IT 316 Linux server and Virtualization

Having successfully completed the module, students should be able to demonstrate knowledge and understanding of:

i. Concepts in computer networking

ii. Elements of Unix Architecture

iii. Basic Linux commands, administration, file systems, command-line and Graphical tools for network configurations and package installations

iv. System and network services

v. Virtualization services vendors, Virtualization types, platforms and methods, virtualization applications

CHAP 0: Linux server and Virtualization

What is Linux?

Just like Windows XP, Windows 7, Windows 8, and Mac OS X, Linux is an operating system. An operating system is software that manages all of the hardware resources associated with your desktop or laptop. To put it simply – the operating system manages the communication between your software and your hardware. Without the operating system (often referred to as the "OS"), the software wouldn't function.

The OS is comprised of a number of pieces:

- **The Bootloader:** The software that manages the boot process of your computer.
- **The kernel:** This is the one piece of the whole that is actually called "Linux". The kernel is the core of the system and manages the CPU, memory, and peripheral devices. The kernel is the "lowest" level of the OS.
- **Daemons:** These are background services (printing, sound, scheduling, etc) that either start up during boot, or after you log into the desktop.
- **The Shell:** You've probably heard mention of the Linux command line. This is the shell – a command process that allows you to control the computer via commands typed into a text interface.
- **Graphical Server:** This is the sub-system that displays the graphics on your monitor. It is commonly referred to as the X server or just "X".
- **Desktop Environment:** This is the piece of the puzzle that the users actually interact with. There are many desktop environments to choose from (Unity, GNOME, Cinnamon, Enlightenment, KDE, XFCE, etc). Each desktop environment includes built-in applications (such as file managers, configuration tools, web browsers, games, etc).
- **Applications:** Desktop environments do not offer the full array of apps. Just like Windows and Mac, Linux offers thousands upon thousands of high-quality software titles that can be easily found and installed.

A **Linux server** is a high-powered variant of the Linux open source operating system that's designed to handle the more demanding needs of business applications such as network and system administration, database management and Web services.

Role of Linux servers

Linux servers are built to address the ever-increasing requirements of business applications like system and network administration, Web services and database management. Linux servers are often preferred over other server operating systems because of their reputation for security, consistency and flexibility.

What does Virtualization mean?

Virtualization refers to the creation of a virtual resource such as a server, desktop, operating system, file, storage or network.

Virtualization is the process of running multiple virtual machines on a single physical machine. The virtual machines share the resources of one physical computer, and each virtual machine is its own environment.

Virtualization can be categorized into different layers: **desktop, server, file, storage and network.**

Benefits of using virtualization
Server virtualization brings positive transformations, such as reduced hardware costs, improved server provisioning and deployment, better disaster recovery solutions, efficient and economic use of energy, and increased staff productivity.

Virtual servers can also be useful in consolidating the number of servers that a business uses. They can reduce the number of physical servers used by migrating applications and operating systems into virtual machines that are running under only one server.

Types of Virtualization

- OS Virtualization—aka Virtual Machines. Virtualizing an operating system environment is the most common form of virtualization.
- Application-Server Virtualization.
- Application Virtualization.
- Administrative Virtualization.
- Network Virtualization.
- Hardware Virtualization
- Storage Virtualization.

1. OS Virtualization—aka Virtual Machines
Virtualizing an operating system environment is the most common form of virtualization. It involves putting a second instance or multiple instances of an operating system, like Windows, on a single machine.

2**. Application-Server Virtualization**
Application-server virtualization is another large presence in the virtualization space and

has been around since the inception of the concept. It is often referred to as ‘advanced load balancing,’ as it spreads applications across servers, and servers across applications.

3. **Application Virtualization**

Application virtualization is often confused with application-server virtualization. What it means is that applications operate on computers as if they reside naturally on the hard drive, but instead are running on a server. The ability to use RAM and CPU to run the programs while storing them centrally on a server, like through Microsoft Terminal Services and cloud-based software, improves how software security updates are pushed, and how software is rolled out.

4. Administrative Virtualization

Administrative virtualization is one of the least-known forms of virtualization, likely due to the fact that it’s primarily used in data centers. The concept of administration, or ‘management,’ virtualization means segmented admin roles through group and user policies. For example, certain groups may have access to read specific servers, infrastructure, application files, and rules, but not to change them.

5. Network Virtualization

Network virtualization involves virtually managing IPs, and is accomplished through tools like routing tables, NICs, switches, and VLAN tags.

6. Hardware Virtualization

Hardware virtualization is one of the rarer forms of virtualization, and when simply explained it is similar to OS virtualization (it is, in fact, often required for OS virtualization). Except, instead of putting multiple software instances on a single machine, chunks of a machine are partitioned off to perform specific tasks.

7. Storage Virtualization

Storage virtualization is an array of servers that are managed by a virtual storage system. The servers aren’t aware of exactly where their data is, and instead function more like worker bees in a hive.

CHAP I: Elements of Unix Architecture

What is Unix

The Unix operating system is a set of programs that act as a link between the computer and the user.

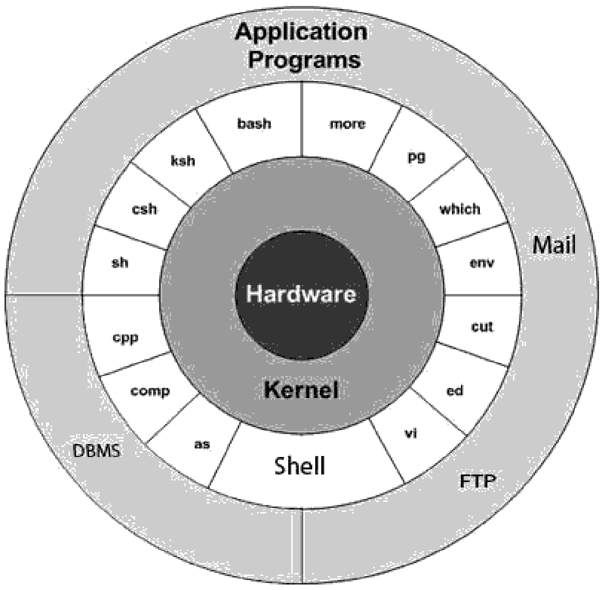
The computer programs that allocate the system resources and coordinate all the details of the computer's internals is called the **operating system** or the **kernel**.

Users communicate with the kernel through a program known as the **shell**. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

- Unix was originally developed in 1969 by a group of AT&T employees Ken Thompson, Dennis Ritchie, Douglas McIlroy, and Joe Ossanna at Bell Labs.
- There are various Unix variants available in the market. Solaris Unix, AIX, HP Unix and BSD are a few examples. Linux is also a flavor of Unix which is freely available.
- Several people can use a Unix computer at the same time; hence Unix is called a multiuser system.
- A user can also run multiple programs at the same time; hence Unix is a multitasking environment.

Unix Architecture

Here is a basic block diagram of a Unix system –



The main concept that unites all the versions of Unix is the following four basics −

- **Kernel** − The kernel is the heart of the operating system. It interacts with the hardware and most of the tasks like memory management, task scheduling and file management.
- **Shell** − The shell is the utility that processes your requests. When you type in a command at your terminal, the shell interprets the command and calls the program that you want. The shell uses standard syntax for all commands. C Shell, Bourne Shell and Korn Shell are the most famous shells which are available with most of the Unix variants.
- **Commands and Utilities** − There are various commands and utilities which you can make use of in your day to day activities. **cp**, **mv**, **cat** and **grep**, etc. are few examples of commands and utilities. There are over 250 standard commands plus numerous others provided through 3rd party software. All the commands come along with various options.
- **Files and Directories** − All the data of Unix is organized into files. All files are then organized into directories. These directories are further organized into a tree-like structure called the **filesystem**.

INSTALL UBUNTU

Minimum requirement for install ubuntu server

Requirements

- 2 GHz dual core processor or better
- 2 GB system memory
- 25 GB of free hard drive space
- Either a DVD drive or a USB port for the installer media
- If you're going to install Ubuntu Server alongside data you wish to keep, ensure you have a recent backup.
- Internet access is helpful

Boot from install media

To trigger the installation process, perform the following:

- Put the Ubuntu DVD into your DVD drive (or insert the USB stick or other install media).
- Restart your computer.

Choose your language

After the boot messages appear, a ‘Language' menu will be displayed.

Choose the correct keyboard layout

Before you need to type anything in, the installer will next display a menu asking you to select your keyboard layout and, if applicable, the variant.

Choose your install

Now we are ready to select what you want to install. There are three options in the menu:

*Install ubuntu

Metal As A Service (MAAS)

Networking

The installer will automatically detect and try to configure any network connections via DHCP.

If no network is found, the installer can continue anyway, it just won't be able to check for updates. You can always configure networking after installation.

Configure storage

The next step is to configure storage. The recommended install is to have an entire disk or partition set aside for running Ubuntu.

If you need to set up a more complicated system, the manual option will allow you to select and reorganize partitions on any connected drives.

Note that Ubuntu no longer *requires* a separate partition for swap space, nor will the automated install create one.

Select a device

This menu will allow you to select a disk from the ones detected on the system.

To help identification, the drives will be listed using their system ID. Use the arrow keys and enter to select the disk you wish to use.

Confirm partitions

With the target drive selected, the installer will calculate what partitions to create and present this information...

If this isn't what you expected to see (e.g., you have selected the wrong drive), you should use the arrow keys and enter to select Back from the options at the bottom of the screen. This will take you back to the previous menu where you can select a different drive.

It is also possible to manually change the partitions here, by selecting Edit Partitions. Obviously, you should only select this if you are familiar with how partitions work.

When you are happy with the disk layout displayed, select Done to continue.

Confirm changes

Before the installer makes any destructive changes, it will show this final confirmation step. Double check that everything looks good here and you aren't about to reformat the wrong device!

Set up a Profile

The software is now being installed on the disk, but there is some more information the installer needs. Ubuntu Server needs to have at least one known user for the system, and a hostname. The user also needs a password.

install software

Once you have finished entering the required information, the screen will now show the progress of the installer. Ubuntu Server now installs a concise set of useful software required for servers. This cuts down on the install and setup time dramatically. Of course, after the install is finished, you can install any additional software you may need.

Installation complete

When the install is complete, you will see a message like this on the screen.

- A swap file (or swap space or, in Windows NT, a pagefile) is a space on a hard disk used as the virtual memory extension of a computer's real memory (RAM). Having a swap file allows your computer's operating system to pretend that you have more RAM than you actually do.

- root is the user name or account that by default has access to all commands and files on a Linux or other Unix-like operating system. It is also referred to as the root account, root user and the superuser.

- The boot partition is a primary partition that contains the boot loader, a piece of software responsible for booting the operating system. For example, in the standard Linux directory layout (Filesystem Hierarchy Standard), boot files (such as the kernel, initrd, and boot loader GRUB) are mounted at /boot/

- Ubuntu generally creates just two partitions: root and swap. The main reason for having a home partition is to separate your user files and configuration files from the operating system files.

System Bootup

If you have a computer which has the Unix operating system installed in it, then you simply need to turn on the system to make it live.

As soon as you turn on the system, it starts booting up and finally it prompts you to log into the system, which is an activity to log into the system and use it for your day-to-day activities.

Login Unix

When you first connect to a Unix system, you usually see a prompt such as the following −

```
login:
```

To log in

- Have your userid (user identification) and password ready. Contact your system administrator if you don't have these yet.

- Type your userid at the login prompt, then press **ENTER**. Your userid is **case-sensitive**, so be sure you type it exactly as your system administrator has instructed.

- Type your password at the password prompt, then press **ENTER**. Your password is also case-sensitive.

- If you provide the correct userid and password, then you will be allowed to enter into the system. Read the information and messages that comes up on the screen, which is as follows.

```
login : amrood
amrood's password:
Last login: Sun Jun 14 09:32:32 2009 from 62.61.164.73
$
```

You will be provided with a command prompt (sometime called the **$** prompt) where you type all your commands. For example, to check calendar, you need to type the **cal** command as follows −

```
$ cal
```

```
     June 2009
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30

$
```

Change Password

All Unix systems require passwords to help ensure that your files and data remain your own and that the system itself is secure from hackers and crackers. Following are the steps to change your password −

Step 1 − To start, type password at the command prompt as shown below.

Step 2 − Enter your old password, the one you're currently using.

Step 3 − Type in your new password. Always keep your password complex enough so that nobody can guess it. But make sure, you remember it.

Step 4 − You must verify the password by typing it again.

```
$ passwd
Changing password for amrood
(current) Unix password:******
New UNIX password:*******
Retype new UNIX password:*******
passwd: all authentication tokens updated  successfully

$
```

Note − We have added asterisk (*) here just to show the location where you need to enter the current and new passwords otherwise at your system. It does not show you any character when you type.

Listing Directories and Files

All data in Unix is organized into files. All files are organized into directories. These directories are organized into a tree-like structure called the filesystem.

You can use the **ls** command to list out all the files or directories available in a directory. Following is the example of using **ls** command with **-l** option.

```
$ ls -l
total 19621
drwxrwxr-x  2 amrood amrood      4096 Dec 25 09:59 uml
-rw-rw-r--  1 amrood amrood      5341 Dec 25 08:38 uml.jpg
drwxr-xr-x  2 amrood amrood      4096 Feb 15  2006 univ
drwxr-xr-x  2 root   root        4096 Dec  9  2007 urlspedia
-rw-r--r--  1 root   root      276480 Dec  9  2007 urlspedia.tar
drwxr-xr-x  8 root   root        4096 Nov 25  2007 usr
-rwxr-xr-x  1 root   root        3192 Nov 25  2007 webthumb.php
-rw-rw-r--  1 amrood amrood     20480 Nov 25  2007 webthumb.tar
```

```
-rw-rw-r--  1 amrood amrood      5654 Aug  9  2007 yourfile.mid
-rw-rw-r--  1 amrood amrood    166255 Aug  9  2007 yourfile.swf

$
```

Here entries starting with **d.....** represent directories. For example, uml, univ and urlspedia are directories and rest of the entries are files.

Who Are You?

While you're logged into the system, you might be willing to know : **Who am I**?

The easiest way to find out "who you are" is to enter the **whoami** command −

```
$ whoami
 amrood

$
```

Try it on your system. This command lists the account name associated with the current login. You can try **who am i** command as well to get information about yourself.

Who is Logged in?

Sometime you might be interested to know who is logged in to the computer at the same time.

There are three commands available to get you this information, based on how much you wish to know about the other users: **users**, **who**, and **w**.

```
$ users
 amrood bablu qadir

$ who
amrood ttyp0 Oct 8 14:10 (limbo)
bablu  ttyp2 Oct 4 09:08 (calliope)
qadir  ttyp4 Oct 8 12:09 (dent)

$
```

Try the **w** command on your system to check the output. This lists down information associated with the users logged in the system.

Logging Out

When you finish your session, you need to log out of the system. This is to ensure that nobody else accesses your files.

To log out

- Just type the **logout** command at the command prompt, and the system will clean up everything and break the connection.

System Shutdown

The most consistent way to shut down a Unix system properly via the command line is to use one of the following commands −

Sr.No.	Command & Description
1	**halt** Brings the system down immediately
2	**init 0** Powers off the system using predefined scripts to synchronize and clean up the system prior to shutting down
3	**init 6** Reboots the system by shutting it down completely and then restarting it
4	**poweroff** Shuts down the system by powering off
5	**reboot** Reboots the system
6	**shutdown** Shuts down the system

You typically need to be the super user or root (the most privileged account on a Unix system) to shut down the system. However, on some standalone or personally-owned Unix boxes, an administrative user and sometimes regular users can do so.

Basic Ubuntu Commands

1. sudo

sudo (SuperUser DO) Linux command allows you to run programs or other commands with administrative privileges, just like “Run as administrator” in Windows. This is useful when, for example, you need to modify files in a directory that your user wouldn’t normally have access to.

2. apt-get

apt-get is the one of the most important Ubuntu commands every beginner must know. It is used to install, update, upgrade and remove any package. apt-get basically works on a database of available packages. Here is the list of different apt-get commands:

- *sudo apt-get update*

apt-get update with super user privileges is the first command you need to run in any Linux system after a fresh install. This command updates the database and let your system know if there are newer packages available or not.

- *sudo apt-get upgrade*

After updating the package database, next step is to to upgrade the installed packages. For upgrading all the packages with available updates you can use this command.

And if you like to upgrade a particular package, the you should tweak the above command a little:

sudo apt-get upgrade <package-name>. Replace the <package-name> with your desired package.

- *sudo apt-get install*

If you know the name of the package, then you can easily install a program using this command:

sudo apt-get install <package-name>. Replace the <package-name> with your desired package.

If you are not sure about the package name, the you can type a few letters and press tab and it will suggest all the packages available with those letters. Thanks for auto-completion feature.

3. ls

ls (list) command lists all files and folders in your current working directory. You can also specify paths to other directories if you want to view their contents.

4. cd

cd (change director”) Linux command also known as chdir used to change the current working directory. It’s one of the most used basic Ubuntu commands. Using this command is easy, just type cd followed by the the folder name. You can use full paths to folders or simply the name of a folder within the directory you are currently working. Some common uses are:

- cd / – Takes you to the root directory.
- cd .. – Takes you up one directory level.
- cd – – Takes you to the previous directory.

Here are some examples to how to use cd command in Ubuntu:

Example 1: ***cd home*** – open home folder in current directory.

Example 2: ***cd Linux\ Drive*** – open Linux Drive named folder in directory. Here you can see I use backslash because the folder name has spaces so *for each space you use “backslash+space”*. Like, if your folder name is “am a programmer” then the cd command will be, “cd am\ a\ programmer”.

5. pwd

pwd (print working directory) Ubuntu command displays the full pathname of the current working directory.

6. cp

cp (copy) Linux command allows you to copy a file. You should specify both the file you want to be copied and the location you want it copied to – f*or example, cp* xyz */home/*myfiles *would copy the file “*xyz*” to the directory “/home/*myfiles*”*.

7. mv

mv (move) command allows you to move files. You can also rename files by moving them to the directory they are currently in, but under a new name. The usage is the same as cp – f*or example mv* xyz */home/*myfiles *would move the file "*xyz*" to the directory "/home/*myfiles*"*.

8. rm

rm (remove) command removes the specified file.

- rmdir ("remove directory") – Removes an empty directory.
- rm -r ("remove recursively") – Removes a directory along with its content.

9. mkdir

mkdir (make directory) command allows you to create a new directory. You can specify where you want the directory created – if you do not do so, it will be created in your current working directory.

10. history

history command displays all of your previous commands up to the history limit.

11. df

df (display filesystem) command displays information about the disk space usage of all mounted filesystems.

12. du

du (directory usage) command displays the size of a directory and all of its subdirectories.

13. free

free – Displays the amount of free space available on the system.

14. uname -a

uname -a – Provides a wide range of basic information about the system.

15. top

top – Displays the processes using the most system resources at any given time. "q" can be used to exit.

16. man

man command displays a “manual page”. Manual pages are usually very detailed, and it’s recommended that you read the man pages for any command you are unfamiliar with. Some uses are :

- man man – Provides information about the manual itself.
- man intro – Displays a brief introduction to Linux commands.

17. info

Similar to man, but often provides more detailed or precise information.

18. <command name> -h or <command name> –help

This command is a third alternative to get help. While not as detailed as the info or man pages, this will provide a quick overview of the command and its uses.