LINUX a Multifunction Server

Ambika J, Sugantha Lakshmi R
Assistant Professor, Department of Computer Science & Engineering,
Kings College of Engineering, Thanjavur -613 303, India.

Abstract – An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs. The Linux open source operating system, or Linux OS, is a freely distributable, cross-platform operating system based on UNIX. The Linux consist of a kernel and some system programs. There are also some application programs for doing work. The kernel is the heart of the operating system which provides a set of tools that are used by system calls. In this paper we discussed the basic concepts and the system administration of Linux System.

Index Terms – Operating System, Kernel, Virtualization, DNS.

1. INTRODUCTION

A Linux-based system is a modular Unix-like operating system. It derives much of its basic design from principles established in UNIX. Such a system uses a monolithic kernel which handles process control, networking, and peripheral and file system access. Linux OS has the possess the following features

Portable - Portability means software can work on different types of hardware in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.

Open Source - Linux source code is freely available and it is community based development project.

Multi-User & **Multiprogramming** - Linux is a multiuser system where multiple users can access system resources like memory/ ram/ application programs at same time. Linux is a multiprogramming system means multiple applications can run at same time.

Hierarchical File System - Linux provides a standard file structure in which system files/ user files are arranged.

Shell - Linux provides a special interpreter program which can be used to execute commands of the operating system.

Security - Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

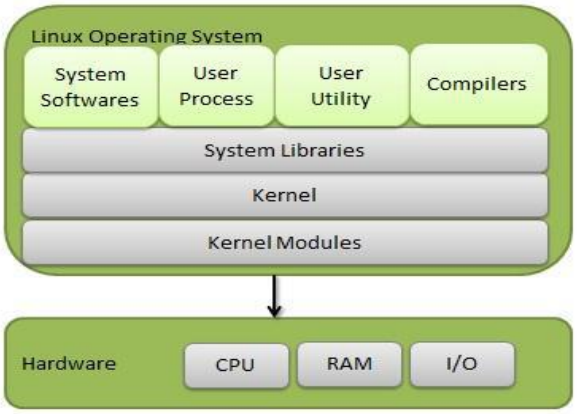
2. COMPONENTS OF LINUX SYSTEM

Linux Operating System has primarily three components

Kernel - Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It is consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.

System Library - System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implements most of the functionalities of the operating system and do not requires kernel module's code access rights.

System Utility - System Utility programs are responsible to do specialized, individual level tasks.



Installed components of a Linux system include the following:

A **bootloader** is a program that loads the Linux kernel into the computer's main memory, by being executed by the computer when it is turned on and after the firmware initialization is performed.

An **init** program is the first process launched by the Linux kernel, and is at the root of the process tree.

Software libraries, which contain code that can be used by running processes. The most commonly used software library on Linux systems, the GNU C Library (glibc), C standard library and Widget toolkits.

User interface programs such as command shells or windowing environments. The user interface, also known as the shell, is either a command-line interface (CLI), a graphical user interface (GUI), or through controls attached to the associated hardware.

3. BASIC CONCEPTS

All paragraphs must be indented. All paragraphs must be justified, i.e. both left-justified and right-justified.

A. System Architecture

The Linux System Architecture is consists of following layers

Hardware layer - Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).

Kernel - Core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.

Shell - An interface to kernel, hiding complexity of kernel's functions from users. Takes commands from user and executes kernel's functions.

Utilities - Utility programs giving user most of the functionalities of an operating systems

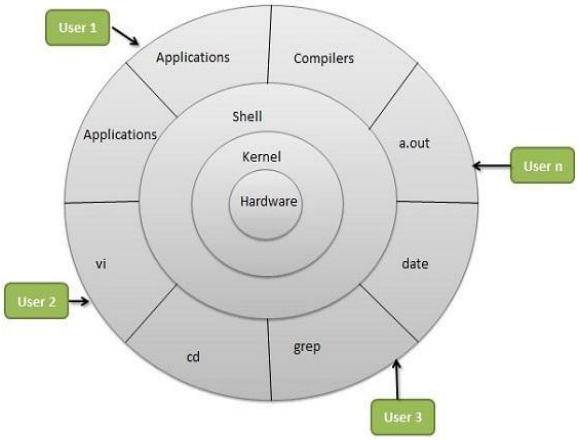


Fig. 2 System Architecture

B. Modes of Operation

No more than 3 levels of headings should be used.

1) *Kernel Mode*

Kernel component code executes in a special privileged mode called *kernel mode* with full access to all resources of the computer.

This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast.

Kernel runs each processes and provides system services to processes, provides protected access to hardware to processes.

User Mode:

The system programs use the tools provided by the kernel to implement the various services required from an operating system. System programs, and all other programs, run `on top of the kernel', in what is called the user mode.

Support code which is not required to run in kernel mode is in System Library.

User programs and other system programs work in User Mode which has no access to system hardware and kernel code.

User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

C. Services Provided by Linux

Initialization (init)

The single most important service in a LINUX system is provided by **init** program. The ***init*** is started as the first process of every LINUX system, as the last thing the kernel does when it boots. When init starts, it continues the boot process by doing various startup chores (checking and mounting file systems, starting daemons, etc).

Logins from terminals (getty)

Logins from terminals (via serial lines) and the console are provided by the **getty** program. **init** starts a separate instance of **getty** for each terminal upon which logins are to be allowed. Getty reads the username and runs the login program, which reads the password. If the username and password are correct, login runs the shell.

Logging and Auditing (syslog)

The kernel and many system programs produce error, warning, and other messages. It is often important that these messages can be viewed later, so they should be written to a file. The program doing this logging operation is known as **syslog**.

Periodic command execution (cron & at)

Both users and system administrators often need to run commands periodically. For example, the system administrator might want to run a command to clean the directories with temporary files from old files, to keep the disks from filling up, since not all programs clean up after themselves correctly.

The **cron** service is set up to do this. Each user can have a *crontab* file, where the lists the commands wish to execute and the times they should be executed.

The **at** service is similar to cron, but it is once only: the command is executed at the given time, but it is not repeated.

Graphical user interface

UNIX and Linux don't incorporate the user interface into the kernel; instead, they let it be implemented by user level programs. This applies for both text mode and graphical environments. This arrangement makes the system more flexible.

The graphical environment primarily used with Linux is called the X Window System (X for short) that provides tools with which a GUI can be implemented. Some popular window managers are blackbox and windowmaker. There are also two popular desktop managers, KDE and Gnome.

Network logins (telnet, rlogin & ssh)

Network logins work a little differently than normal logins. For each person logging in via the network there is a separate virtual network connection. It is therefore not possible to run a separate getty for each virtual connection. There are several different ways to log in via a network, **telnet** and **ssh** being the major ones in TCP/IP networks.

Most of Linux system administrators consider telnet and rlogin to be insecure and prefer ssh, the ``secure shell", which encrypts traffic going over the network, thereby making it far less likely that the malicious can ``sniff" the connection and gain sensitive data like usernames and passwords.

Network File System (NFS & CIFS)

One of the more useful things that can be done with networking services is sharing files via a network file system. Depending on your network this could be done over the Network File System (NFS), or over the Common Internet File System (CIFS).

NFS is typically a 'UNIX' based service. In Linux, NFS is supported by the kernel. CIFS however is not. In Linux, CIFS is supported by **Samba**. With a network file system any file operations done by a program on one machine are sent over the network to another computer.

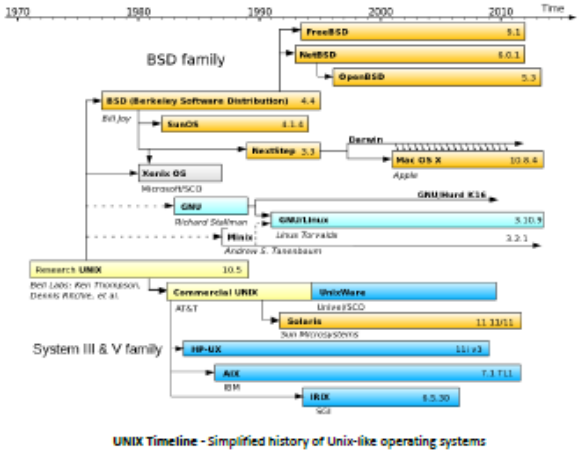


Fig. 3 Unix Timeline

D. System Administrator

A system administrator is a person who is responsible for the configuration and reliable operation of computer systems, especially multi-user computers, such as servers.

The system administrator seeks to ensure that the uptime, performance, resources, and security of the computers without exceeding the budget.

To meet these needs, a system administrator may acquire, install, or upgrade computer components and software, provide routine automation, maintain security policies AND troubleshoot.

1) Responsibilities of a Administrator

A system administrator's responsibilities might include:

1. Installing and configuring new hardware and software.
2. Applying operating system updates, patches, and configuration changes.
3. Analyzing system logs and identifying potential issues with computer systems.
4. Introducing and integrating new technologies into existing data center environments and configuring, adding, and deleting file systems.
5. Performing routine audit of systems and software.
6. Adding, removing, or updating user account information, resetting passwords, etc.
7. Responsibility for security and documenting the configuration of the system.
8. Troubleshooting any reported problems.
9. System performance tuning.

2) System Administrator Roles

In a larger company, these may all be separate positions within a computer support or Information Services (IS) department. In a smaller group they may be shared by a few sysadmins, or even a single person.

A **database administrator** (DBA) maintains a database system, and is responsible for the integrity of the data and the efficiency and performance of the system.

A **network administrator** maintains network infrastructure such as switches and routers, and diagnoses problems with these or with the behaviour of network-attached computers.

A **security administrator** is a specialist in computer and network security, including the administration of security devices such as firewalls, as well as consulting on general security measures.

A **web administrator** maintains web server services (such as Apache or IIS) that allow for internal or external access to web sites. Tasks include managing multiple sites, administering security, and configuring necessary components and software.

A **computer operator** performs routine maintenance and upkeep, such as changing backup tapes or replacing failed drives in a redundant array of independent disks (RAID).

A **postmaster** administers a mail server.

A **Storage Administrator (SAN)** can create, provision, add or remove Storage to/from Computer systems. Storage can be attached locally to the system or from a storage area network (SAN) or network-attached storage (NAS).

3) *Requirements for LINUX System Administrator*

1. While specific knowledge is a boon, system administrator should possess basic knowledge about all aspects of Linux. For example, a little knowledge about Solaris, BSD, nginx or various flavors of Linux.
2. Knowledge in at least one of the upper tier scripting language such as Python, Perl, Ruby or more.
3. To be a system administrator, he/she at least needs to have some hands-on experience of system management, system setup and managing Linux or Solaris based servers as well as configuring them.
4. Knowledge in shell programming such as Buorne or Korn and architecture.
5. Knowledge about storage technologies like FC, NFS or iSCSI is great, while knowledge regarding backup technologies is a must for a system administrator.
6. Knowledge in testing methodologies like Subversion or Git is great, while knowledge of version control is also an advantage.
7. Knowledge about basics of configuration management tools like Puppet and Chef.
8. Skills with system and application monitoring tools like SNMP or Nagios are also important, as they show your ability as an administrator in a team setting.
9. Knowledge about how to operate virtualized VMWare or Xen Server, Multifunction Server and Samba
10. An ITIL Foundation certification for Linux system administrator

E. DNS

sd Debian provides a stable version of BIND in its repositories. BIND can be installed, setup and secure it in a *chroot* environment, meaning it won't be able to see or access files outside its own directory tree. This is an important security technique.

The term *chroot* refers to the trick of changing the root filesystem (the /directory) that a process sees, so that most of the system is effectively inaccessible to it.

The BIND server also can be configured to run as a non-root user. That way, if someone gains access to BIND, he/she won't gain root privileges or be able to control other processes.

1. To install BIND on your Debian server, run this command:

apt-get install bind9

Debian downloads and configures the file as an Internet service and the status can be seen on the console:

Setting up bind9 (9.2.4-1)

Adding group `bind' (104) - Done.

Adding system user `bind'

Adding new user `bind' (104) with group `bind'.

Not creating home directory.

Starting domain name service: named.

2. To put BIND in a secured environment, create a directory where the service can run unexposed to other processes. First stop the service by running the following command: # /etc/init.d/bind9 stop

3. Edit the file /etc/default/bind9 so that the daemon will run as the unprivileged user bind, chrooted to

/var/lib/named. Change the line:

OPTS="-u bind"

So that it reads:

OPTIONS="-u bind -t /var/lib/named"

4. To provide a complete environment for running BIND, create the necessary directories under /var/lib:

 # mkdir -p /var/lib/named/etc

 # mkdir /var/lib/named/dev

 # mkdir -p /var/lib/named/var/cache/bind

 # mkdir -p /var/lib/named/var/run/bind/run

Then move the config directory from /etc to

/var/lib/named/etc:

 # mv /etc/bind /var/lib/named/etc

Next, create a symbolic link to the new config directory from the old location, to avoid problems when BIND is upgraded in the future:

 # ln -s /var/lib/named/etc/bind /etc/bind

Make null and random devices for use by BIND, and fix the permissions of the directories:

 # mknod /var/lib/named/dev/null c 1 3

 # mknod /var/lib/named/dev/random c 1 8

Then change permissions and ownership on the files:

 # chmod 666 /var/lib/named/dev/null

/var/lib/named/dev/random

 # chown -R bind:bind /var/lib/named/var/*

 # chown -R bind:bind /var/lib/named/etc/bind

5. Finally, start BIND:

 # /etc/init.d/bind9 start

6. To check whether named is functioning without any trouble. Execute this command:

server1:/home/admin# rndc status

number of zones: 6

debug level: 0

xfers running: 0

xfers deferred: 0

soa queries in progress: 0

query logging is OFF

4. VIRTUALIZATION

Virtualization refers to the act of creating a virtual (rather than actual) version of something, including a virtual computer hardware platform, operating system (OS), storage device, or computer network resources.

A. Benefits of Virtualization

1. Instead of deploying several physical servers for each service, only one server can be used. Virtualization let multiple OSs and applications to run on a server at a time. Consolidate hardware to get vastly higher productivity from fewer servers.
2. If the preferred operating system is deployed as an image, so we needed to go through the installation process only once for the entire infrastructure.
3. **Improve business continuity:** Virtual operating system images allow us for instant recovery in case of a system failure. The crashed system can be restored back by coping the virtual image.
4. **Increased uptime:** Most server virtualization platforms offer a number of advanced features that just aren't found on physical servers which increases servers' uptime. Some of features are live migration, storage migration, fault tolerance, high availability, and distributed resource scheduling.

B. Virtual Machine Server – A Layered Approach

Hardware virtualization or platform virtualization refers to the creation of a virtual machine that acts like a real computer with an operating system. Software executed on these virtual machines is separated from the underlying hardware resources.

Hardware virtualization hides the physical characteristics of a computing platform from users, instead showing another abstract computing platform.

For example, a computer that is running Microsoft Windows may host a virtual machine that looks like a computer with the Ubuntu Linux operating system; Ubuntu-based software can be run on the virtual machine.

5. CONCLUSION

A program that acts as an intermediary between a user of a computer and the computer hardware.Operating system goals:1. Execute user programs and make solving user problems easier. 2. Make the computer system convenient to use. 3. Use the computer hardware in an efficient manner. In this paper we could able to understand the basics of Linux system and perform administrative tasks on Linux Servers.

REFERENCES

[1] http://nptel.ac.in/courses/106108101/pdf/Lecture_Notes/Mod%201_LN.pdf
[2] http://codex.cs.yale.edu/avi/os-book/OS8/os8c/slide-dir/PDF-dir/ch13.pdf
[3] http://www.ics.uci.edu/~ics143/lectures/oslecture1.pdf
[4] http://axp.rnet.missouri.edu/cs4520/Slides/ch5-CPU_Scheduling-2.pdf
[5] http://www.tapavg.ee/server/OReilly.Linux.System.Administration.Mar.2007.pdf
[6] https://help.ubuntu.com/community/Xen