databased

The Missing Semester

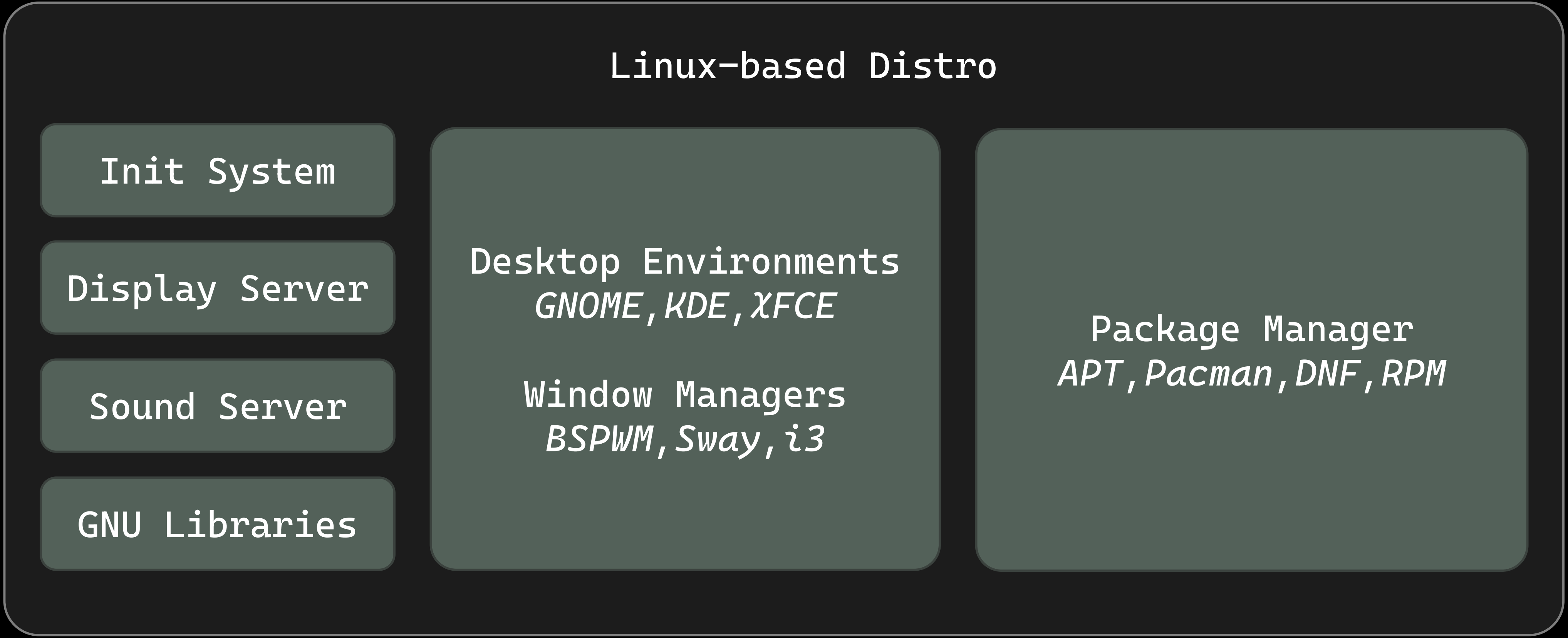
Introduction to Linux and Shells

Brought to you by Keval Pithadiya

Keval Pithadiya (kevalp@iisc.ac.in)

Linux and Linux-based Distros

- Linux in itself is only an Operating System Kernel.
- It needs to be bundled with other software to provide a complete user experience. Called a *distribution* or *distro*.



Keval Pithadiya (kevalp@iisc.ac.in)

Installing and Updating Software

- Distributions have a preinstalled package manager to allow for installing and managing software.
- These distributions manage “repositories” of a huge range of software packages. These maintain a list of installable software with either executable files or compilation scripts.
- Package managers are one of the major distinguishing factors of various distributions.

Debian-based

APT and dpkg

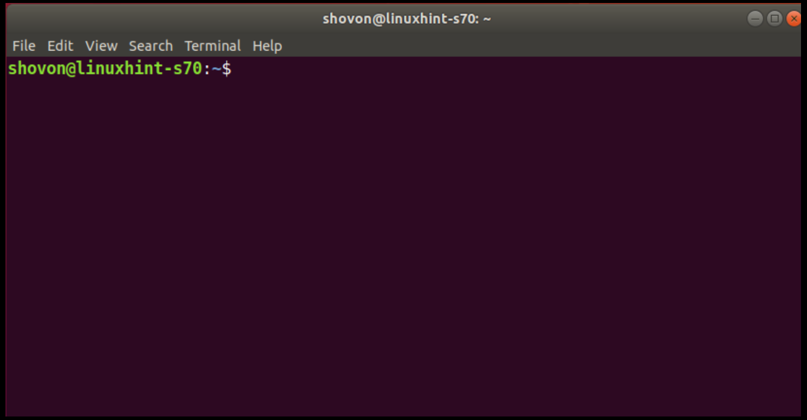
Arch-based

pacman

Keval Pithadiya (kevalp@iisc.ac.in)

Terminal

Actual Terminals



shovon@linuxhint-s70: ~

```
shovon@linuxhint-s70:~$
```

Terminal Emulators

Shell

Any software which allows you to interact with the operating system without writing your own code.

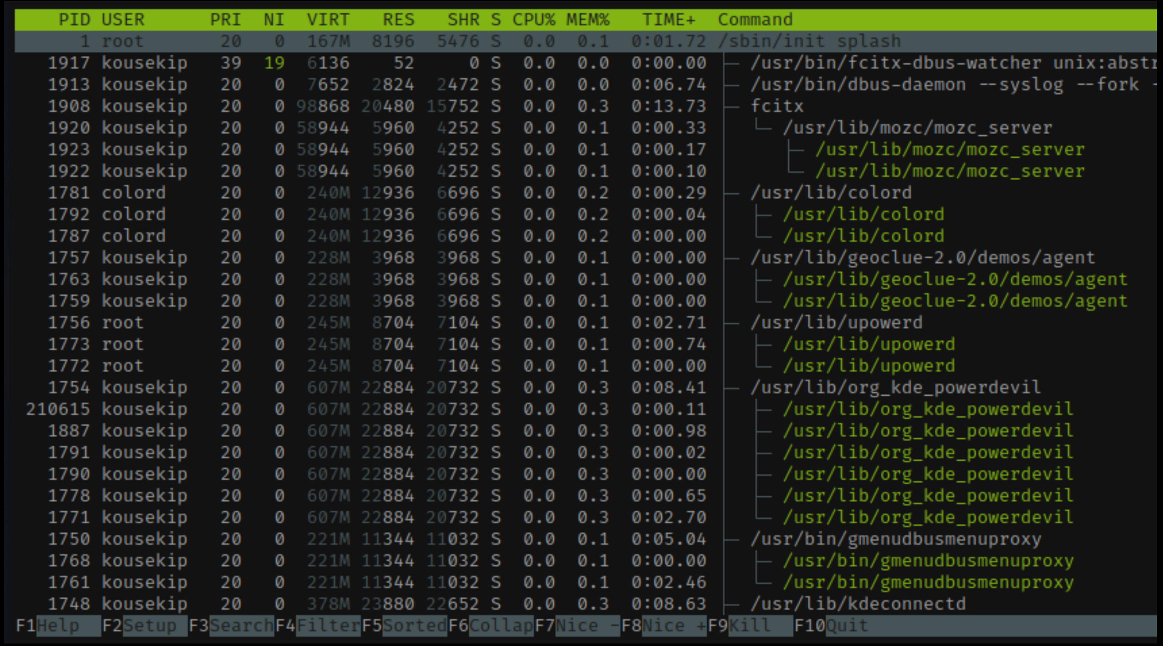
Some popular command-line shells:

- sh - Bourne Shell (developed for the UNIX system, found in Linux too)
- bash - Bourne-Again Shell (improved on sh, standard on Linux systems)
- zsh - Fancy extension of sh (default on Kali and MacOS)
- fish and many others

Keval Pithadiya (kevalp@iisc.ac.in)

Processes

- An instance of a program running on the computer.
- The operating system manages and stores a lot of book-keeping information about active processes to provide features like multitasking, security, and inter-process communication.
- Each process also has a bunch of context information stored in the form of environment variables which are inherited from the parent process or set explicitly.



```
    PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
      1 root       20   0  167M  8196  5476 S  0.0  0.1  0:01.72 /sbin/init splash
   1917 kousekip   39  19  6136    52     0 S  0.0  0.0  0:00.00 ├─ /usr/bin/fcitx-dbus-watcher unix:abstr
   1913 kousekip   20   0  7652  2824  2472 S  0.0  0.0  0:06.74 ├─ /usr/bin/dbus-daemon --syslog --fork -
   1908 kousekip   20   0 98868 20480 15752 S  0.0  0.3  0:13.73 ├─ fcitx
   1920 kousekip   20   0 58944  5960  4252 S  0.0  0.1  0:00.33 │  └─ /usr/lib/mozc/mozc_server
   1923 kousekip   20   0 58944  5960  4252 S  0.0  0.1  0:00.17 │     ├─ /usr/lib/mozc/mozc_server
   1922 kousekip   20   0 58944  5960  4252 S  0.0  0.1  0:00.10 │     └─ /usr/lib/mozc/mozc_server
   1781 colord     20   0  240M 12936  6696 S  0.0  0.2  0:00.29 ├─ /usr/lib/colord
   1792 colord     20   0  240M 12936  6696 S  0.0  0.2  0:00.04 │  ├─ /usr/lib/colord
   1787 colord     20   0  240M 12936  6696 S  0.0  0.2  0:00.00 │  └─ /usr/lib/colord
   1757 kousekip   20   0  228M  3968  3968 S  0.0  0.1  0:00.00 ├─ /usr/lib/geoclue-2.0/demos/agent
   1763 kousekip   20   0  228M  3968  3968 S  0.0  0.1  0:00.00 │  ├─ /usr/lib/geoclue-2.0/demos/agent
   1759 kousekip   20   0  228M  3968  3968 S  0.0  0.1  0:00.00 │  └─ /usr/lib/geoclue-2.0/demos/agent
   1756 root       20   0  245M  8704  7104 S  0.0  0.1  0:02.71 ├─ /usr/lib/upowerd
   1773 root       20   0  245M  8704  7104 S  0.0  0.1  0:00.74 │  ├─ /usr/lib/upowerd
   1772 root       20   0  245M  8704  7104 S  0.0  0.1  0:00.00 │  └─ /usr/lib/upowerd
   1754 kousekip   20   0  607M 22884 20732 S  0.0  0.3  0:08.41 ├─ /usr/lib/org_kde_powerdevil
 210615 kousekip   20   0  607M 22884 20732 S  0.0  0.3  0:00.11 │  ├─ /usr/lib/org_kde_powerdevil
   1887 kousekip   20   0  607M 22884 20732 S  0.0  0.3  0:00.98 │  ├─ /usr/lib/org_kde_powerdevil
   1791 kousekip   20   0  607M 22884 20732 S  0.0  0.3  0:00.02 │  ├─ /usr/lib/org_kde_powerdevil
   1790 kousekip   20   0  607M 22884 20732 S  0.0  0.3  0:00.00 │  ├─ /usr/lib/org_kde_powerdevil
   1778 kousekip   20   0  607M 22884 20732 S  0.0  0.3  0:00.65 │  ├─ /usr/lib/org_kde_powerdevil
   1771 kousekip   20   0  607M 22884 20732 S  0.0  0.3  0:02.70 │  └─ /usr/lib/org_kde_powerdevil
   1750 kousekip   20   0  221M 11344 11032 S  0.0  0.1  0:05.04 ├─ /usr/bin/gmenudbusmenuproxy
   1768 kousekip   20   0  221M 11344 11032 S  0.0  0.1  0:00.00 │  ├─ /usr/bin/gmenudbusmenuproxy
   1761 kousekip   20   0  221M 11344 11032 S  0.0  0.1  0:02.46 │  └─ /usr/bin/gmenudbusmenuproxy
   1748 kousekip   20   0  378M 23880 22652 S  0.0  0.3  0:08.63 ├─ /usr/lib/kdeconnectd
F1Help  F2Setup F3SearchF4FilterF5SortedF6CollapF7Nice -F8Nice +F9Kill  F10Quit
```

Keval Pithadiya (kevalp@iisc.ac.in)

Files

- In the UNIX philosophy, *everything is a file*.
- Everything from stored data files, executables, and folders to the entire RAM and connected hardware devices is a file.

- In a Linux system, each file has an owner user associated with it.
- Each file then has read, write and execute permissions for the owner user, owner group, and others.

- Executable files can be of two types:
  - Binary Executables in Machine Code
  - Scripts written in languages like shell script, Python, etc.

Working in a Shell

- Any shell session starts at a working directory location stored in the *$PWD* environment variable.
- The shell can be issued a variety of commands along with arguments.
- In general, shell commands can be of 3 types:
  - An in-built command
  - An executable which can be resolved by consulting the *$PATH*
  - An absolute or relative path which resolves to an executable
- Relative paths are resolved from the current working directory.
- Shells can be configured using a configuration file stored in the user’s home directory (e.g., ~/.bashrc).

Working in a Shell

Linux maintains a separate administrator user account (generally *root*) and an ordinary user account. By default, we access the system with the ordinary account.

However, some commands and operations require administrator permissions to run. For example, managing and installing packages, modifying system configuration files.

The *sudo* command allows us to run shell commands as the administrator.

Alternatively, you can start a shell session as the administrator by using the “*su* –” command.

The Working Directory

- *pwd* – Prints the current working directory
- *cd* – Changes the working directory to the given folder path
- *pushd* and *popd* – Can be used to quickly go back and forth between working directories
- *mkdir* – Creates a new directory with given name

Working with files

- *ls* – List files in the current or given directory
  - *-a*: Print all files (including hidden)
  - *-l*: Long listing format, prints with file details
  - *-R*: List subdirectories recursively
- *touch* – Creates a new empty file
- *file* – Determines file type
- *stat* – Display file status, info like size, permissions, inode number
- *chmod* – Changes file permissions
- *chown* – Changes file ownership
- *cat* – For concatenating files. Mostly used for reading the files.
- *less* – A better way to read files in a terminal with scrolling.

Shortcuts in Linux

Hard Links:

- Used to make a path point to some data stored in a storage device
- All files are initialized with 1 hard link
- The data is deleted after all hard links are deleted
- Can only be used within a filesystem and for regular files

Soft / Symbolic Links:

- Used to make a path point to some other path
- Rely on the pointed path existing, otherwise the link is broken
- Deleting a soft link does not affect the actual file data
- Can be used to point to any type of file (including other symlinks)

Command Execution

- Each process being executed has 3 files associated with it:
  - Standard Input or *stdin*
  - Standard Output or *stdout*
  - Standard Error or *stderr*
- The standard input can be written to by other programs or by piping some data into a command.
- The standard output and error is printed as the command output in the terminal.
- Shells allow piping of data to and from files and commands:
  - < : used to redirect data from a file to *stdin*
  - > : used to redirect data from *stdout* or *stderr* to a file
  - >> : same as > but appends to the file instead of overwriting
  - | : used to pipe the *stdout* or *stderr* of one command to *stdin* of another