CS699
Software Systems Lab

Kavi Arya

CSE, IIT Bombay

Session 2: Introduction to Linux

Contents

- A quick guide to Linux
  - Background
  - Using Linux
- Linux in the Marketplace
- Commercial Linux Applications
- Additional Resources

What is Linux

- A fully-networked 32/64-Bit Unix-like Operating System
  - Unix Tools Like sed, awk, and grep (explained later)
  - Compilers Like C, C++, Fortran, etc.
  - Network Tools Like telnet, ftp, ping, traceroute
- Multi-user, Multitasking, Multiprocessor
- Has the X Windows GUI
- Coexists with other Operating Systems
- Runs on multiple platforms
- Includes the Source Code

Where did it come from?

- Linus Torvalds created it
  - with assistance from programmers around the world
  - first posted on Internet in 1991
- Linux 1.0 in 1994; 2.2 in 1999; 2.6 beta 15-7-03
- Today used 10s of millions of computers
  - 10s of 1000's of programmers working to enhance it

Open Source **Software**

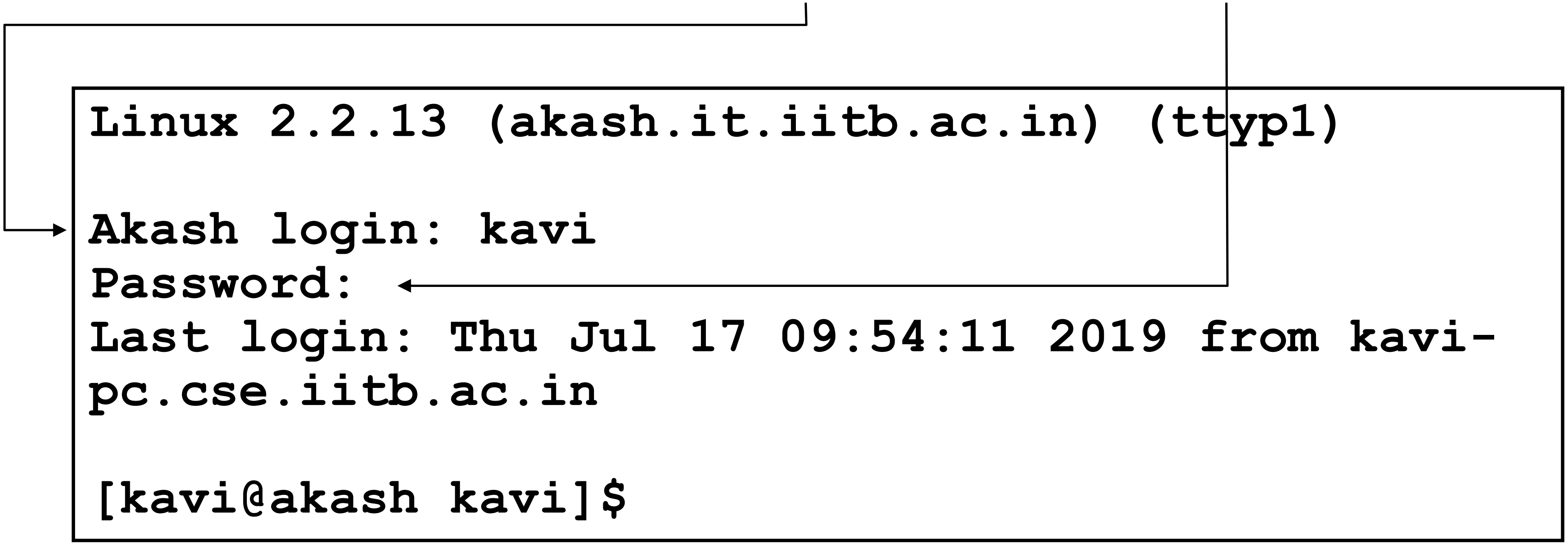
- When programmers on Internet can read, redistribute, and modify source of sw, **it evolves**
- People improve it, adapt it, fix bugs at speed that, compared to conventional software development, seems **astonishing**

Why is it significant?

- Growing popularity
- Powerful
  - Runs on multiple hardware platforms
  - Users like its speed and stability
  - No requirement for latest hardware
- It’s “free”
  - Licensed under GPL
  - Vendors are distributors who package Linux

Logging In

- Before you can use it you must login by specifying your account and password:



```
Linux 2.2.13 (akash.it.iitb.ac.in) (ttyp1)

Akash login: kavi
Password:
Last login: Thu Jul 17 09:54:11 2019 from kavi-
pc.cse.iitb.ac.in

[kavi@akash kavi]$
```

Rule Number 1

- Do not login as root unless you **have to**
- Root is system superuser
  - Normal protection mechanisms can be overridden
  - Careless use can cause damage
  - Has access to everything by default
- Root is only user defined when you install
  - First thing is to change root’s password
  - Second is to define “normal” users for everyday use

Creating a new user

- Use the `useradd` command
- Use the `passwd` command to set password

Adding a new user

- Limits on users can be controlled by
  - Quotas
  - `ulimit` command
- Authority levels for a user controlled by group membership

Users and Groups

- Groups define functional areas/responsibilities
- They allow a collection of users to share files
- A user can belong to multiple groups
- We can see what groups we belong to using `groups` command:

```
faculty
```

Using the new user

- Now logoff using `exit` command
- login as the new user

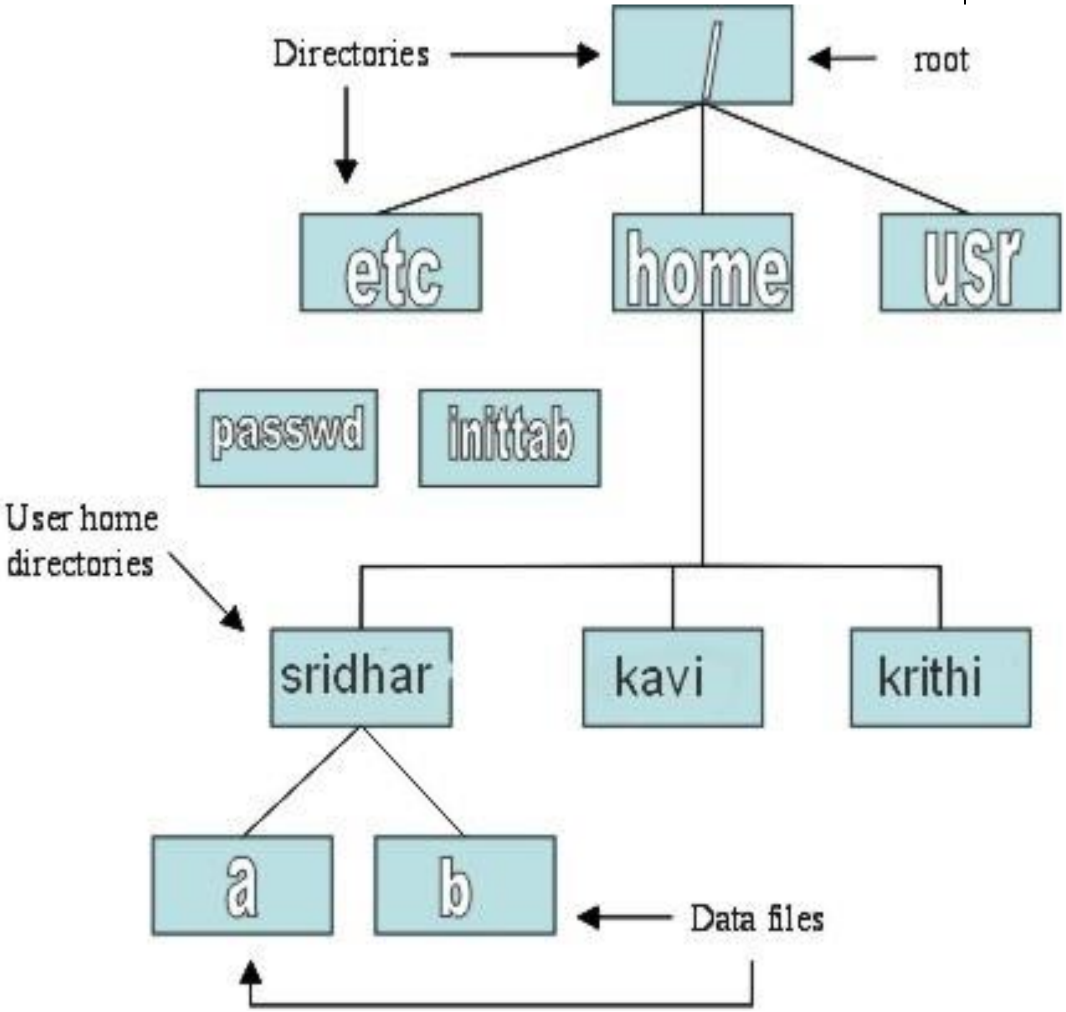
```
Linux 2.2.13 (akash.it.iitb.ac.in) (ttyp1)

Akash login: kavi
Password:
Last login: Thu Jul 17 09:54:11 2019 from kavi-
pc.it.iitb.ac.in

[kavi@akash kavi]$
```

You need help?

- Linux equivalent of HELP is `man` (manual)
  - **`man -k <keyword>`** finds cmds with that keyword
  - **`man <command>`** displays help for that command
    - Output is presented a page at a time.
    - Use **`b`** for to scroll backward,
      `f` or a space to scroll forward and
      `q` to quit

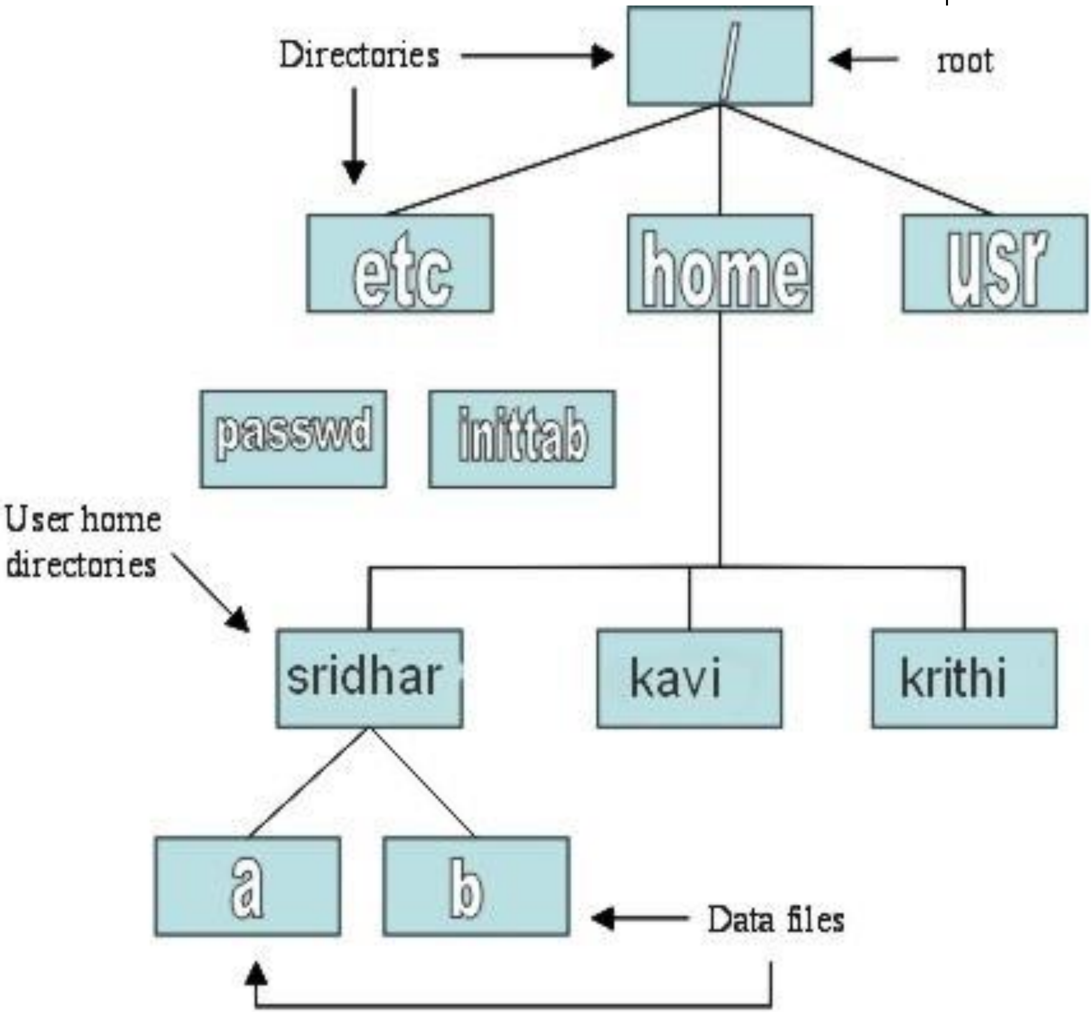
Linux File System Basics

- Linux files stored in single rooted, hierarchical file system
  - Data files stored in directories (folders)
  - Directories nested as deep as needed



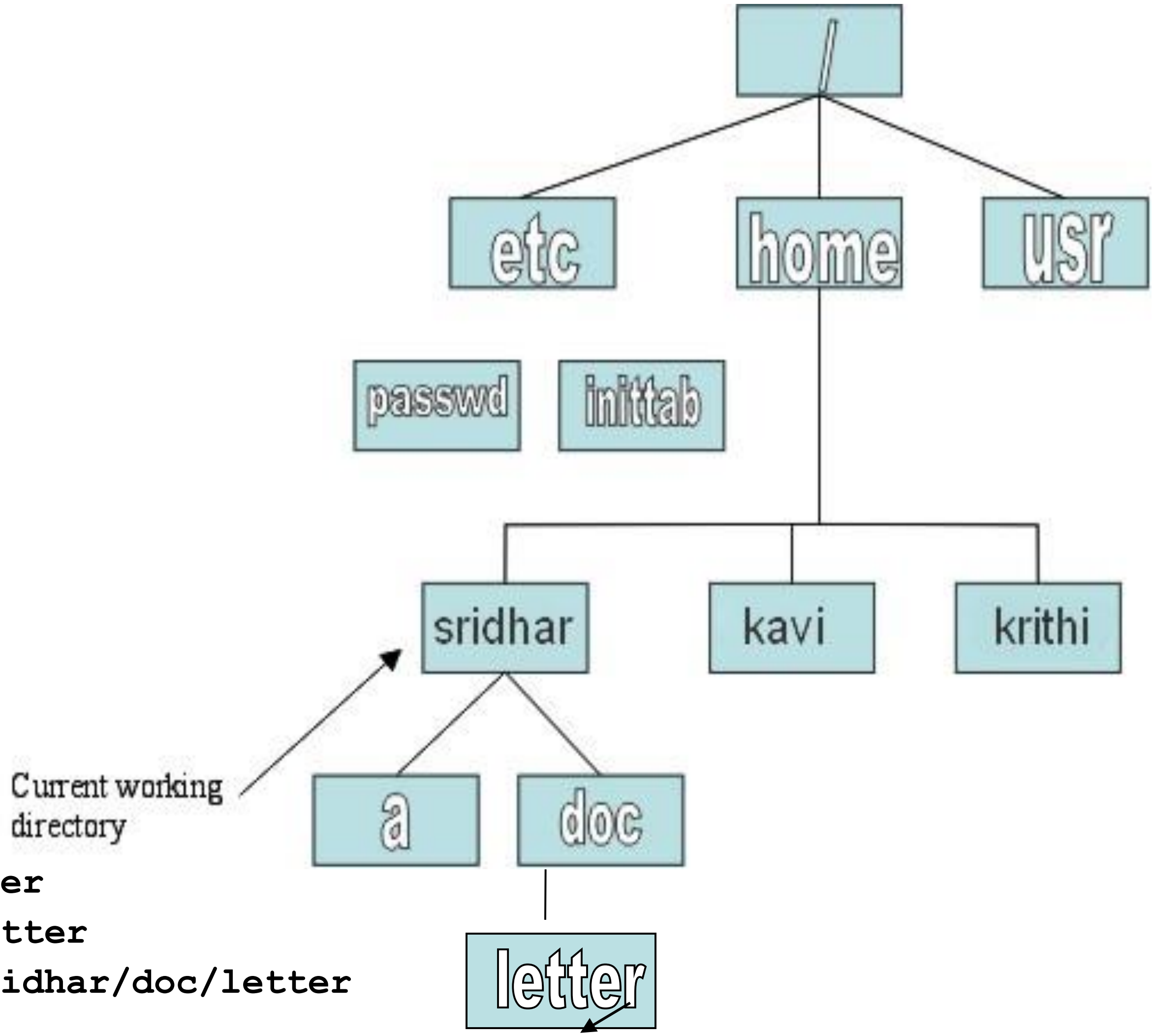
Naming Files

- Files named by
  - naming each containing directory
  - starting at root
- Known as *pathname*



The Current Directory

- One directory is designated *current working directory*
  - if you omit leading / then path name is relative to current working directory
  - Use pwd to find out where you are



```
doc/letter
./doc/letter
/home/sridhar/doc/letter
```

Some Special File Names

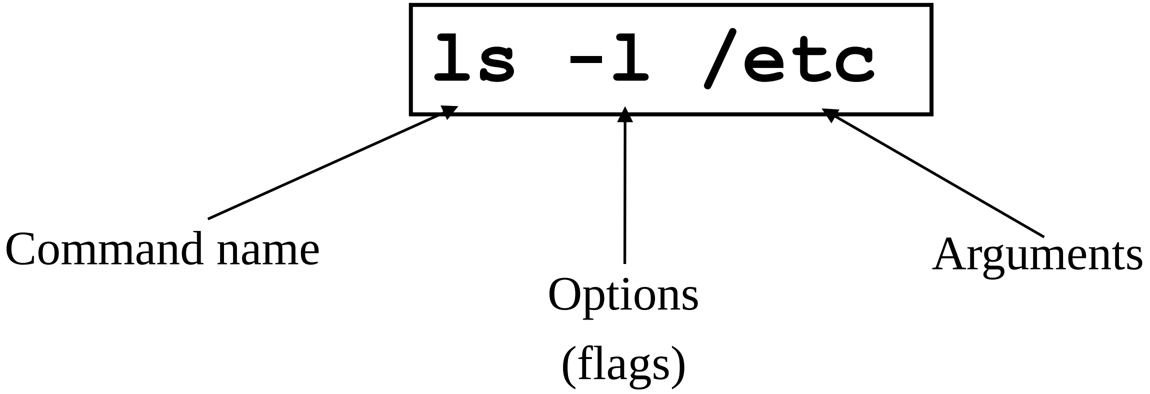
- Some file names are special:
  - **`/`** Root directory (not to be confused with root user)
  - **`.`** Current directory
  - **`..`** Parent (previous) directory
  - **`~`** My home directory
- Examples:
  - **`./a`** same as **`a`**
  - **`../kavi/x`** go up one level then look in dir **`kavi`** for **`x`**

Special Files

- **`/home`** - all users' home directories stored here
- **`/bin`**, **`/usr/bin`** - system commands
- **`/sbin`**, **`/usr/sbin`** - cmds used by sysadmins
- **`/etc`** - all sorts of configuration files
- **`/var`** - logs, spool directories etc.
- **`/dev`** - device files
- **`/proc`** - special system files

Linux Command Basics

- To execute a command, type its name and arguments at command line



```
ls -l /etc
```

Command name

Options (flags)

Arguments

Standard Files

- UNIX concept of “standard files”
  - – **standard input** (where cmd gets input)
    - default is terminal
  - – **standard output** (where cmd writes output)
    - default is terminal
  - – **standard error** (where cmd writes error msgs)
    - default is terminal

Redirecting Output

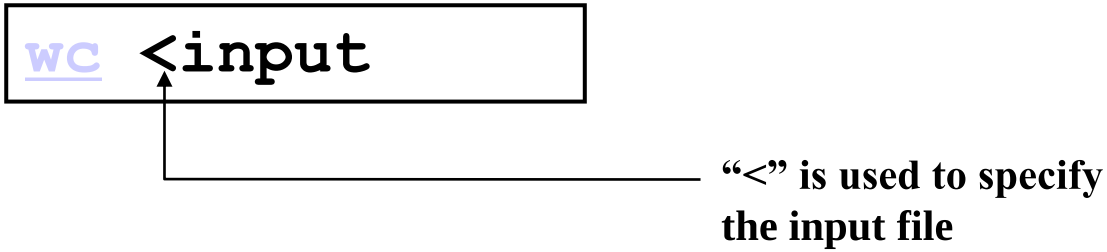
- Output of cmd may be sent (piped) to a file:

```
ls -l >output
```

“>” is used to specify the output file

Redirecting Input

- The input of a cmd may come (be piped) from a file:

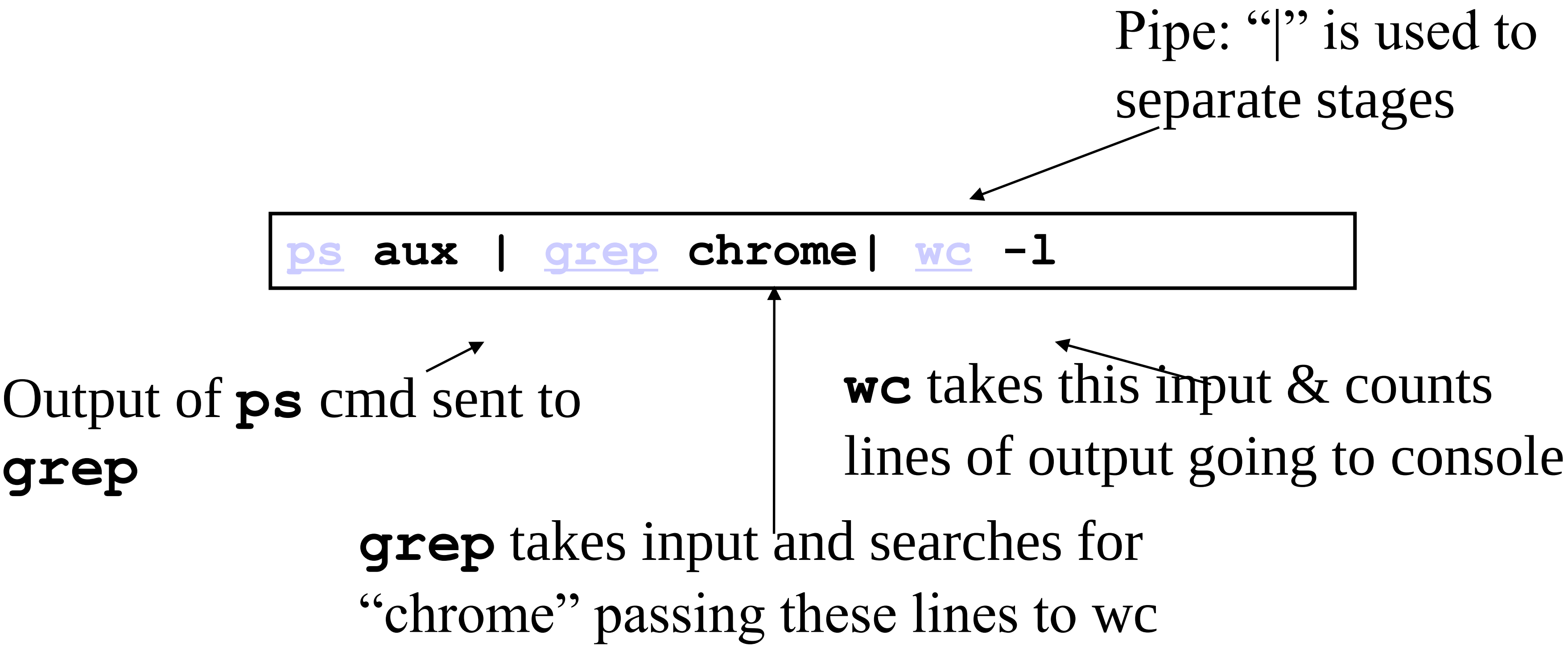


```
wc <input
```

“<” is used to specify the input file

Connecting commands with Pipes

- The output of one command can become the input of another:



Pipe: “|” is used to separate stages

```
ps aux | grep chrome| wc -l
```

Output of **ps** cmd sent to **grep**

wc takes this input & counts lines of output going to console

grep takes input and searches for “chrome” passing these lines to wc

Common Commands

- `pwd` - print (display) the working directory
- `cd` ***`<dir>`*** - change current working directory to ***dir***
- `ls` – list files in current working directory
- `ls` –**`l`** – list files in current working directory in long format

File Commands

- `cp <fromfile> <tofile>`
  - Copy from the <fromfile> to the <tofile>
- `mv <fromfile> <tofile>`
  - Move/rename the <fromfile> to the <tofile>
- `rm <file>`
  - Remove the file named <file>
- `mkdir <newdir>`
  - Make a new directory called <newdir>
- `rmdir <dir>`
  - Remove an (empty) directory

More Commands

- `who`
  - List who is currently logged on to system
- `whoami`
  - Report what user you are logged on as
- `ps`
  - List your processes on system
- `ps aux`
  - List all processes on system
- `echo` ***"A string to be echoed"***
  - Echo a string (or list of arguments) to terminal

More Commands

- `alias` - used to tailor commands:
  - `alias erase=rm`
  - `alias grep=”grep -i”`

More Commands

- **awk** - file processing language suited to data manipulation and retrieval of information from text files
- **chown** - sets user ID (UID) to owner for files and directories named by pathname arguments. Useful when going from test to production

```
chown -R apache:httpd /usr/local/apache
```

More Commands

- `diff` - attempts to determine minimal set of changes needed to convert a file specified by first argument into file specified by second argument
- `find` - Searches a given file hierarchy specified by path, finding files that match the criteria given by expression

More Commands

- grep - Searches files for one or more pattern arguments. It does plain string, basic regular expression, and extended regular expression searching

```
find ./ -name "*.c" | xargs grep -i "fork"
```

More Commands

- kill - sends a signal to a process or process group
- You can only kill your own processes unless you are root

```
UID          PID  PPID  C STIME TTY          TIME CMD
root        6715  6692  2 14:34 ttyp0    00:00:00 sleep 10h
root        6716  6692  0 14:34 ttyp0    00:00:00 ps -ef
[root@akash log]# kill 6715
[1]+  Terminated              sleep 10h
```

More Commands

- tar - manipulates archives
  - – Archive is single file that contains complete contents of a set of other files; it preserves directory hierarchy containing original files.

```
tar -tzf imap-4.7.tar.gz
imap-4.7/
imap-4.7/src/
imap-4.7/src/c-client/
imap-4.7/src/c-client/env.h
imap-4.7/src/c-client/fs.h
```

Shells

- An interface between the Linux system and the user
- Used to call commands and programs
- An interpreter
- Powerful programming language
  - **“Shell scripts” = .bat .cmd EXEC REXX**
- Many available (**bsh; ksh; csh;** **bash**; **tcsh**)

Another definition of a Shell

- A shell is any program that takes input from the user, translates it into instructions that the operating system can understand, and conveys the operating system's output back to the user.
  - i.e. Any User Interface
  - Character Based v Graphics Based

Why Do I Care About The Shell?

- Shell is Not Integral Part of OS
  - UNIX Among First to Separate
  - Compare to MS-DOS, Mac, Win95, VM/CMS
  - GUI is NOT Required
  - Default Shell Can Be Configured
    - `chsh -s /bin/bash`
    - `/etc/passwd`
  - Helps To Customize Environment

Shell Scripts

```
#!/bin/bash
while
true
do
    cat somefile > /dev/null
    echo  .
done
```

```
/* */
do forever
   ‘PIPE < SOME FILE | hole’
   say ‘.’
end
```

File Permissions

- Every file
  - Is owned by someone
  - Belongs to a group
  - Has certain access permissions for owner, group, and others
  - Default permissions determined by `umask`

File Permissions

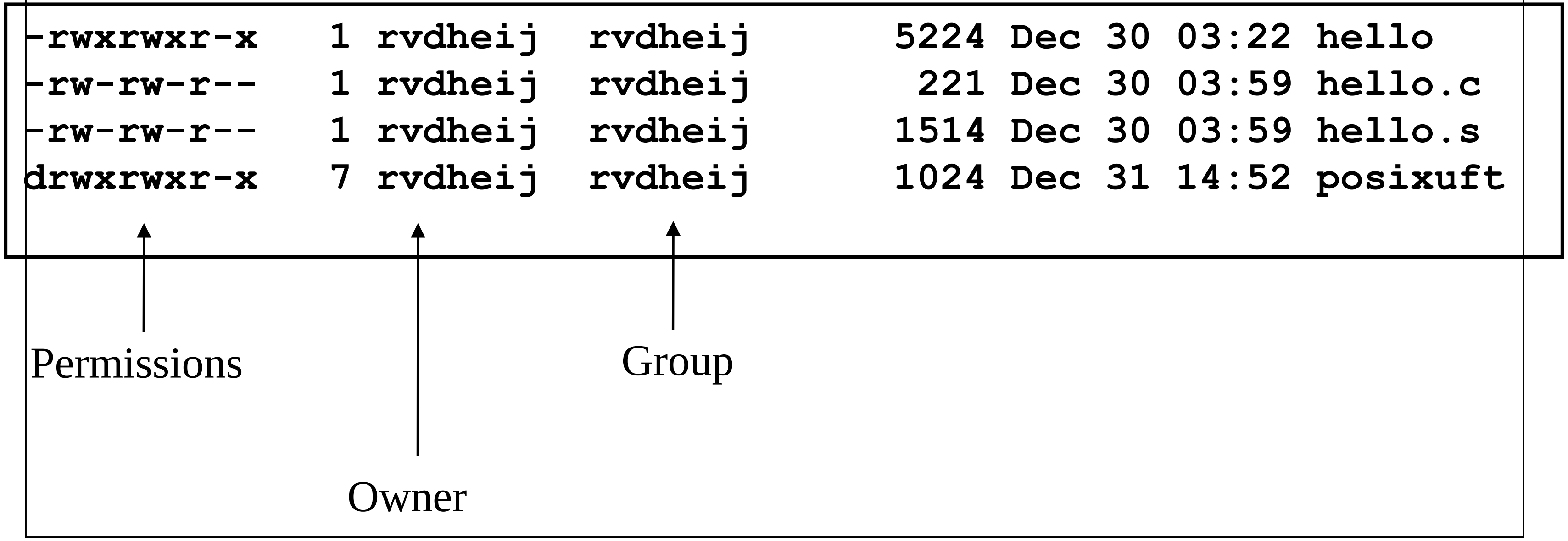
- Every user:
  - Has a uid (login name), gid (login group) and membership of a "groups" list:
    - The ***uid*** is who you are (name and number)
    - ***gid*** is your initial “login group” you normally belong to
    - The ***groups list*** is the file groups you can access via group permissions

File Permissions

- Linux provides 3 kinds of permissions:
  - **Read** - users with read permission may read the file or list the directory
  - **Write** - users with write permission may write to the file or new files to the directory
  - **Execute -** users with execute permission may execute the file or lookup a specific file within a directory

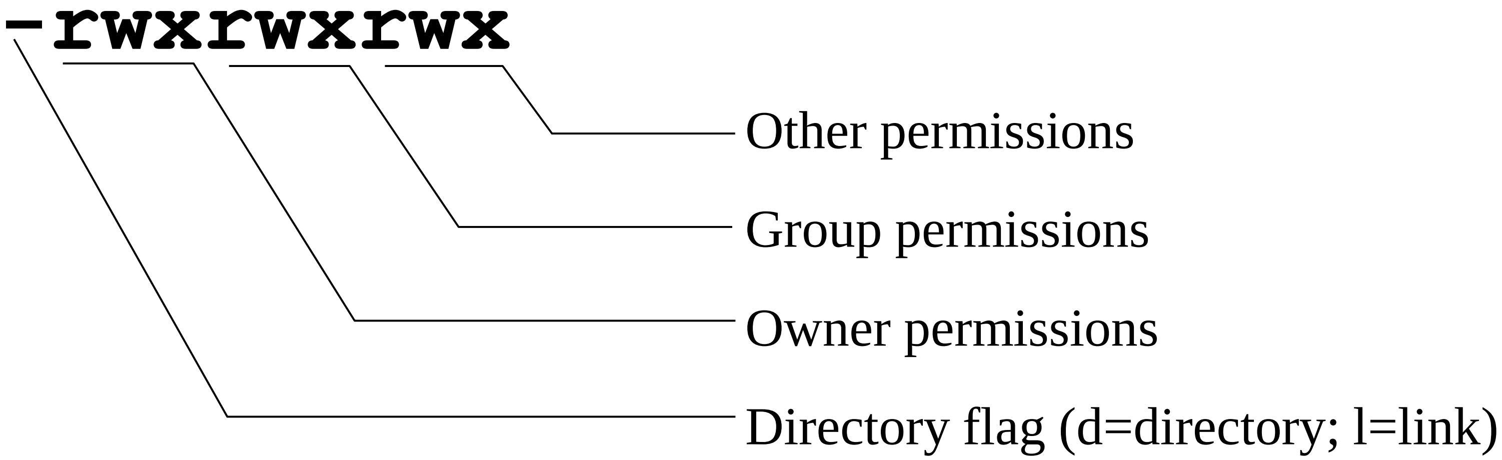
File Permissions

- The long version of a file listing (**`ls -l`**) will display the file permissions:



```
-rwxrwxr-x   1 rvdheij  rvdheij      5224 Dec 30 03:22 hello
-rw-rw-r--   1 rvdheij  rvdheij       221 Dec 30 03:59 hello.c
-rw-rw-r--   1 rvdheij  rvdheij      1514 Dec 30 03:59 hello.s
drwxrwxr-x   7 rvdheij  rvdheij      1024 Dec 31 14:52 posixuft
```

Interpreting File Permissions



```
-rwxrwxrwx
```

Other permissions

Group permissions

Owner permissions

Directory flag (d=directory; l=link)

Changing File Permissions

- Use `chmod` cmd to change file permissions
  - – Permissions are encoded as an octal number

```
chmod 755 file  # Owner=rwx Group=r-x Other=r-x
chmod 500 file2 # Owner=r-x Group=--- Other=---
chmod 644 file3 # Owner=rw- Group=r-- Other=r--

chmod +x  file  # Add execute permission to file for all
chmod o-r file  # Remove read permission for others
chmod a+w file  # Add write permission for everyone
```

Links?

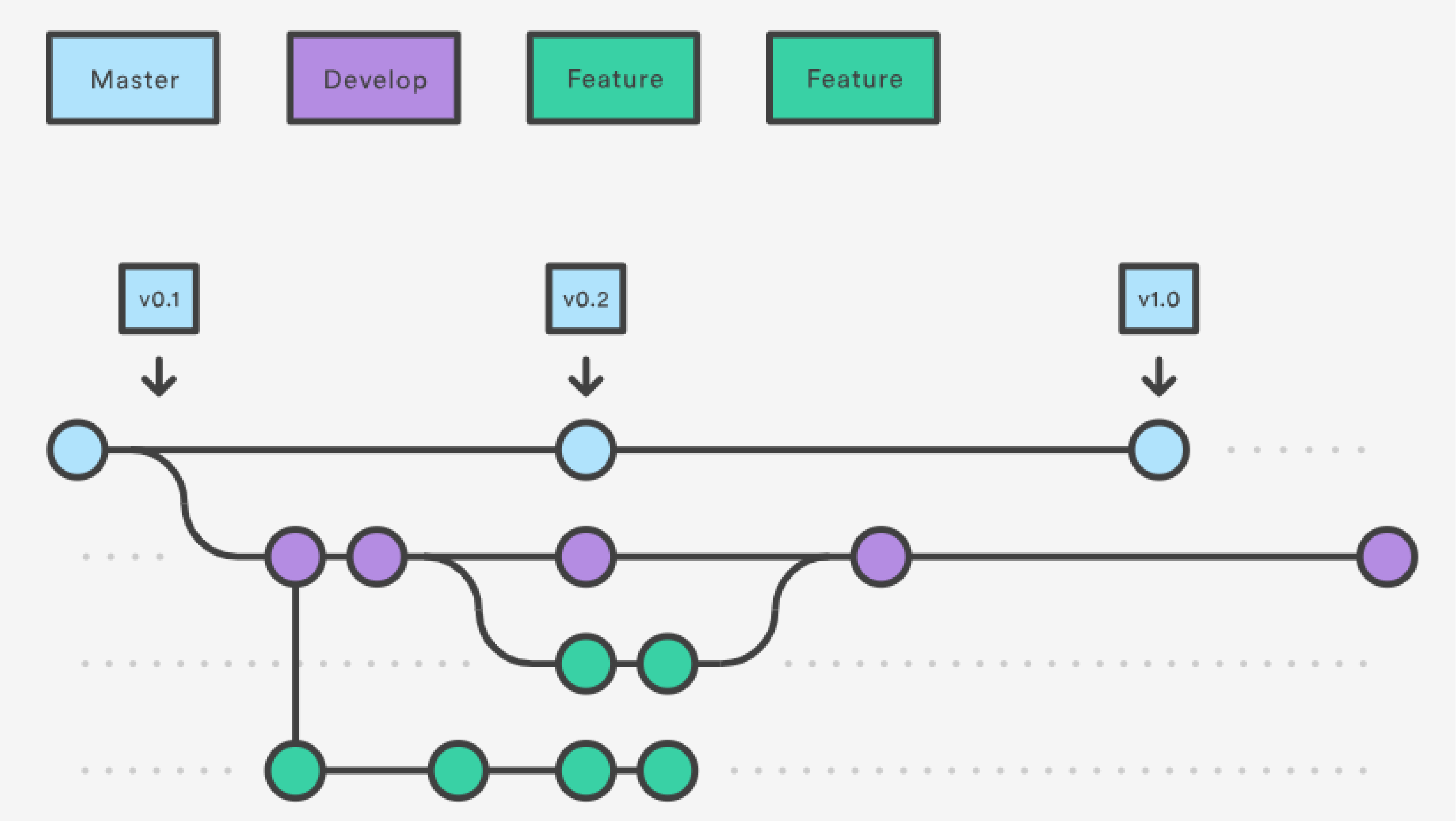
- Links are references to files (aliases)
- Two forms:
  - Hard
  - Symbolic
    - Can point to files on different physical devices
    - Delete of original leaves link
    - Delete of link leaves original
    - Can be created for directories
- Create using `ln` command

Editors

- People are fanatical about their editor
- Several choices available:

– `vi`	Standard UNIX editor
– `the`	XEDIT-like editor
– `xedit`	X windows text editor
– **`emacs`**	Extensible, Customizable Self-Documenting Display Editor
– **`pico`**	Simple display-oriented text editor
– **`nedit`**	X windows Motif text editor

Git: Version Control System



Version Control System

Software tools that help a software team manage changes to source code over time.

Version control software keeps track of every modification to the code in a special kind of database.

It records changes to a file or set of files over time so that you can recall specific **versions** later.

Git

Git is a distributed version-control system for tracking changes in source code during software development

GitHub

Bitbucket

GitLab

CSE Department hosts its own Git server using GitLab

Basic Git Commands

```
$ git init <project-name>
```

Creates a new local repository with the specified name

```
$ git config --global user.email abc@mail.com
```

Sets the email you want attached to your commit transactions

```
$ git config --global user.name “Your Name”
```

Sets the name you want attached to your commit transactions

```
$git add  --all
```

Snapshots the file in preparation for versioning

```
$git commit -m  “commit message”
```

Records file snapshots permanently in version history

```
$git push origin master
```

Uploads all local branch commits to GitHub

Thank you!