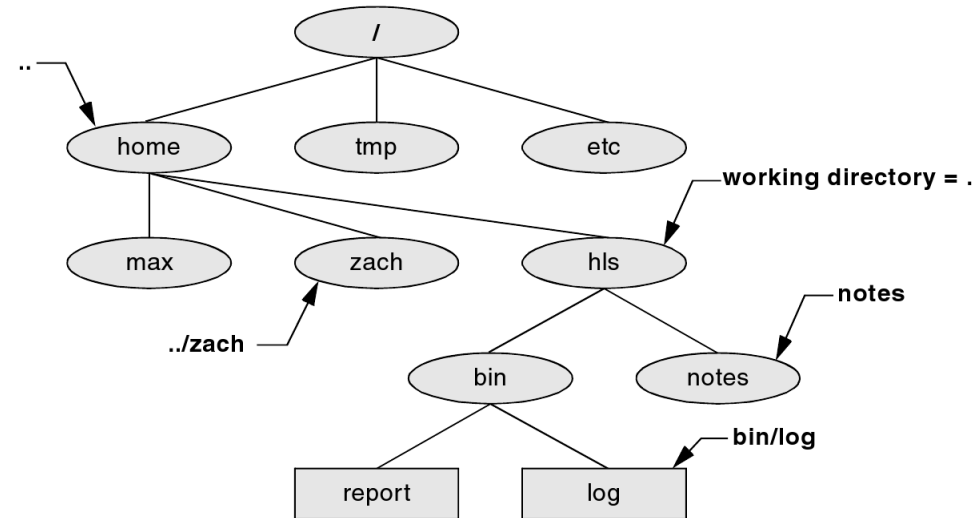


Figure 6-6 Relative pathnames

Important Directories

- ◆ Filesystem Hierarchy Standard (FHS) defines the main directories and their contents in Linux file systems
 - http://en.wikipedia.org/wiki/Filesystem_Hierarchy_Standard
- ◆ Major directories
 - /bin essential command binaries
 - /boot boot loader files
 - /dev devices
 - /etc system configuration files
 - /home user's place
 - /media mount points for removable storage
 - /mnt temporarily mounted file system
 - /root for the root (admin) user
 - /sbin essential system binaries
 - /usr user's read-only data

File Names

- ◆ File names are not recorded in inodes, but in the content block of directory files
- ◆ Naming rules
 - Can contain any characters other than "/"
 - Length: 255 characters
 - Case sensitive
 - No extension is required
 - No same file name in the same directory
 - Hidden files: file name starting with "."

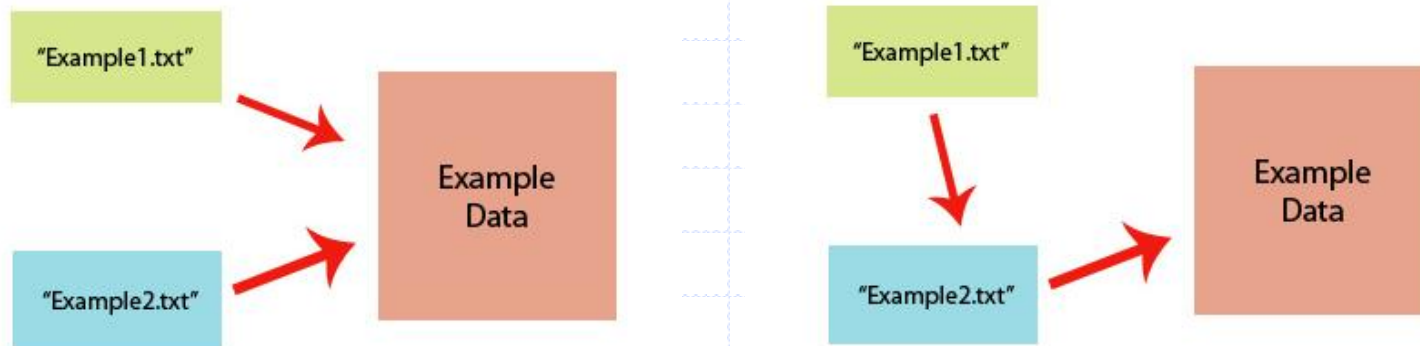
Special Characters in File Names

- ◆ File name with special characters like *, -, (blank)
- ◆ Use ' ' around these names

Link

◆ Links are used to point to files from different places.

- Hard link vs. Symbolic (soft) link



- Hard link is created to have the same inode as the target file
- Soft link is created to have a different inode
- Hard links don't rely on one another; while a soft link is dependent on another file name and location.

◆ Use the "ln" command to create links

Basic File Permissions

- ◆ Traditional UNIX does not allow permissions to be set per user
 - See later sessions for the use of the Access Control List

- ◆ Basic rules

- Each file has 9 permission bits
- These 9 bits are divided into 3 sets (in order): Owner, Group, Others
- Each permission set has 3 permissions (in order): r (read), w (write), x (execute)
- For each permission bit, "-" means denied

- Example:
rwxr-x---

The owner has all permissions, while the group does not have the write permission; all other users do not have any permission.

- Each user fits into only one of the three permission sets. The permissions used are those that are most specific.
- The "root" user has read and write permissions for all files, regardless how their permissions are set.
- Permissions are not inheritable

Permission Effects

Symbol	File	Directory	Symbolic Link
R	View content	Read file names only in the directory	Permissions are determined by its linked target file.
W	Change content	Create, delete, move, rename files if "x" is granted	
X	Execute binary or script files	Can be accessed, entered or passed through (cd), can access its files (inode information)	

◆ Common directory settings

- r-x: allows the content of the directory to be listed.
- -wx: allows files to be created, deleted, and renamed within the directory.

Permissions Representation

Octal Value	Binary Format	Symbolic Value	Permission
0	000	---	none
1	001	--x	execute only
2	010	-w-	write only
3	011	-wx	write and execute
4	100	r--	read only
5	101	r-x	read and execute
6	110	rw-	read and write
7	111	rwX	all granted

◆ Octal values can also be used

- $rwXrwXr-x = 775$
- $rw-rw-rw- = 666$

Table 6-2 Examples of numeric permission specifications

Mode	Meaning
777	Owner, group, and others can read, write, and execute file
755	Owner can read, write, and execute file; group and others can read and execute file
711	Owner can read, write, and execute file; group and others can execute file
644	Owner can read and write file; group and others can read file
640	Owner can read and write file, group can read file, and others cannot access file

Default Permissions

- ◆ Default permissions are set when a new file or directory is created
 - In Unix all base permissions begin as the following;
Directories [Octal 777 / Binary 11111111]
Files [Octal 666 / Binary 110110110]
- ◆ Permission mask
 - Defines the permissions denied by default
 - For example: 022 (the default mask) means 2 (write permission) is restricted for group and others.
 - Use the "umask" command to view and set mask values.

Use "umask -S" to display symbolic output of the mask effect

```
root@ubuntu0: ~  
File Edit View Search Terminal Help  
root@ubuntu0:~# umask  
0022  
root@ubuntu0:~# umask -S  
u=rwx,g=rx,o=rx  
root@ubuntu0:~# touch newfile  
root@ubuntu0:~# ls newfile -l  
-rw-r--r-- 1 root root 0 2011-09-15 03:53 newfile  
root@ubuntu0:~# umask 777  
root@ubuntu0:~# touch new2  
root@ubuntu0:~# ls -l new2  
----- 1 root root 0 2011-09-15 04:03 new2  
root@ubuntu0:~# █
```

Other File Attributes

◆ Timestamp

- Access time
- Modify time
- Change time

◆ File owner: user, group

◆ Number of links

- For normal files, this shows the number of hard links
- For directories, this shows the number of sub-directories (always including "." and "..")

Directory/File Commands Summary

◆ Navigation

- cd, pwd

◆ View directory/file content and properties

- ls
- file, stat

◆ Modification (creation, change, deletion)

- mkdir, cp, mv, rm, rmdir, ln

◆ Permissions

- chmod, umask

ls

◆ Basic usage

- List files in the current directory: `ls`
- List a specified file/directory: `ls [file/directory name]`
- Use file name expansion: `ls /etc/*.conf`

◆ Common options

- `-l`: long listing (see next slide)
- `-i`: view inode number
- `-h`: human readable format for file size
- `-F`: show file type sign
- `-a`: show all files (including hidden files)
- `-d`: display directories as ordinary files, not its content
- `-R`: Includes the contents of all subdirectories.
- `-1`: 1 item per line
- Sorting: `-S`: size, `-t`: time, `-r`: reverse order

◆ Reference

- <http://www.computerhope.com/unix/uls.htm>

Pathname Expansion

- ◆ `*`, `?` and `[]` can be used for matching file names for commands that take multiple file names, applied to
 - `ls`, `cp`, `rm`, `mv`, `file`, etc.

◆ Examples

```
ls *.conf
```

```
ls .*
```

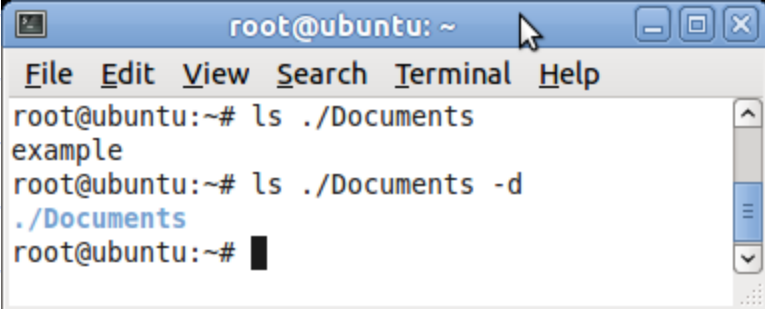
```
rm file?
```

```
cp ../file[0-9]
```

ls and Directories

◆ -d option

- If an argument is a directory it only lists its name not its contents.



```
root@ubuntu: ~  
File Edit View Search Terminal Help  
root@ubuntu:~# ls ./Documents  
example  
root@ubuntu:~# ls ./Documents -d  
./Documents  
root@ubuntu:~#
```

◆ -R option

- Display the contents of all levels of subdirectories.

More ls Examples

◆ Use . * / in ls argument

- . current directory, or start of a hidden file
- * wildcard file name expansion
- / directory

◆ Combination examples

- ls ./* list all files in the current directory and sub-directories
- ls /* list all hidden files in the root directory, and files in the hidden directory under the root directory
- ls .*/ list all hidden sub-directories

ls -l Output

◆ Long listing: ls -l

File type	Symbol	Type of file	File access permissions	ACL flag	Links	Owner	Group	Size	Date and time of modification	Filename
Regular file	-		-rwxrwxr-x+		3	max	pubs	2048	2010-08-12 13:15	memo

Figure 6-12 The columns displayed by the `ls -l` command

◆ File type by file colors

- Black: normal file
- Blue: directory
- Green: executable file
- Cyan: link

More File Information

◆ Common file colors

- Black: normal file
- Blue: directory
- Green: executable file
- Cyan: link

```
drwxr-xr-x 19 root root 4096 2011-08-27 23:51 lib
drwx----- 2 root root 16384 2011-08-16 06:04 lost+found
drwxr-xr-x 3 root root 4096 2011-09-05 05:28 media
drwxr-xr-x 3 root root 4096 2011-09-05 05:26 mnt
```

◆ Long listing: 'ls -l'

Type of file	File access permissions	ACL flag	Links	Owner	Group	Size	Date and time of modification	Filename
	<u>-rwxrwxr-x+</u>		3	max	pubs	2048	<u>2010-08-12 13:15</u>	memo

Figure 6-12 The columns displayed by the `ls -l` command

View File Metadata

◆ file

- This command describes the type of file(s).

file file1

file *

◆ stat

- This command display detailed information of a file

stat file1

Copy/Move Files and Directories

◆ cp

- Copy files/directories
- cp [source] [destination] (Note: default is to overwrite)
 - ◆ cp ../file1 file2
 - ◆ cp ../file1 .
- -i prompt for overwrite choices
- -r copy all files from a directory and its sub-directories

◆ mv

- Move (rename) files/directories
- mv [source] [destination] (Note: default is to overwrite)
 - ◆ mv ../file1 /root/Documents
 - ◆ mv file1 file2 (rename if in the same directory)
 - ◆ mv files1 ../file2 (move and rename)
- -i prompt for overwrite choices

Create and Delete Directory

◆ mkdir

- Create directory

◆ rmdir

- Delete directory

Removing Directory and File

◆ rm

- Deletes a file without confirmation (by default).

◆ Options

- **-i:** interactive. Prompted for confirmation of deletion
- **-r:** recursive. Delete all file and sub-directories.

In

◆ Create a hard link

- In [target file] [link name]
- In Documents/file1 alink

◆ Create a soft link -s

- In -s Documents/file1 alink

chmod

◆ Syntax

- chmod [permission setting] files

◆ Permission settings

- + add permission
- - remove permission
- = directly set permission
- a all
- u user (owner)
- g group
- o others

◆ Examples

- u+w
 - ◆ Adds write permission for the owner of the file
- ug=rw,o=r
 - ◆ Gives r/w permission to owner and group, and read permission to others
- a-x
 - ◆ Removes execute permission for all categories (owner/group/other)
- g=u
 - ◆ Makes the group permissions be the same as the owner permissions

Summary

◆ Key concepts

- File system
- inode
- File, directory, working directory
- Path, absolute path, relative path
- Link, symbolic link, hard link
- File permission symbols, octal number meanings

◆ Key practices

- Use the following commands for common file system operations
 - ◆ Files and directories: `ls, cd, pwd, mkdir, cp, mv, rm, rmdir, ln`
 - ◆ View file attributes: `file, stat, (ls)`
 - ◆ Permission: `chmod, umask`

Good Resrouces

◆ Unix LS Command: 15 Practical Examples

- <http://www.thegeekstuff.com/2009/07/linux-ls-command-examples/>

◆ File permissions

- <http://www.tuxfiles.org/linuxhelp/filepermissions.html>

◆ Linux Directory Permission Confusion

- <http://www.hackinglinuxexposed.com/articles/20030424.htm>

◆ Fixing Unix/Linux/POSIX Filenames

- <http://www.dwheeler.com/essays/fixing-unix-linux-filenames.html>