Basic UNIX

Console Login

Enter userid and password

Login: edsall
password:

Basic UNIX

Remote Login

Telnet (Bad)

<lister> telnet -a -x isua.iastate.edu
Trying 129.186.1.202...
Connected to isua2.iastate.edu (129.186.1.202).
Escape character is '^]'.
Waiting for encryption to be negotiated...
[Trying KERBEROS4 ...]
[Kerberos V4 accepts you]
[Kerberos V4 challenge successful]
done.

Using encryption for Input and Output

login: edsall
password:
Last login: Mon Nov 24 07:44:28 from du139-19.aitlabs

Basic UNIX

Remote Login

SSH (Good)

<isua2> ssh lister.ait
edsall's password:
Authentication successful.

Basic UNIX

Post Login

Message of the Day

Processes soft limited to 30 CPU minutes
Processes hard limited to 30 CPU minutes

Welcome to Project Vincent (OSF1 4.0)

Digital UNIX V4.0F (Rev. 1229); Tue Feb 18 15:33:10 CST 2003

This machine is provided specifically for telnet access to Project Vincent.

This machine has a 500 Mhz processor and 1GB of memory.

Do not leave background jobs running unattended.
You MAY run X windows application on this machine (yes, really).

Basic UNIX

UNIX Commands

Syntax

command ***-option [option value]*** **<u>args</u>**

Where:

command – full or relative path

option – usually one letter

<u>args</u> – arguments used by the command such as file names

Basic UNIX

UNIX Commands

Getting Help - “Man” pages

1) Keyword search:

man -k *keyword*

2) Command name is known

man <*command*>

3) Multiple commands, same name

man <*section*> <*command*>

Basic UNIX

UNIX Commands

“Man” page Examples

<lister> man -k chmod

chmod (1) - change file access permissions
chmod (2) - change permissions of a file
fchmod [chmod] (2) - change permissions of a file

Basic UNIX

UNIX Commands

“Man” page Examples

<lister> man chmod

CHMOD(1) User Commands CHMOD(1)

NAME

chmod – change file access permissions

SYNOPSIS

chmod [OPTION]... MODE[,MODE]... FILE...
chmod [OPTION]... OCTAL-MODE FILE...
chmod [OPTION]... --reference=RFILE FILE...

DESCRIPTION

This manual page documents the GNU version of **chmod**.
chmod changes the permissions of each given file according to mode, which can be either a symbolic representation of changes to make, or an octal number representing the bit pattern for the new permissions.

Basic UNIX

UNIX Commands

“Man” page Examples

<lister> man 2 chmod

CHMOD(2) Linux Programmer’s Manual CHMOD(2)

NAME

chmod, fchmod – change permissions of a file

SYNOPSIS

#include <sys/types.h>
#include <sys/stat.h>

int chmod(const char *path, **mode_t** mode);
int fchmod(int fildes, **mode_t** mode);

DESCRIPTION

The mode of the file given by path or referenced by fildes is changed.

Modes are specified by or’ing the following:

S_ISUID 04000 set user ID on execution

Basic UNIX

UNIX Commands

Exercise

Find a command that deals with your favourite topic and then call up the man page for one of those commands.

For those lacking imagination, try

man -k file

Basic UNIX

Files, Devices and Directories

Everything is a “File”

- Text files – usually created with an editor
- Binary files – executable programs or data
- Devices – terminals, printers, tape drives, disks, CDROM, serial ports, etc.
- Directories – contain *information* about files
- Links

Basic UNIX

Files, Devices and Directories

Determining a File's Type

- Open it with an editor

```
<lister> yo. 'sup, home boy? Vi /bin/ls

^?ELF^A^A^A^@^@^@^@^@^@^@^@^@^B^@^C^@^A^@^@^@À<98>^D^
H4^@^@^@D^E^A^@^@^@^@^@4^@^@^G^@
(^@^Y^@^X^@^F^@^@^@4^@^@^@4<80>^D^H4<80>^D^Hà^@^@^@à^
@^@^@^E^@^@^@^D^@^@^@^C^@^@^@^T^A^@^@^T<81>^D^H^T<81>
^D^H^S^@^@^@^S^@^@^@^D^@^@^@^A^@^@^@^A^@^@^@^@^@^@^@^
@<80>^D^H^@<80>^D^H^X^@^A^@^X^@^A^@^E^@^@^@^@^P^@^@^A
^@^@^@ ^@^A^@<90>^E^H
```

- Use the **file** command

```
<lister> file /bin/ls
/bin/ls: ELF 32-bit LSB executable, Intel 80386,
   version 1 (SYSV), for GNU/Linux 2.2.5,
   dynamically linked (uses shared libs), stripped
```

- Look at the file's *permissions* (more on this later)

Basic UNIX

Files, Devices and Directories

“Hidden” Files

Sometimes you don't want a user to know about a file

Sometimes you don't want “config file clutter”

Hidden files are not a type of file

Hidden files are a type of file *naming*

Hidden files have file name beginning with a “.” (dot)

Basic UNIX

Files, Devices and Directories

Links, Directories and Inodes

Every file has a **name** and an **inode** number associated with it

Inodes contain information about the file and its physical location

Directories are files which contain the file names and inode numbers

Links are alternate ways to access files

Basic UNIX

Files, Devices and Directories

Hard Links

Hard links create a separate reference to a file.

They are directory entries that allow multiple names to point to the same inode number

If you delete a file by one name, the file is still accessible because another directory entry still points to that inode number.

Basic UNIX

Files, Devices and Directories

Symbolic Links

Symbolic links are files containing a path name (more on this later) to a file.

Removing the link does not remove a directory entry.

If you remove the file, the link goes to nothing (and UNIX will let you know that!)

Basic UNIX

Files, Devices and Directories

Properties of Links

Symbolic links can cross devices. Hard links cannot since they depend on inodes.

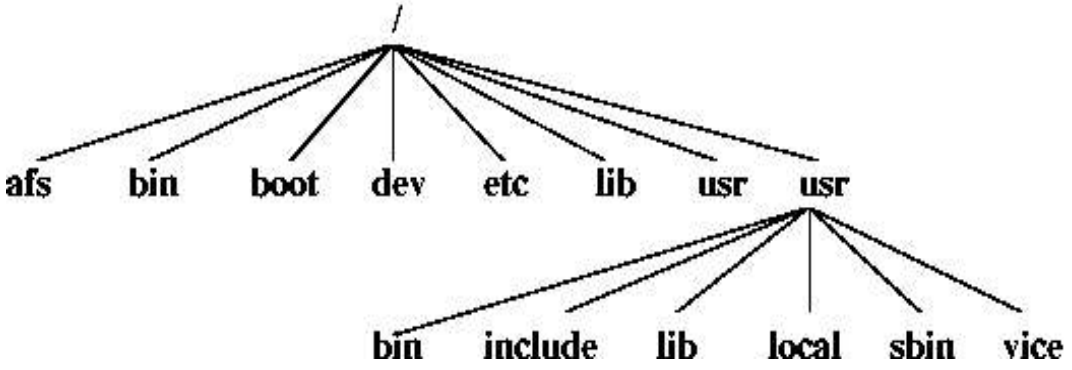
Links allow you to point to the same file from multiple directories.

Links can be point to directories as well.

Basic UNIX

Files, Devices and Directories

UNIX Directory Tree



Files are arranged in a tree

Branches – directories

Leaves – files

Basic UNIX

Files, Devices and Directories

File Names and Paths

Pathname is the set of pointers to be followed to get to a file

Two types of paths:

- *Absolute* – starts from the *root* directory
- *Relative* – starts from the *current working directory*. "Dot" directories (. and ..) are needed in some cases

Basic UNIX

Files, Devices and Directories

Relative Paths

“Dot” directories

Unix provides two convenient shortcuts, the “dot” directories

(. and ..)

- “.” (dot) refers to the current working directory
- “..” (dot-dot) refers to the parent directory of the current working directory

Basic UNIX

Files, Devices and Directories

File Names and Paths

Example – Absolute Path

```
/home/edsall/myfile
```

(look in *home* for the entry to *edsall*, look in *edsall* for the entry to *myfile*)

Example – Relative Path

Starting from /home/edsall, to reach /home/camelot/WWW:

```
../camelot/WWW
```

(look up one directory then look for the entry to *camelot*, then the entry for *WWW*)

Basic UNIX

Files, Devices and Directories

Moving Around
The **cd** command

cd allows you to **c**hange **d**irectories. The directory to which you change becomes the *current working directory (CWD)*

Examples:

```
<lister> cd /home/edsall
```

/home/edsall is now the CWD

```
<lister> cd ../../usr/games/
```

/usr/games is now the CWD

Basic UNIX

Files, Devices and Directories

Exercises

Use **cd** to go to your home directory

cd

Use **cd** to go into the parent directory

cd ..

Basic UNIX

Files, Devices and Directories

Exercises

Use **cd** to go to */usr* using an absolute path

cd /usr

Return to your home directory. Use **cd** to go to */usr* using a relative path

cd

cd ../../usr

Basic UNIX

Files, Devices and Directories

File Attributes
The **ls** command

ls allows you to **list** directories.

Example:

```
<lister> ls /usr
afsws    etc        libexec     src
athena   games      local       tmp
bin      include    lost+found  vice
dict     kerberos   sbin        X11R6
doc      lib        share
```

Basic UNIX

Files, Devices and Directories

Exercise

The **ls** command

Use **ls** to get a listing of your current working directory

```
# ls
# ls .
```

Use **ls** to get a listing of the parent directory

```
# ls ..
```

Basic UNIX

Files, Devices and Directories

File Attributes
The **ls** command

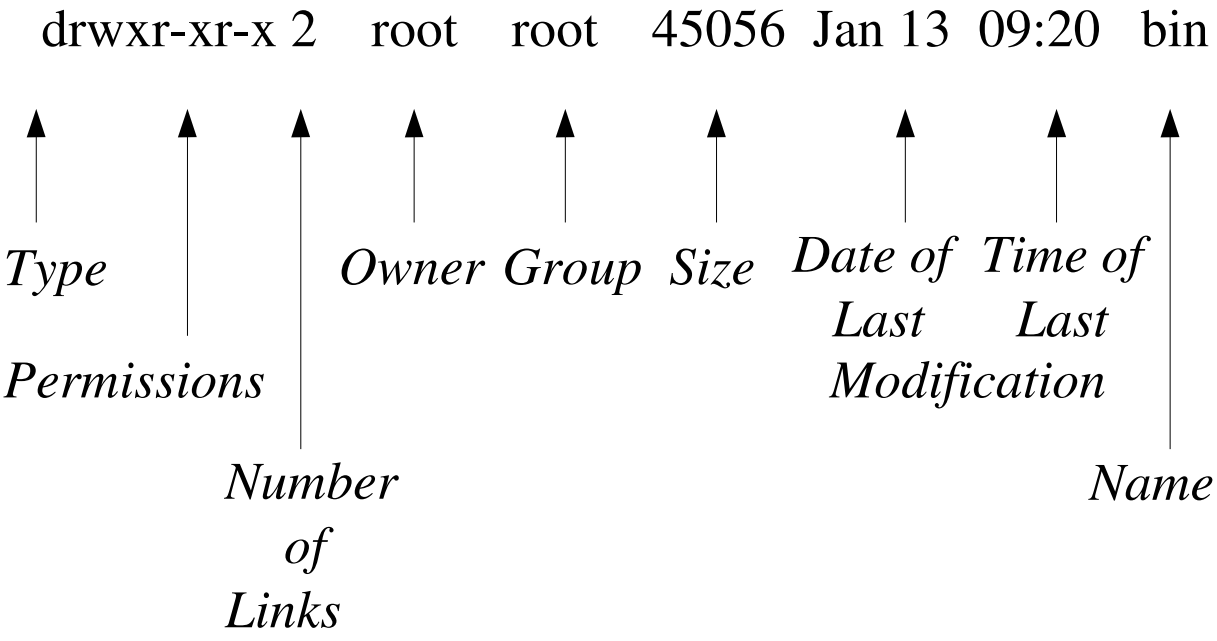
Example: A *long* listing

```
<lister> ls -l /usr
total 176
drwxr-xr-x    4 root     root  4096 Nov  3 14:08 afsws
drwxr-xr-x    8 root     root  4096 Nov  3 14:08 athena
drwxr-xr-x    2 root     root 45056 Jan 13 09:20 bin
drwxr-xr-x    2 root     root  4096 Feb  6  1996 dict
drwxr-xr-x    3 root     root  4096 Nov 10 10:17 doc
drwxr-xr-x    2 root     root  4096 Feb  6  1996 etc
drwxr-xr-x    4 root     root  4096 Nov  1 16:39 games
drwxr-xr-x  113 root     root  8192 Nov  3 14:08 include
drwxr-xr-x    8 root     root  4096 Apr  4  2003 kerberos
drwxr-xr-x   94 root     root 49152 Jan 13 09:20 lib
drwxr-xr-x    8 root     root  4096 Nov  3 13:52 libexec
drwxr-xr-x   17 root     root  4096 Jan 13 09:24 local
drwx------    2 root     root 16384 Nov  1 16:23 lost+found
drwxr-xr-x    2 root     root  8192 Nov  4 09:39 sbin
drwxr-xr-x  200 root     root  4096 Nov 12 14:50 share
drwxr-xr-x    5 root     root  4096 Nov  3 14:08 src
lrwxrwxrwx    1 root     root    10 Nov  1 16:25 tmp -> ../
   var/tmp
drwxr-xr-x    4 root     root  4096 Nov  3 14:08 vice
drwxr-xr-x    8 root     root  4096 Nov  1 16:49 X11R6
```

Basic UNIX

Files, Devices and Directories

File Attributes



Basic UNIX

Files, Devices and Directories

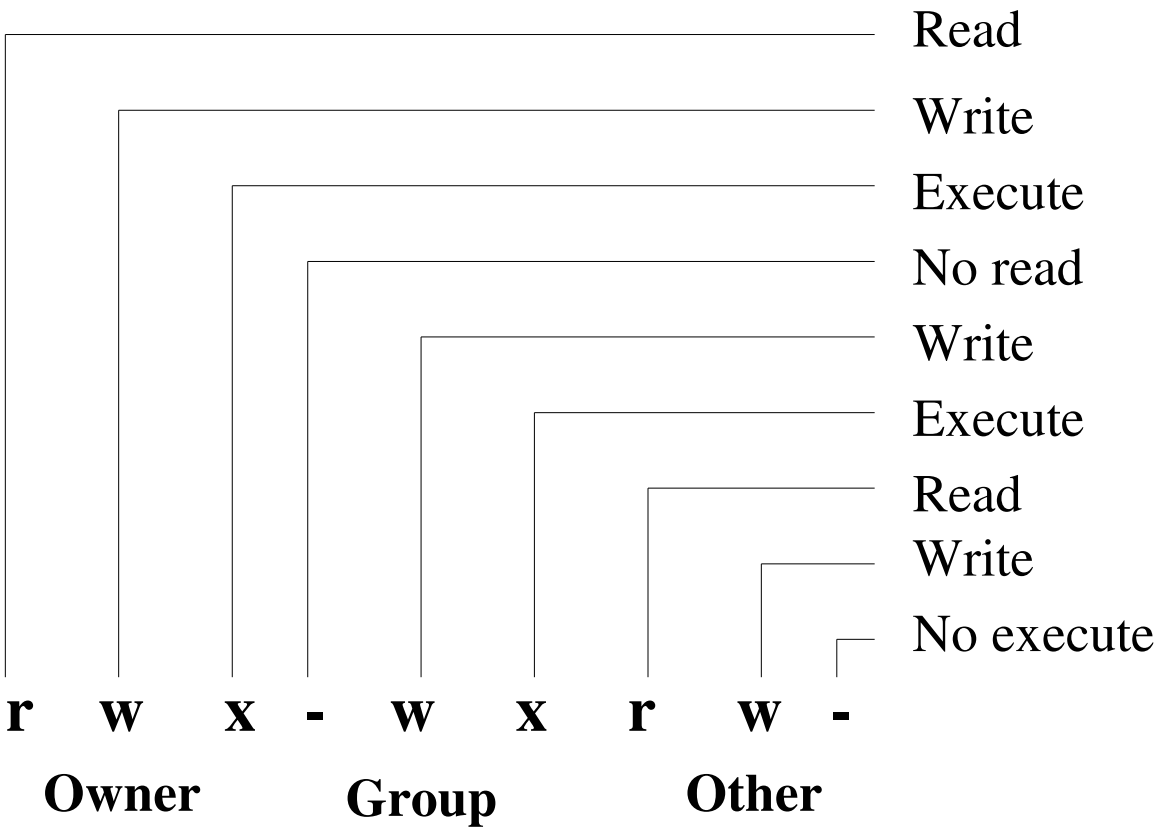
File Types

- **d** – directory
- **b** – block special device
- **c** – character special device
- **l** – link
- **s** – socket
- **-** – normal file

Basic UNIX

Files, Devices and Directories

File Permissions (Modes)



Basic UNIX

Files, Devices and Directories

File Permissions (Modes)

Permissions depend on the type of file

- Read/List
  - Files – you can view the contents
  - Directories – you can list the files in the directory
- Write/Modify
  - Files – you can edit the contents
  - Directories – you can add and remove files

Basic UNIX

Files, Devices and Directories

File Permissions (Modes)

- Execute/Enter
  - Files – you execute the file/program
  - Directories – you can enter the directory (**cd** into it)

    This makes sense. To run a program, you must “load” its address into memory. To enter a directory, you must “load” the inode to the directory into memory.

Basic UNIX

Files, Devices and Directories

Changing Permissions (chmod)

Use **chmod** to change file *modes*

chmod [*options*] *mode* file

Two ways to specify modes:

1. Use mnemonics

2. Use *octal* modes

Basic UNIX

Files, Devices and Directories

Changing Permissions (chmod)

Using mnemonics

Specify the recipient of the permission followed by the permission:

u=rwx
g+rw
o=-x
a=r

u=user, g=group, o=other, a=all

r=read, w=write, x=execute/search

Basic UNIX

Files, Devices and Directories

Changing Permissions (chmod)

Using *octal*

The permissions correspond to *bits* being *set*

4=read
2=write
1=execute/search

chmod 755 myprog

7 = 4+2+1 = r + w + x for owner
5 = 4+0+1 = r + - + x for group
5 = 4+0+1 = r + - + x for other

Basic UNIX

Files, Devices and Directories

Exercises

Create a zero length file in your home directory

cd
touch myfile

Make the file executable by you

chmod u+x myfile

Check your work

#ls -l myfile

Basic UNIX

Files, Devices and Directories

Useful File Commands

- **cp** *file1 file2* - copy a file
- **mv** *file1 file2* - rename or relocate a file
- **rm** *file* - remove a file (BE VERY CAREFUL)
- **mkdir** *directory* - create a new directory
- **rmdir** *directory* - delete a directory
- **grep** *regexp file* - search a file for an expression
- **diff** *file1 file2* - compare two files
- **more** *file* - “page” a file
- **lpr** *file* - print a file