TNE30019/TNE80014 – Unix for Telecommunications

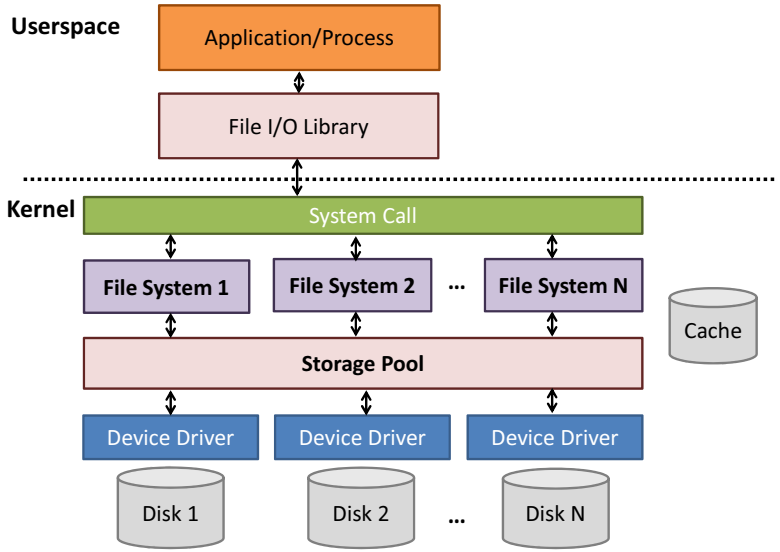
The Unix Kernel – File Systems & Permissions

Dr. Jason But

Swinburne University

Outline

Block Devices – File Systems



File System Formats

FAT32

Windows 95/98 or earlier, Windows floppy disks, USB sticks
Driver provided with most UNIX systems

NTFS

Windows XP, Vista, 7, 8
Read-only driver provided with most UNIX systems

ext2/ext3/ext4

Linux Extended File System 2/3/4

ZFS

Current standard BSD File System

Many others

Accessible on Unix if file-system driver is available

Unix Global File System

- MS-DOS / Windows world uses multiple disks
  - Each disk is named and independent (e.g. **c:**, **d:**)
- In Unix there is only one virtual disk
- Different disks are "mounted" and form directory tree in global or "root" file system (root is /)

Example:

- Floppy disk drive is mounted to `/mnt/floppy`
- All files in `/mnt/floppy` are on floppy disk – copying file to this directory copies it to floppy disk
- Floopy disk can be mounted to any directory (not just `/mnt/floppy`)
- Un-mounting `/mnt/floppy` forces OS to write cached data to disk and allow you to eject it

Unix File System Structure

`/bin`
System binaries

`/boot`
Kernel and boot files

`/dev`
Device driver interfaces

`/etc`
System and service configuration files

`/home`
User home directories

Unix File System Structure

`/lib`
System libraries

`/mnt`
Standard place for mount points

`/proc`
System process information

`/root`
Root (super-user) home directory

`/sbin`
System super-user binaries
Only executable by root users

Unix File System Structure

`/usr` – Not system stuff

`/usr/bin`	Regular user binaries
`/usr/lib`	Regular user libraries
`/usr/local`	Non-standard stuff (machine specific)
`/usr/sbin`	Super-user binaries
`/usr/share`	Common shared files
`/usr/src`	System source code

`/var` – Run time information storage

`/var/cron`	Cron job information
`/var/log`	System log files
`/var/run`	Runtime information (process ID files)

Unix File System Structure

/usr/local – Non-standard stuff (machine specific)

- `/usr/local/bin` Non-standard binaries
- `/usr/local/lib` Non-standard user libraries
- `/usr/local/etc` Non-standard configuration files
- `/usr/local/sbin` Non-standard super-user binaries
- `/usr/local/share` Non-standard shared files
- `/usr/local/src` Non-standard source code

Unix File System – Partitions

Partition

- Part of physical/logical disk
- Each partition can have its own file system
- Special partition for swap

Traditionally often multiple partitions

- For example, separate partitions for `/`, `/home` and `/var`
- File system corruption contained to partition
- Less chance of full file system problem (logging to `/var`)
- Easier upgrading, e.g. `/home` can be left untouched

Single partition

- Easier to setup
- More efficient disk use

Unix File System – Files

File Names

- Maximum length 255 bytes (commonly)
- No limit on path name length (commonly)
- As many periods as you like
- No extensions required to signify executable

File Owners

- One owner – tied to single numeric user id (e.g. **root** = 0)
- One group – tied to numeric group id (e.g. **wheel** = 0)

Links

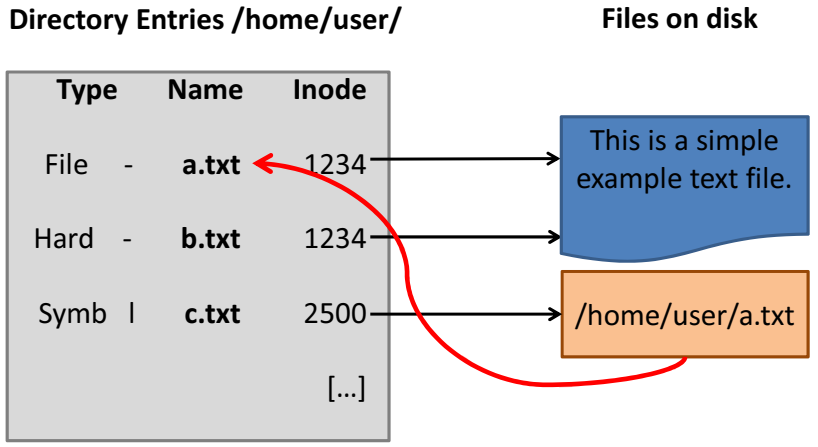
Symbolic link Like a Windows shortcut (aka soft link)
File contains path to another file/directory it links to

Hard link File pointing to another file's location on disk

Unix File System – Symbolic vs. Hard Link

Create link

```
ln [-s] <target_name> <link_name>
```



Unix File System – Navigating the Tree

Show files in current directory

```
ls
```

Show current working directory

```
pwd
```

Change directory

```
cd <path>
```

Two types of paths

- **Absolute**: Specific directory/file in tree
- **Relative**: Directory/file relative to working directory

Multi-User Concept

- Concept of many users
  - Some tasks can **only** be performed by certain users (privileges)
  - Some files can **only** be accessed by certain users
- User groups allow managing of privileges for multiple users
- Security through permissions
  - Regular users cannot access memory or processes of other users
  - Regular users can only access devices or files if permitted
  - Regular users cannot change ownership/permissions of devices or other user's files
  - Only system administrator (aka **root**) can access all processes, files, unmount file systems, change all permissions, etc.

Multi-User – Changing Users

You can log out and log in as different user... cumbersome

su (substitute user)

```
su <user_name>
```

sudo (substitute user to do something)

- Allow subsets of users to execute commands as another user (usually root)
- File `/etc/sudoers` lists users and commands they can execute
- Huge amount of configurability compared to setuid/setgid
- When is it useful?
  - Give users administrative rights without need to be root
  - Allow users to execute few privileged applications

Unix File System – Permissions

- Each file has permission set for three sets of users

User	What owner can do with file
Group	What users in owner group can do
Others	What everybody else can do

- Allowable permission bits

Read	Specified user(s) can access file
Write	Specified user(s) can modify file
Execute	File is executable by specified user(s)

- Special permissions bits for executable files

setuid	File is executed as if owner executed it
setgid	File is executed as if group member executed it
sticky	If set only owner of directory can rename and remove files

Unix File System – Permission-related Commands

Show permissions

```
ls -l
```

Show groups

```
groups or id
```

Change permissions

```
chmod <permissions> <files>
```

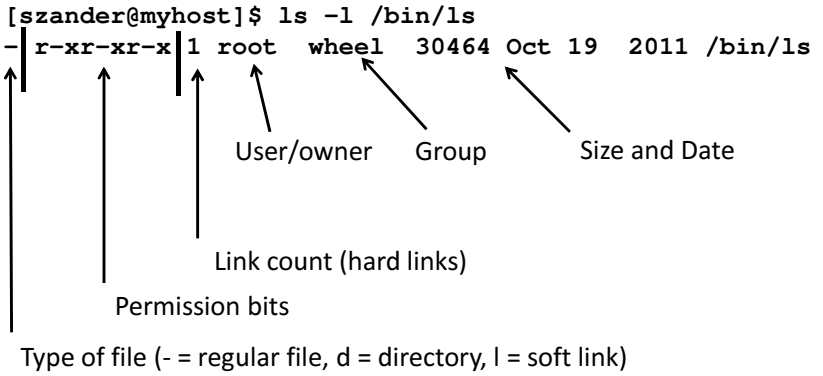
Change owner

```
chown <owner> <files>
```

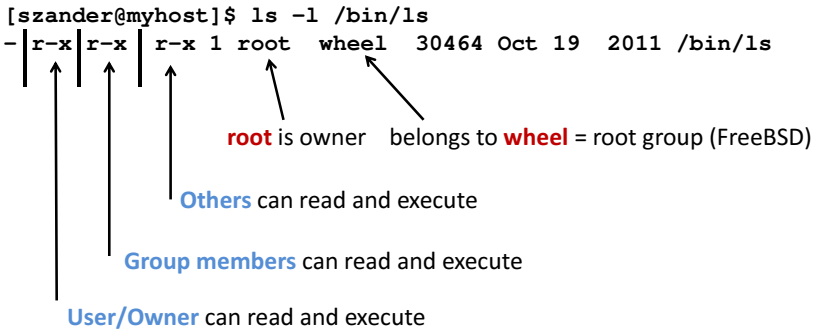
Change group

```
chgrp <group> <files>
```


Unix File System – File Permissions Example



Unix File System – File Permissions Example



Unix File System – File Permissions Example