Files and directories

Mythili Vutukuru

CSE, IIT Bombay

File System

- An organization of files and directories on disk
- An OS can implement one or more file systems
- Disks expose a set of blocks (usually 512 bytes)
- File system organizes files onto blocks
  - System calls translated into reads and writes on blocks
- We will study the following concepts about file systems
  - The file abstraction and system call API exposed to users
  - Data structures to organize data and metadata on disk and memory
  - Implementation of system calls like open, read, write using the data structures

File abstraction

```
fd = open(“/home/foo/a.txt”)
read(fd, ..)
write(fd, ..)
close(fd)
```

- File: sequence of bytes, stored persistently on disk
- Directory: container for files and other sub-directories
- Steps to access a file
  - Open a file using system call, get a file descriptor
  - File descriptor is a handle to refer to file for read/write
  - Close file when done accessing it
- Filesystem: OS subsystem that stores files and directories persistently on secondary storage like hard disks
  - File-related system calls are exposed to users to access files

Directory tree

- Files and directories arranged in a tree, starting with root ("/")

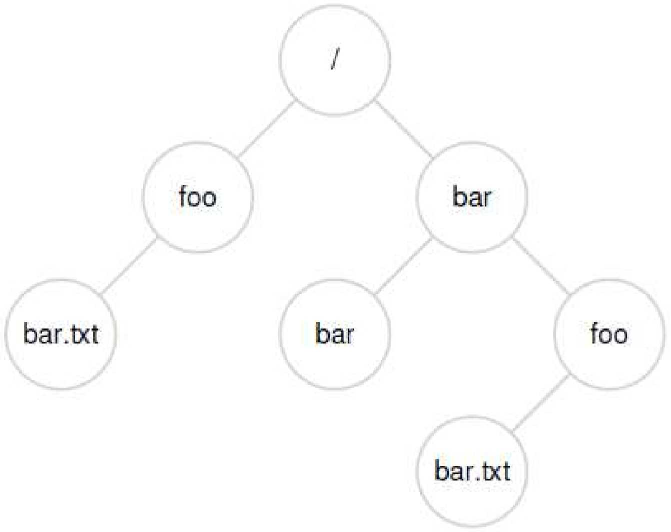


Figure 39.1: **An Example Directory Tree**

Image credit: OSTEP

Reading and writing a file

```
fd = open("/home/foo/a.txt")
char buf[64]
n = read(fd, buf, 64)
buf[0] = …
n = write(fd, buf, 64)
```

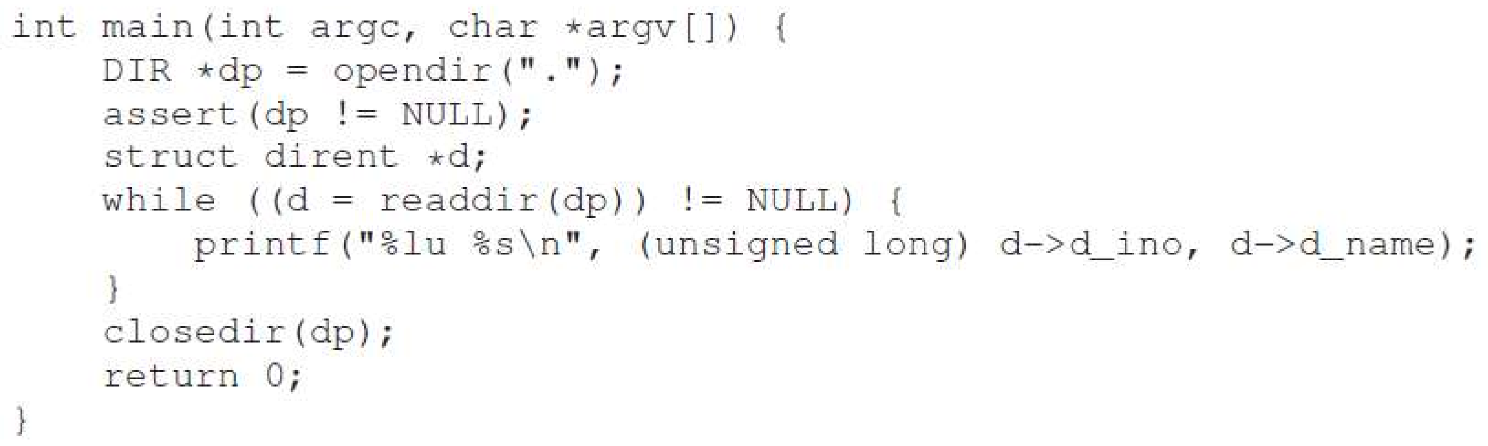
- After opening a file, process can read/write to a file as a stream
  - File descriptor used as handle to refer to the open file stream
  - Read system call reads specified number of bytes into a user-defined buffer, returns number of bytes read
  - Write system call writes specified number of bytes from a user-defined buffer, returns number of bytes written
- Read and write system calls update the offset into a file
  - After reading N bytes, next read will return the next set of bytes
  - Can also update the offset from which to read/write using a seek system call
  - Every open file descriptor will read/write file as independent stream, with independent offsets

Sockets, pipes, …

- Opening sockets and pipes also returns file descriptor
  - Serves as a handle for stream of data from socket or pipe
- Every process has array of file descriptors
  - One file descriptor for every open file / socket / pipe
  - First 3 entries are streams for stdin, stdout, stderr
- Several I/O streams handled via file-like interfaces (everything is a file!)

Operations on directories

- Every file is identified by a unique inode number in filesystem
- Directory is a special file that contains mappings between filenames and inode number of the file
- Directories can also be accessed like files, e.g., create, open, read, close
- For example, the “ls” program opens and reads all directory entries
  - Directory entry contains file name, inode number, type of file (file/directory) etc.

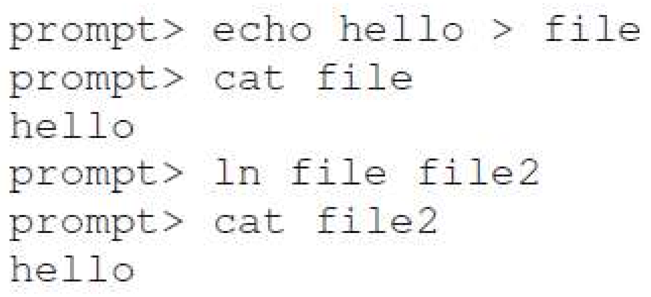


```
int main(int argc, char *argv[]) {
    DIR *dp = opendir(".");
    assert(dp != NULL);
    struct dirent *d;
    while ((d = readdir(dp)) != NULL) {
        printf("%lu %s\n", (unsigned long) d->d_ino, d->d_name);
    }
    closedir(dp);
    return 0;
}
```

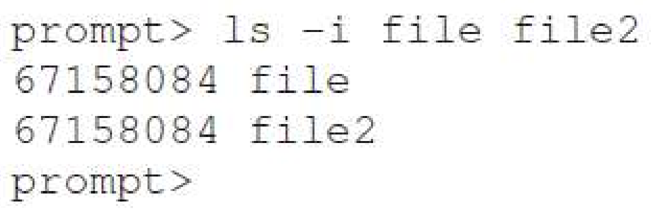
Image credit: OSTEP

Hard links

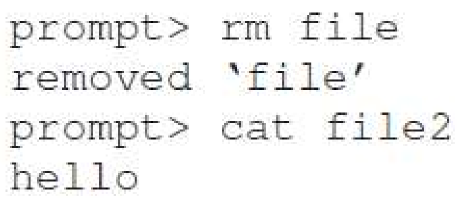
- Hard linking creates another file that points to the same inode number (and hence, same underlying data)
- If one file deleted, file data can be accessed through the other links
- Inode maintains a link count, file data deleted only when no further links to it
- You can only unlink, OS decides when to delete



```
prompt> echo hello > file
prompt> cat file
hello
prompt> ln file file2
prompt> cat file2
hello
```



```
prompt> ls -i file file2
67158084 file
67158084 file2
prompt>
```

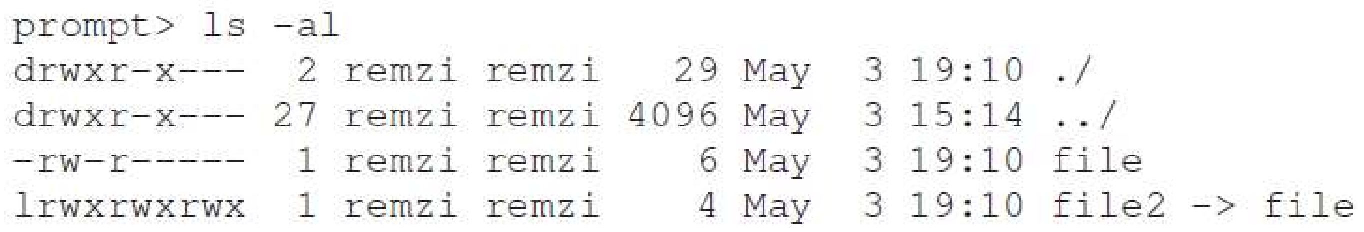


```
prompt> rm file
removed 'file'
prompt> cat file2
hello
```

Image credit: OSTEP

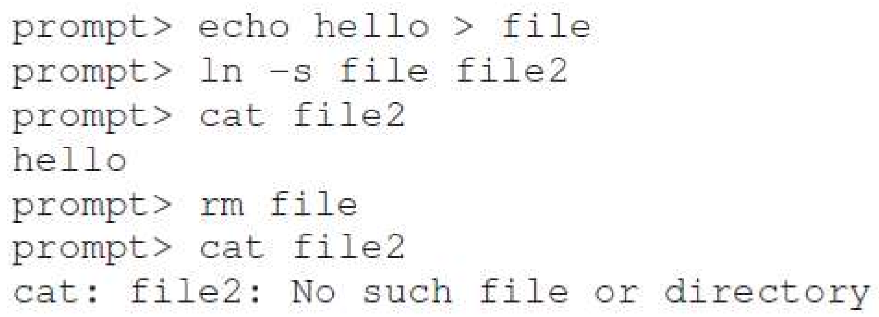
Soft links or symbolic links

- Soft link is a file that simply stores a pointer to another filename



```
prompt> ls -al
drwxr-x---  2 remzi remzi   29 May  3 19:10 ./
drwxr-x--- 27 remzi remzi 4096 May  3 15:14 ../
-rw-r-----  1 remzi remzi    6 May  3 19:10 file
lrwxrwxrwx  1 remzi remzi    4 May  3 19:10 file2 -> file
```

- If the main file is deleted, then the link points to an invalid entry: dangling reference

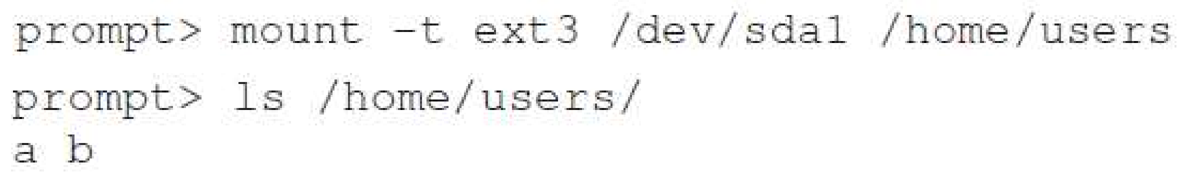


```
prompt> echo hello > file
prompt> ln -s file file2
prompt> cat file2
hello
prompt> rm file
prompt> cat file2
cat: file2: No such file or directory
```

Image credit: OSTEP

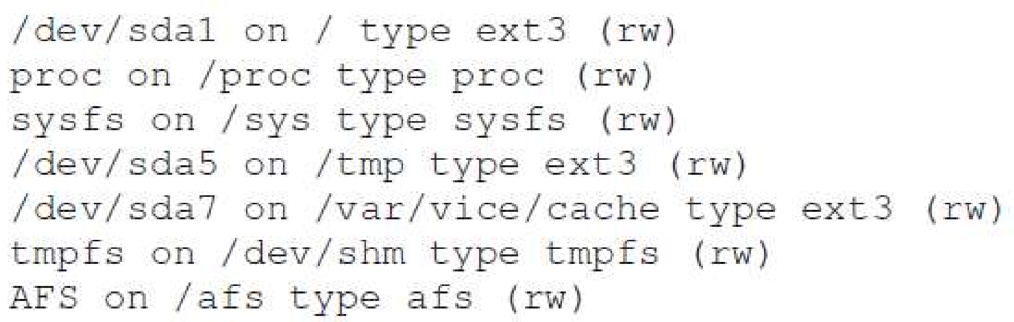
Mounting a file system

- Mounting a file system connects the files to a specific point in the directory tree



```
prompt> mount -t ext3 /dev/sda1 /home/users
prompt> ls /home/users/
a b
```

- Several devices and file systems are mounted on a typical machine, accessed with `mount` command



```
/dev/sda1 on / type ext3 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
/dev/sda5 on /tmp type ext3 (rw)
/dev/sda7 on /var/vice/cache type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
AFS on /afs type afs (rw)
```

Image credit: OSTEP