System Programming

UNIX File Systems

UNIX File System

- The file system is your interface to:
  - physical storage (disks) on your machine
  - storage on other machines (NFS)
  - input/output devices
  - etc.
- *Everything* in Unix is a file (programs, text files, peripheral devices, terminals, …)
- Directory is a file to contain (references to) other files
- There are no drive letters in Unix! The file system provides a *logical* view of the storage devices

Working Directory

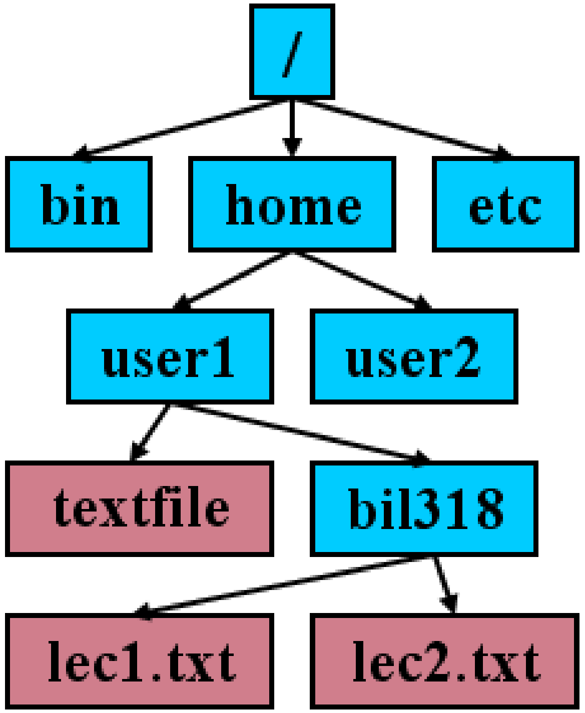
- Working directory: your current position of the file system
- `pwd` (print working directory) command outputs the absolute path (more later) of your working directory
- Unless you specify another directory, a command will assume that you want to operate within the working directory

Home Directory

- Home directory: personel user space
- At login, your working directory will be set to your home directory
- The path (more later) to your home directory can be referred to by the ~ (tilde) symbol
- The home directory of user1 can be referred to by `~user1`

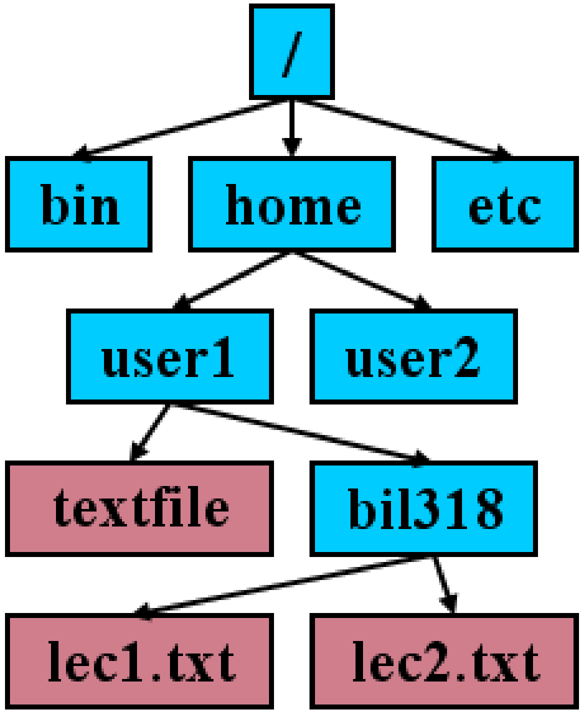
Unix File Hierarchy

- *Root directory*: /
- Directories may contain plain files or other directories
- Result is a tree structure for the file system
- Unix does not recognize any special filename extensions



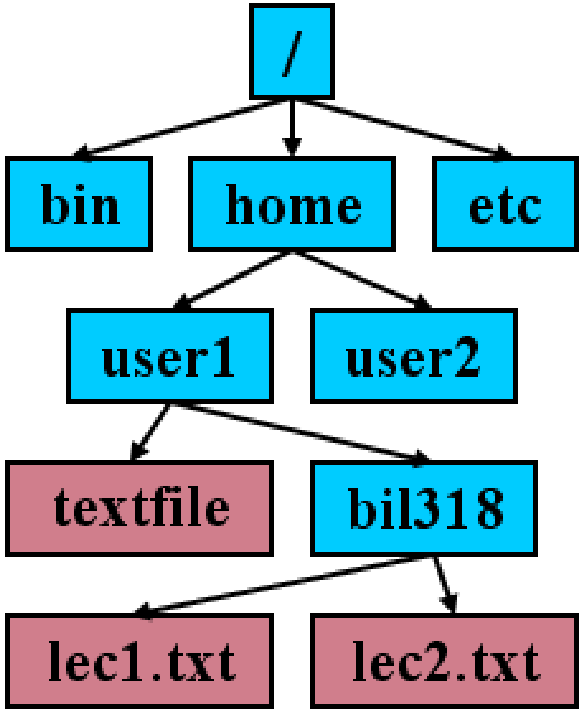
Unix Paths

- Separate directories by /
- Absolute Path
  - start at the root and follow the tree
- Examples:
  - /home/user1/textfile
  - ~user1/textfile
  - ~/textfile



Unix Paths (cont.)

- Relative Path
  - start at working directory
  - .. – level above
  - . – working directory
- Examples:
  - textfile
  - bil318/lec1.txt



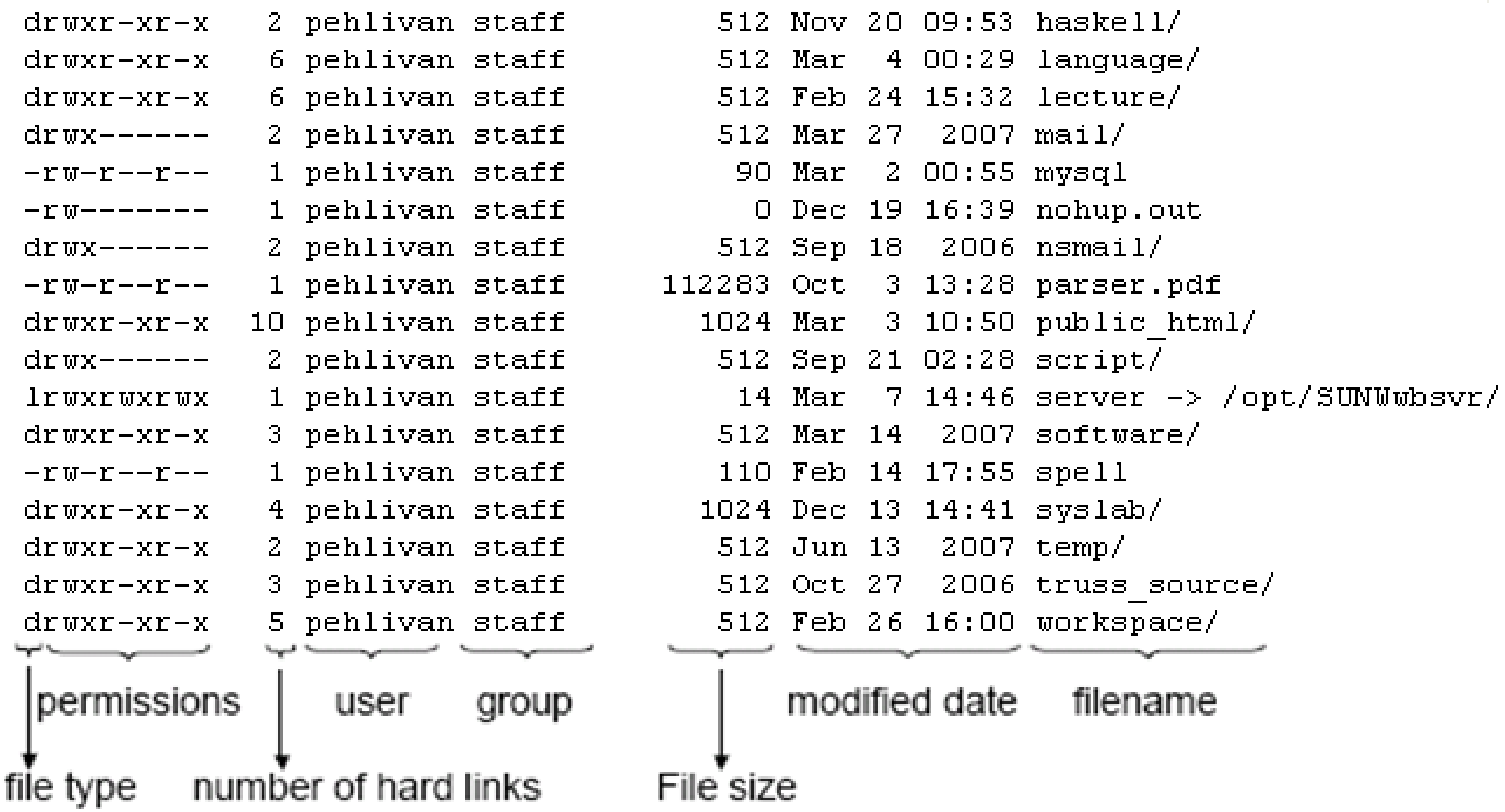
Some Standard Directories

- / – root directory
- /bin – standard commands and utilities; executables
- /dev – block and character device directory
- /etc – host-specific configuration; host services
- /home – users’ home directories
- /lib – library directory for various languages
- /sbin – system commands and utilities (needed to boot)
- /tmp – temporary files
- /usr – user utilities and applications; /usr/local/
- /var – system files that vary (logs, spools, email)

Changing Directories

- cd – changes the working directory
  - cd <directory path>
  - can use absolute or relative path names
  - cd without any arguments is the same: `cd ~`
  - `Examples:`
    - `cd /home/user1`
    - `cd ../../user1`

File Information (ls –al)



```
drwxr-xr-x   2 pehlivan staff        512 Nov 20 09:53 haskell/
drwxr-xr-x   6 pehlivan staff        512 Mar  4 00:29 language/
drwxr-xr-x   6 pehlivan staff        512 Feb 24 15:32 lecture/
drwx------   2 pehlivan staff        512 Mar 27  2007 mail/
-rw-r--r--   1 pehlivan staff         90 Mar  2 00:55 mysql
-rw-------   1 pehlivan staff          0 Dec 19 16:39 nohup.out
drwx------   2 pehlivan staff        512 Sep 18  2006 nsmail/
-rw-r--r--   1 pehlivan staff     112283 Oct  3 13:28 parser.pdf
drwxr-xr-x  10 pehlivan staff       1024 Mar  3 10:50 public_html/
drwx------   2 pehlivan staff        512 Sep 21 02:28 script/
lrwxrwxrwx   1 pehlivan staff         14 Mar  7 14:46 server -> /opt/SUNWwbsvr/
drwxr-xr-x   3 pehlivan staff        512 Mar 14  2007 software/
-rw-r--r--   1 pehlivan staff        110 Feb 14 17:55 spell
drwxr-xr-x   4 pehlivan staff       1024 Dec 13 14:41 syslab/
drwxr-xr-x   2 pehlivan staff        512 Jun 13  2007 temp/
drwxr-xr-x   3 pehlivan staff        512 Oct 27  2006 truss_source/
drwxr-xr-x   5 pehlivan staff        512 Feb 26 16:00 workspace/
```

Types of Files

- Plain (–): most files, binary or text
- Directory (d): points to a set of files
- Symbolic link (l): pointer to another file or directory
- Special files
  - Character device (c): keyboard, printer, joystick
  - Block device (b): disk, CD-ROM
- Communication files
  - FIFO (p): a temporary storage device (queue)
  - Socket (s): an endpoint for communication

File List

- ls –F command shows what a file’s type is, printing a special character after it
  - (blank) : Regular file
  - * : Executable program or command file
  - / : Directory
  - @ : Symbolic link
  - | : FIFO (named pipe)
  - = : Socket
- ls –i command prints i-node number for each file

Inodes

- Administrative information for each object in the filesystem.
- Inodes reside on disk and do not have names. Instead, they have indices (numbers) indicating their positions in the array of inodes.
- Each inode generally contains:
  - The location of the item's contents on the disk, if any
  - The item's type (e.g., file, directory, symbolic link)
  - The item's size, in bytes, if applicable
  - The time the file's inode was last modified (the *ctime*)
  - The time the file's contents were last modified (the *mtime*)
  - The time the file was last accessed (the *atime*) for *read* , *exec,* etc
  - A reference count: the number of names the file has
  - The file's owner (a UID)
  - The file's group (a GID)
  - The file's *mode bits* (also called *file permissions* or *permission bits*)

Manipulating Files

- `touch <file>`
  - create a new file or change last modified date
- `mv <file1> <file2>`
  - Rename file1 as file2
- `mv <file1> <dir>`
  - move file1 into the dir directory
- `mv <file1> <dir/file2>`
  - move file1 into dir and rename as file2
- `cp <file1> [<file2>|<dir>|<dir/file2>]`
  - copy file with new name into directory, or both
- `rm [-i] <file(s)>`
  - remove file or list of files

Creating and Removing Directories

- `mkdir <directory_name>`
  - create a subdirectory of the current directory
- `rmdir <directory_name>`
  - remove directory
  - only works for empty directories
- `rm –r <directory_name>`
  - remove directory and all of its contents, including subdirectories, recursively (-r)

Creating Links

- `ln –s <existing_file> <link_name>`
  - creates a symbolic link (-s)
  - *link_name* is a pointer to *existing file*, which may be in another directory or even on another physical machine
  - omit –s to create a hard link – must be in same physical partition of same device; *link_name* is another name for *existing_file*

File Ownership

- Each file has a single owner
- chown command can be used to change the owner; usually only root can use this command
- Each file also belongs to a single group
- Groups may have different permissions than everyone else

File Permissions

- Permissions used to allow or disallow access to files or directories
- Three types of permission:
  - Read (r)
  - Write (w)
  - Execute (x)
- Permissions exists on three levels
  - User (u)
  - Group (g)
  - Other (o)

File Permissions (cont.)

- `chmod <mode> <file(s)>`
  - `chmod 700 textfile`

$$\begin{tikzcd}
& \texttt{chmod 700 textfile} \arrow[dl] \arrow[d] \arrow[dr] & \\
\texttt{rwx} & \texttt{rwx} & \texttt{rwx} \\
\texttt{user} & \texttt{group} & \texttt{other}
\end{tikzcd}$$

  - `chmod g+rw textfile`

$$\begin{tikzcd}
& \texttt{chmod g+rw textfile} \arrow[dl] \arrow[d] \arrow[dr] & \\
\texttt{ugo} & \texttt{+/-} & \texttt{rwx}
\end{tikzcd}$$

  - `g+rw changes permissions to 760 (octal)`

File Modification Date

- Last time a file was changed
- Useful when ...
  - there are many copies of a file
  - many users are working on a file
- `touch` command can be used to update the modification date to the current date (or to create a file if it does not yet exist)

Looking at File Contents

- `cat textfile1 textfile2`
  - short for concatenate
  - output the contents of *textfile1*, then the contents of *textfile2*
- `less/more textfile`
  - scroll through *textfil*e one screen at a time
  - allows forward and backward scrolling and searching

Filename Substitution (Globbing)

- It is the process by which the shell expands a string containing wildcards into a list of filenames
- All of the commands covered here that take file names as arguments can also use wildcards
  - Asterisk (*) matches zero or more characters
    - x* matches the file x as well as x1, x2, xabc, etc.
  - Question mark (?) matches exactly one character
    - x? matches the file x1 as well as x2, xy, etc.
  - Square brackets ([ ]) matches a range of characters
    - [abc] or [a-c] matches one letter a, b, or c
    - [a-np-z]* matches all files that start with any lowercase letter but o
  - If a ! follows the [, any character is matched except those enclosed in the brackets
    - [!a-z] matches any character except a lowercase letter
    - *[!o] matches any file that does not end with the lowercase letter o

Getting Help on UNIX Commands

- `man <command_name>`
  - shows all of the documentation for a command (`less`-style output)
- `apropos <keyword>`
  - shows you all of the commands with the specified keyword in their description
- `type <string>`
  - shows files whose absolute path contains *string*

Other File Systems

- SunOS has 3 different types of file systems
  - disk-based
  - distributed
  - pseudo
- The **disk-based** file systems include hard-disks,CDROMs, diskettes.
- The **distributed** file systems manage network resources.
- The **pseudo** file systems are memory-based and do not use any disk space.

Other File Systems (cont.)

- Disk-based file systems
  - ufs
    - UNIX File System, based on BSD Fat Fast File System (default)
  - hsfs
    - High Sierra File System, used by CDROMs. Very similar to ufs, except that it does not support writable media or hard links
  - pcfs
    - PC File System, to allow read/write access to DOS formatted disks
  - cachefs
    - Cache File System, allows use of local disk to store frequently accessed data from a remote file system or CDROM

Other File Systems (cont.)

- Distributed file systems
  - nfs
    - Network File System, the default distributed file system type
  - rsfs
    - Remote File Share, AT&Ts RFS product
  - autofs
    - Automount File System, automounts NFS file systems, as needed, using NIS and NIS+ maps

Other File Systems (cont.)

- Pseudo file systems
  - tmpfs
    - Temporary File System, file storage in memory and swap without the overhead of writing to a ufs file
  - specfs
    - Special File System, allows access to the special character and block devices
  - lofs
    - Loopback File System, creates a virtual file system which can overlay or duplicate existing files
  - tfs
    - Translucent File System, allows mounting of a file system on top of existing files, with both visible

Other File Systems (cont.)

- procfs
  - Process Access File System, allows access to active processes and their images
- fdfs
  - File Descriptor File System, allows access to file names using descriptors
- namefs
  - Name File System, used by STREAMS for dynamic mounts of file descriptors on top of files
- fifos
  - First In First Out File System, allows process access to named pipe files
- swapfs
  - Swap File System, used by the kernel to manage swap space