***C*hapter 1: Introduction**

1. **What is an Operating System?** Page 2

Extended Machine:

Manage Resources

**1.1 Operating System Components** Page 3

Operating Systems – Kernel

Compilers

Editors (Text Editors)

Utilities

Shell

CLI

GUI

POSIX 1003.2 Shell Standard.

**1.2 Open Source:** Page 5

OSI

FSF

GNU

**1.3 A few words about Linux** Page 6

http://distrowatch.com/

**1.4 A Few Words about Files and Directories:** Page 6

File Names

File Types

ASCII files

NON-ASCII

A Hierarchical Structure

Root Directory

Home Directory

Current Working Directory

Absolute Path

Relative Path

**1.5 Getting Started with Linux (Debian)** Page 8

Several Introductory Linux commands

cd; ls

Directory structure of Linux

**1.6 On Line Manual Assistance: (man)** Page 10

Pipelining

The wc command:

ls –l |wc –l

**1.7 Exercise.**

Page 12

**1.0 What is an Operating System?**

Most texts define two functions of an operating system as:

* Act as an extended machine
* Act as a resource manager

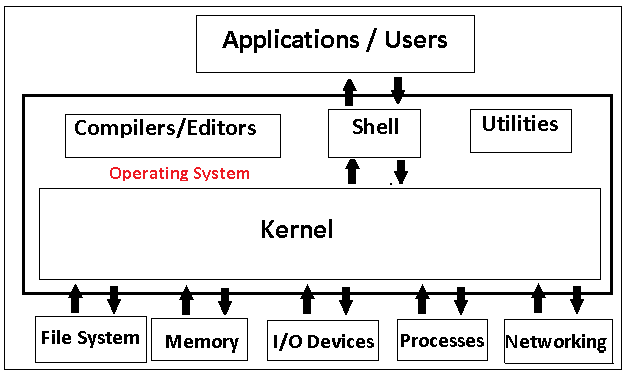
**Extended Machine:**

One of the functions of an operating system is to interface between the user or application and the resources. As the following diagram illustrates, an operating system can be thought of as consisting of two levels of software. One level consists of a set of tools consisting of compilers/editors, utilities and a shell. The second level is the kernel which interfaces with the resources “the hardware”.

**Manage Resources:**

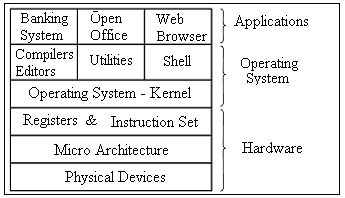
A second task of the operating system is to manage the resources. It is generally accepted that the resources managed are:

* Files and the file system (Disk Drives)
* Input/Output (I/O devices)
* Processes (Programs)
* Memory
* Networks



Another popular view of a computer system is:

* Hardware
* Operating Systems
* Application Programs



* 1. **Operating System Components**

In this course we concentrate on the “Operating Systems” layer in the above diagram.

Another view of a computer system would be:

**Operating Systems – Kernel**

The kernel is the name given to the program that does most of the housekeeping tasks for the operating system. The boot loader hands control over to the kernel, which initializes the various devices to a known state and makes the computer ready for general operation.

After preparing the hardware for use, the kernel provides an abstraction layer between the hardware and software running on the system. Through an API, or application programming interface, software can request the kernel to perform hardware tasks, as well as access many common functions. Most current kernels also multi-task the hardware, sharing it between the various processes running on the system.

**Compilers**

Compilers are programs that “translate” high-level programming languages into executable code. In your studies you will write programs in Java or C and then use a compiler to translate that code into machine executable code.

**Editors (Text Editors)**

Editors are programs that allow you to create “text” files in computers. Common text editors are notepad in windows and vi, pico, nano in linux. We discuss text editors in detail in chapter two of these notes.

**Utilities**

Utilities are programs that are included on computers that perform particular tasks. For example a computer system might include backup, compression or security utilities. Utilities are general purchased from “third party” vendors or can be open source.

**Shell – A Wikipedia Definition**

In computing, a shell is a piece of software that provides an interface for users. Typically, the term refers to an operating system shell which provides access to the services of a kernel. The name 'shell' originates from shells being an outer layer of interface between the user and the innards of the operating system (the kernel).

Shells generally fall into one of two categories: command line and graphical. Command line shells provide a command line interface (CLI) to the kernel, while graphical shells provide a graphical user interface (GUI). In either case, users interface with the kernel via shell.

**Shell - Windows**

We are all familiar with the windows GUI shell (windows explorer) and in later in the course we will investigate the windows command line interface (CLI).

**Shell –Unix**

In linux several command line interface shells are common. Some are listed here.

**sh** aka "Bourne" shell, written by Steve Bourne at AT&T Bell Labs for Unix V7 (1979). Small, simple, and (originally) very few internal commands, so it called external programs for even the simplest of tasks. It is always available on everything that looks vaguely like Unix.

**csh** The "C" shell. Written by Bill Joy at Berkeley (who went on to found Sun Microsystems). Many things in common with the Bourne shell, but many enhancements to improve interactive use. The internal commands used only in scripts are very different from "sh", and similar (by design) to the "C" language syntax.

**tcsh**  The "TC" shell. Freely available and based on "csh". It has many additional features to make interactive use more convenient.

**ksh** The "Korn" shell, written by David Korn of AT&T Bell Labs (now Lucent).

Written as a major upgrade to "sh" and compatible with it, it has many internal commands for the most frequently used functions. It also incorporates many of the features from tcsh which enhance interactive use (command line history recall etc.).

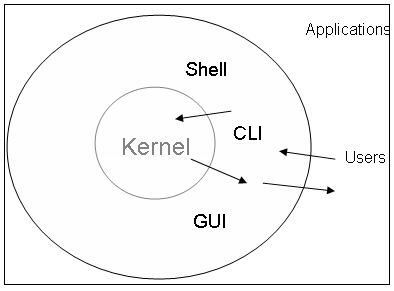
**bash**  The "Bourne again" shell was written as part of the GNU/Linux Open Source effort, and the default shell for Linux and Mac OS-X. It is a functional clone of sh, with additional features to enhance interactive use, add POSIX compliance, and partial ksh compatible.

**zsh** A freeware functional clone of sh, with parts of ksh, bash and full POSIX compliance, and many new interactive command-line editing features. It was installed as the default shell on early MacOSX systems.

**POSIX 1003.2 Shell Standard.**

POSIX (Portable Operating Systems Interface Exchange) is a set of standards established by IEEE that define various functions of operating systems. There are POSIX standards for writing shells, API’s and function calls to the kernel.

**Kernel & Shells (Another Look)**

****

**1.2 Open Source:**

Linux is an open source operating system. A google search on “open source” produced **187,000,000** “hits”. What does the term open source mean? Here, I present three major “players” in the Open Source game. I give definitions taken from the Web.

**GNU** is a computer operating system composed entirely of free software. Its name is a recursive acronym for *GNU's Not Unix*; it was chosen because its design is Unix-like, but differs from Unix by being free software and by not containing any Unix code. GNU was founded by Richard Stallman and was the original focus of the Free Software Foundation (FSF).

The project to develop GNU is known as the *GNU Project*, and programs released under the auspices of the GNU Project are called *GNU packages* or *GNU programs*. The system's basic components include the GNU Compiler Collection (GCC), the GNU Binary Utilities (binutils), the bash shell, the GNU C library (glibc), and GNU Core Utilities (coreutils).

**The Open Source Initiative (OSI)**

The Open Source Initiative (OSI) is a California public benefit corporation founded in 1998. The OSI are the stewards of the Open Source Definition (OSD) and the community-recognized body for reviewing and approving licenses as OSD-conformant. The OSI is actively involved in Open Source community-building and education. OSI Board members frequently travel the world to attend Open Source conferences and events, meet with open source developers and users, and to discuss with executives from the public and private sectors about how Open Source technologies, licenses, and models of development can provide economic and strategic advantages.

Open-source software (OSS) is computer software that is available in source code form: the source code and certain other rights normally reserved for copyright holders are provided under an open-source license that permits users to study, change, improve and at times also to distribute the software.

Open source software is very often developed in a public, collaborative manner. Open-source software is the most prominent example of open-source development and often compared to (technically defined) user-generated content or (legally defined) open content movements.

**The Free Software Foundation FSF**

The Free Software Foundation (FSF), established in 1985, is dedicated to promoting computer users' rights to use, study, copy, modify, and redistribute modified computer programs. The FSF promotes the development and use of free software, particularly the GNU operating system, used widely in its GNU/Linux variant. The FSF maintains the Free Software Definition.

**1.3 A few words about Linux**

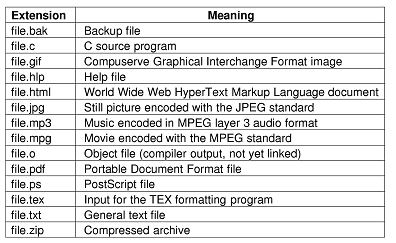
Linux is arguably the most well know “open source” operating system although there are others. Linux was started by Linus Torvalds and developed via the internet. Many “distributions of Linux exist. See **http://distrowatch.com/** for a ranking of the 100 most popular (today).

**1.4 A Few Words about Files and Directories:**

On a computer system everything is stored in files and files have data and filenames. Filenames generally consist of a string of characters followed by an optional extension. Some examples of filenames are:

**File Names:** Most operating systems support two-part file names separated by a period.

The following table gives examples of some popular file naming conventions.



**File Types:** There are ‘lots’ types of files in computing. At the simplest level, we can categorize files as being ASCII or NON-ASCII.

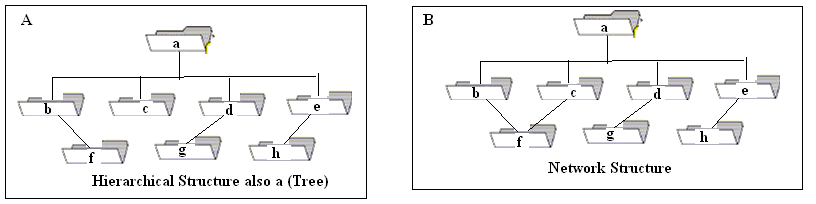
**ASCII files** are sometimes referred to as ordinary text files and are created with a “text editor”. We discuss text editors later in the course.

**NON-ASCII** files are all the “other files”. For example, when you save a file using Word, the text of the document is saved (ASCII) but also all formatting controls are also saved in the file. Similarly when you compile a Java Source File (ASCII) you get an executable object file (NON-ASCII). In this course we deal mainly with ASCII files.

As you will see, there are other ‘types’ of files that play a role in Linux/Unix. Some of the other file types which we will encounter are **character special files, block special files** and **linked files.**

**A Hierarchical Structure:**

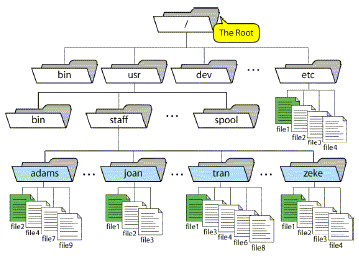
In order to understand the ‘structure’ of a disk you must understand what a hierarchical “tree” structure is. In a hierarchical structure there is only one play from any “node” to any other “node”. This differs from a “network” which allows for more than one path between nodes.



In the above, A is a hierarchical structure (only one path from any node to any other node) however, B is not a hierarchical structure since there are two paths from **node a** to **node f**.

**Hierarchical (Tree) Structure of a Disk:**

In computer disks resemble hierarchical structures. The following would be a typical structure of a Linux disk.

****

Notes:

* Every disk (partition) has a root directory. Top of the tree.
* There is a unique “path” from the root to any node in the hierarchy.
* A node can be either a directory or a file.
* Files are always at the bottom of the tree (the leaves)

**About Directories:** Here is some terminology that you need to be familiar with.

**Root Directory:** Every disk system have a root directory which Linux is / while in Microsoft is \.

**Home Directory:** Every user on a Linux/ Unix system is assigned a unique directory.

This is referred to as your home directory. In the above, if you name is adams,

the path to your home directory is **/usr/staff/adams**.

**Current Working Directory:** As you move around the file system you are always in a

directory. This directory is referred to the current working directory. The shell command **pwd** displays the absolute path to the current working directory. Think of **pwd** as **print working directory.**

**Absolute Path:** The “path” starting at the root, and declaring each directory to the

object being referenced. The path **/usr/staff/adams/file9** is the absolute path of the object **file9**.

**Relative Path:**The “path” is relative to current working directory. It **does not**

start with “/”. If the cwd (current working directory) is staff then **adams/file9** is the relative path to object **file9**.

**Your home directory** Every user has a unique ‘home’ directory. This directory was

created when you user account was established and it is your ‘default’ or ‘home’ directory.

**Several Introductory Commands:**

**>pwd** Gives the path to the Current Working Directory **(cwd)**

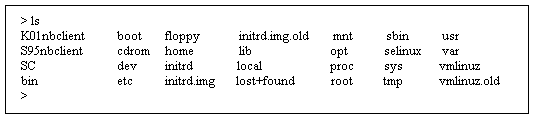
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**>cd** Change directory

Change directory to the root (/). **cd /**

**>ls** Show a directory listing. If you are in the root directory you will

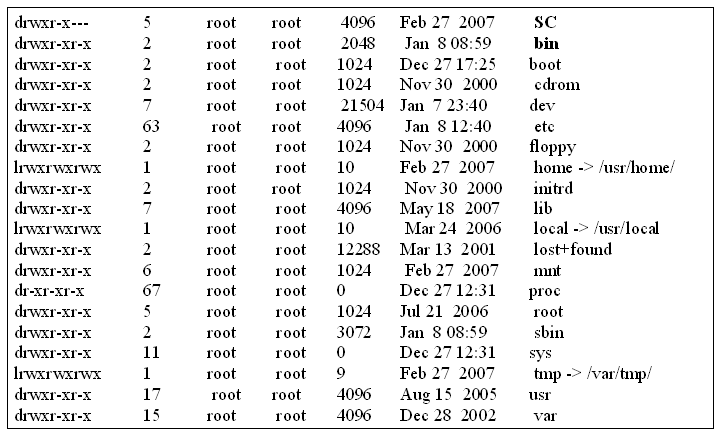
get a listing such as:

The entries in the root “/” directory will vary on the installation and the root 

The table below gives a more generic listing of a root directory.

| **Directory** | **Content** |
| --- | --- |
| /bin | Common programs, shared by the system, the system administrator and the users. |
| /boot | The startup files and the kernel, vmlinuz. In some recent distributions also grub data. Grub is the GRand Unified Boot loader and is an attempt to get rid of the many different boot-loaders we know today. |
| /dev | Contains references to all the CPU peripheral hardware, which are represented as files with special properties. |
| /etc | Most important system configuration files are in /etc, this directory contains data similar to those in the Control Panel in Windows |
| /home | Home directories of the common users. |
| /initrd | (on some distributions) Information for booting. Do not remove! |
| /lib | Library files, includes files for all kinds of programs needed by the system and the users. |
| /lost+found | Every partition has a lost+found in its upper directory. Files that were saved during failures are here. |
| /misc | For miscellaneous purposes. |
| /mnt | Standard mount point for external file systems, e.g. a CD-ROM or a digital camera. |
| /net | Standard mount point for entire remote file systems |
| /opt | Typically contains extra and third party software. |
| /proc | A virtual file system containing information about system resources. More information about the meaning of the files in proc is obtained by entering the command **man *proc*** in a terminal window. The file proc.txt discusses the virtual file system in detail. |
| /root | The administrative user's home directory. Mind the difference between /, the root directory and /root, the home directory of the *root* user. |
| /sbin | Programs for use by the system and the system administrator. |
| /tmp | Temporary space for use by the system, cleaned upon reboot, so don't use this for saving any work! |
| /usr | Programs, libraries, documentation etc. for all user-related programs. |
| /var | Storage for all variable files and temporary files created by users, such as log cd files, the mail queue, the print spooler area, space for temporary storage of files downloaded from the Internet, or to keep an image of a CD before burning it. |

>**ls –l** Get a long listing of a directory (partial listing given here).



**Notes:**

* The listing gives the same output but under different format.
* We discuss the meaning of each the above fields later in the course.

**1.6 On Line Manual Assistance: (man)**

There is an on-line manual which gives “help” for a specified command. Simply type

>**man command** and manual assistance with be given. The problem

is that if you are new to Linux these pages are hard to read but with practice you will get used to it.

**>man ls** The man command displays the on-line manual

assistance for the specified command.

Give sample output from man ls

For example type: **man ls** displays all options of the ls command. Several common options are give**n:**

**ls ls –l ls –a ls –la ls –R ls - -color=never**

**>pwd** Print working directory. This command displays

the absolute path to the current working directory.

**>ls –la Note the . and .. directory**

**What is the . directory? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**What is the .. directory? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Give the path to your home directory**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**>cd Change directory**

You can ‘move around’ the directories on the disk using the cd command.

**cd / moves to the root directory**

**pwd \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**cd /bin**

**Pipelining:** You can use the output from one command as input to another command using the | key. Note: This is the key just below the Backspace key (on my keyboard).

**>wc** useful counting command. (word count)

>**man wc**

**ls –l |wc –l** Will give you the number of lines of output of the **ls –l** command**.**

**Exercise**:

1. Give the path to your directory

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. How many entries in:

Your directory: ls –l|wc –l \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ls –la|wc –l \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is the significance of the:

. (dot) directory

.. (double dot- directory

1. Give the number of entries in each of the following directories.

Hint: Answer each of a two step problem.

* **First** move to the directory (using cd path)
* **Second** execute ls –l|wc -l

1. The root directory / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. /bin \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. /sbin \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
4. /usr \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
5. /usr/bin \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
6. /usr/sbin \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
7. How many users current logged onto atlas. The command **“who”** shows who is logged on.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ number

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ command used.

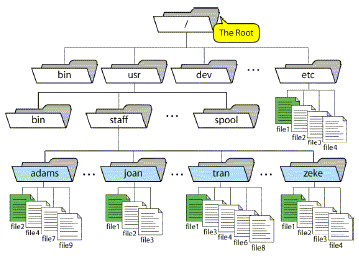
1. Using **cal** Use **man cal** to determine how to find which day of the week you were born.

On which day of the week were you born? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. How can you display a calendar of 2011 on the screen?

Give the command \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Given the following hierarchical structure.



1. Give the absolute path to the **file9**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. If the current working directory is **/usr/staff**, give the relative path to **file9**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_