

Week1 Lab01b Notes

The Boot Process

Booting is the process of bringing the system from an *off* status to a running operating system. It is very important to understand this process, in order to be able to fix boot problems when they occur and for modifying system states called *runlevels* or on **systemd** devices, *targets*.

The boot process takes place in four main stages, some of which are modified by administrators, while for the others, it is sufficient just to be aware of the actions taking place:

- Firmware Stage(BIOS)
- Bootloader
- Kernel Stage
- Init Stage

The firmware stage is the first stage to take place after the computer is powered on. Most PC firmware is referred to as the **Basic Input/Output System (BIOS)**. The BIOS is stored on the motherboard in non-volatile memory such as Read Only Memory (ROM) or flash memory. The **BIOS** has three main jobs to perform as part of the first stage of the boot process:

- Execute a *power-on self test (POST)* in order to ensure the hardware of the system is functioning properly. The POST runs some basic functional checks on the CPU, memory, and peripherals so that obvious errors, such as missing memory chips or malfunctioning disk devices are found early in the boot cycle.
- Enumerate available hardware such as memory, disks, and USB devices.
- Find the proper boot drive from the available storage devices and load the **Master Boot Record (MBR)** from that device. The Master Boot Record is the first sector (or 512 bytes) of the disk.

The **bootloader**'s job is to give you a choice (if configured) of options to load one or more versions of Linux, or even other operating systems, and then to load the kernel of the chosen option and get it started. The two bootloaders most commonly used with Linux are the **Grand Unified Bootloader (GRUB)**, or the **Grand Unified Bootloader 2 (GRUB 2)**. GRUB is a GNU project with the goal of making a bootloader that can boot many different kernels on a variety of systems. GRUB can read filesystems and dynamically learn about the hardware. The kernel that the bootloader is trying to run could be a Linux kernel, it could be a Microsoft Windows image, or it could be a bootable CD. The bootloader can also pass parameters to the kernel, such as to boot into a maintenance mode or to enable or disable certain hardware. This is done by manipulating the bootloader configuration.

The **kernel** dictates which program gets which pieces of memory, it starts and kills programs, it interprets instructions given to it by the user, and it handles more common and simple tasks such as displaying text on a monitor. The **kernel** must initialize any hardware drivers and get the root / filesystem mounted for the next stage. This kernel typically lives in the */boot* partition which, on most hardware, is in a separate partition that's kept close to the beginning of the hard drive. This location is important for some BIOS and bootloader combinations that can only access the first 1024 cylinders of the disk. The */boot* directory, alternatively the */boot/efi* directory, is one of the most

important directories in the File Hierarchy Standard (FHS). The kernel's final job is to start the first process on the system. The first process will normally have a process id (PID) of 1; on a System V system, the name of this process is **init**.

The **init** process has three important responsibilities:

- Continue the booting process to get services running, login screens displaying, and consoles listening.
- Start all other system processes.
- Adopt any process that detaches from its parent.

Until recently, this process followed a design that was established with the release of **System V** of Unix, which is sometimes referred to as **SysVinit**. The actual process that is executed is the **init** process. Recently, other programs have emerged to compete with and replace the traditional **init** process with **Upstart** and **systemd**.

If the system uses the traditional **init** program, then the `/etc/inittab` file is used to determine what scripts will be executed to start the services that will be available on the system.

The `inittab` file points to other scripts that do the work, usually stored in the `/etc/init.d` directory. In general, each service the system will run has a script that can start, stop, and restart a service, and **init** will cause each of these to be run in turn as needed to get the system to the desired state.

If the traditional **init** has been replaced with Upstart, the scripts in the `/etc/init` directory are used to complete system initialization.

If the traditional **init** has been replaced with Systemd, then the files in the `/etc/systemd` directory are used for starting up and running the system.

Kernel Messages

The `dmesg` command can be executed after booting the system to see the messages generated by the kernel during boot time.

Kernel messages are stored in a ring buffer of limited size; therefore, the messages that are generated at boot time may be overwritten later as the buffer fills up. It is common to execute the `dmesg` command upon connecting a new device to the system. This allows the administrator to see how the kernel dealt with the new device and usually to see what path name the new device has been assigned.

Kernel messages and other system-related messages are typically stored in the `/var/log/messages` file. This file, which is considered the main log file, is alternatively named `/var/log/syslog` in some distributions.

On a systemd-based system, the `journald` daemon is the logging mechanism, and it's configured by the `/etc/systemd/journald.conf` file.

The `journald` log files are viewed with the `journalctl` command,

Boot Target

Linux uses the concept of different runlevels to define what services or processes will be running. Although the Linux kernel can recognize runlevel values from 0 to 9, typically only runlevels 0 through 6 are used. Traditionally, `init` and **Upstart** used these runlevels to define which services were started according to the needs of a particular runlevel.

Recently, these programs have been replaced on many distributions by **systemd**, a service and system manager originally designed by Red Hat. It does something similar to runlevels called targets.

Runlevel	Purpose	systemd Target
0	Halt or shut off the system	<code>poweroff.target</code>
1	Single-user mode for administrative tasks	<code>rescue.target</code>
2	Multi-user mode without configured network interfaces or network services	<code>multi-user.target</code>
3	Normal startup of the system	<code>multi-user.target</code>
4	User-definable	<code>multi-user.target</code>
5	Start the system normally with a graphical display manager	<code>graphical.target</code>
6	Restart the system	<code>reboot.target</code>

To check the current runlevel on a Linux system, list the <code>/etc/systemd/system/default.target</code> file.	<pre>[root@localhost ~]# ls -l /etc/systemd/system/default.target lrwxrwxrwx 1 root root 37 Dec 4 14:39 /etc/systemd/system/default.target ->/lib/systemd/system/multi-user.target</pre>
to set the system to boot into single-user mode for	<pre>[root@localhost ~]# systemctl enable rescue.target</pre>

troubleshooting or recovery operations	<pre>Created symlink /etc/systemd/system/kbrequest.target, pointing to /usr/lib/systemd/system/rescue.target. [root@localhost ~]# systemctl set-default rescue.target Removed /etc/systemd/system/default.target. Created symlink /etc/systemd/system/default.target, pointing to /usr/lib/systemd/system/rescue.target.</pre>
To change the system to graphical mode after booting	<pre>[root@localhost ~]# systemctl isolate graphical.target</pre>

The systemctl command

The `systemctl` command is used in systems that have systemd as a replacement for the traditional `init` process. This one command can be used to manually control the state of services, enable or disable automatic starting of services, as well as change system targets.

To manually control the state of a service, use the `systemctl` command to start, stop, or check the status of that service. For example, to start a service like the web server, execute the following:

```
systemctl start httpd.service
```

To shut down the service:

```
systemctl stop httpd.service
```

To check the state of the service:

```
systemctl status httpd.service
```

To view the status of all services:

```
systemctl -a
systemctl --all
```

To configure a service to start automatically, execute the following:

```
systemctl enable httpd.service
```

To configure a service not to start automatically, execute the following:

```
systemctl disable httpd.service
```

It is possible to change to a different runlevel with the `systemctl` command:

```
systemctl isolate DESIRED.TARGET
```

The `systemctl` command can also manage the low or no power states of the system with command lines such as:

```
systemctl hibernate  
systemctl suspend  
systemctl poweroff  
systemctl reboot
```

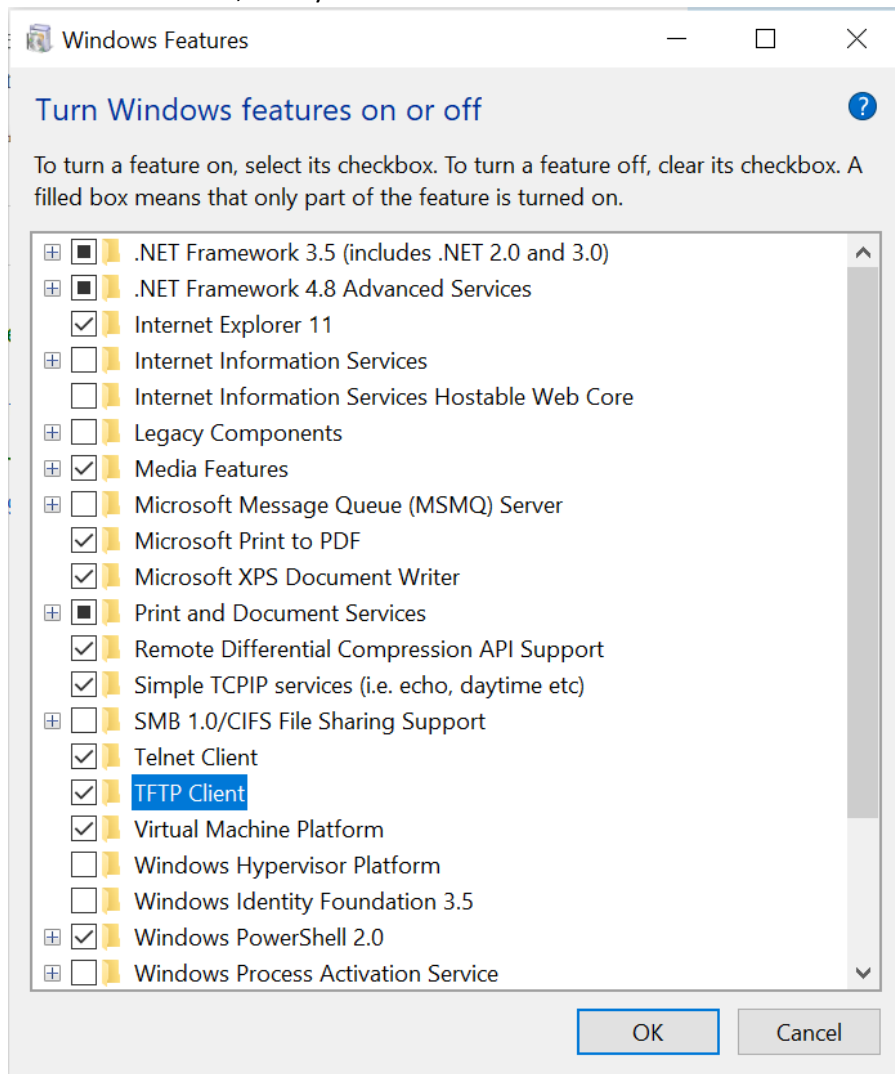
Week 1 lab01C Windows Server Notes

Telnet:

In the lab video, Jingsong shows you how to use your host machine to telnet to Sever2019vm. When you try to telnet from your host to VM, you may get a message like this “**Telnet is not recognized as an internal or external command**”. This error indicates that the Telnet utility is not installed on your

system. However, Telnet is a default feature in **Windows 10**, just the protocol isn't enabled yet. Follow the following steps to enable Telnet.

1. Open control panel
2. Select programs
3. Choose "Turn Windows features on or off"
4. Tick "Telnet Client", then you should be able to use telnet in CLI.



Simple TCP/IP service

Simple TCP/ IP services are used for testing the functionality of TCP/IP services. These services are not required to perform essential network operations but the vulnerabilities in them could be exploited by hackers to perform Denial of Service (DOS) or bandwidth-hogging attacks. We only enable it for testing purpose, remember disable it after the lab.

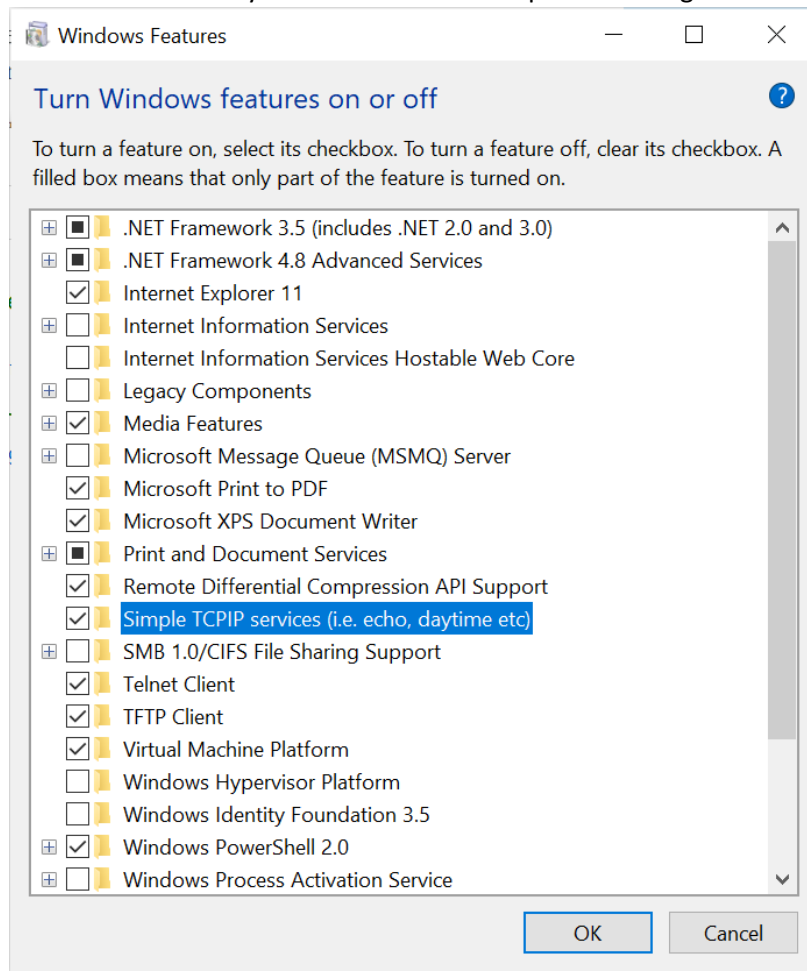
Simple TCP/IP Services (simptcp) implements support for the following protocols and ports:

- Echo, port 7, RFC 862

- Discard, port 9, RFC 863
- Character Generator, port 19, RFC 864
- Daytime, port 13, RFC 867
- Quote of the Day, port 17, RFC 865

To add "Simple TCP/IP" Services:

1. Open control panel Step
2. Navigate to programs and features
3. select "Turn Windows features on or off."
4. tick "Simple TCP/IP services"
5. You need to reboot your PC to finish the requested change.



Netsh Command Syntax and contexts

Netsh is a command-line scripting utility that allows you to display or modify the network configuration of a computer that is currently running. Netsh commands can be run by typing commands at the netsh prompt and they can be used in batch files or scripts. Remote computers and the local computer can be configured by using netsh commands. Use "netsh help" to obtain a list of options.

Lab 2a System Documentation Notes

Man pages are used to describe the features of commands. They provide a basic description of the purpose of the command, as well as details regarding available options.

To view a man page for a command, use the **man** command:

```
man command
```

By default, there are nine sections of man pages:

1. General Commands
2. System Calls
3. Library Calls
4. Special Files
5. File Formats and Conventions
6. Games
7. Miscellaneous
8. System Administration Commands
9. Kernel Routines

To specify a different section, provide the number of the section as the first argument of the man command. The following command displays the passwd man page located in section 5, which is associated with the passwd file:

```
sysadmin@localhost:~$ man 5 passwd
```

/etc/man_db.conf is used by the man-db package to configure the man and cat the paths. **SECTION** in this file decides the search sequence.

The **whatis** command (or **man -f**) returns what section a man page is stored in.

whatis command in Linux is used to get a one-line manual page descriptions. **whatis** searches the manual page names and displays the manual page descriptions of any *name* matched. **whatis** use **index** databases during the search, the **index** databases are updated by the **mandb** program. Depending on your installation, this may be run by a periodic cron job, or may need to be run manually after new manual pages have been installed. To manually create the database : **mandb -cqs** The **index** databases contains information relevant to the current state of the manual page system and the information stored within them is used by the man-db utilities to enhance their speed and functionality

The `info` command also provides documentation on operating system commands and features. The goal is slightly different from man pages: to provide a documentation resource that gives a logical organizational structure, making reading documentation easier.

All of the documentation is merged into a single "book" representing all of the documentation available. Within info documents, information is broken down into categories that work much like a table of contents in a book. Hyperlinks are provided to pages with information on individual topics for a specific command or feature.

Another advantage of info over man pages is that the writing style of info documents is typically more conducive to learning a topic. Consider man pages to be more of a reference resource and info documents to be more of a learning guide.

To display the info documentation for a command, use the `info` command.

```
info command
```

Week2 Lab 2b System updates Notes

Managing Packages with yum

Centos use the `yum` command to locate and download packages on the internet and resolve dependencies automatically.

Typically, the `yum` command is configured by editing the `/etc/yum.conf` file and the files found in the `/etc/yum.repos.d` directory. These configuration files are used to specify servers (the repositories) on the internet where the `yum` command can obtain the RPM files automatically. If the system does not have networking enabled or doesn't have access to the internet, then it will not be able to use the `yum` commands to manage packages. In those situations, the `rpm` commands can be used for package management.

YUM QUERIES

SUBCOMMAND	DESCRIPTIONS AND TASKS
------------	------------------------

help	Display yum commands and options yum help Show yum subcommands and options
-------------	---

Individual packages

list	List package names from repositories yum list available List all available packages yum list installed List all installed packages yum list all List installed and available packages yum list kernel List installed and available kernel packages
-------------	--

info	Display information about a package yum info vsftpd List info about vsftpd package
-------------	---

deplist	Display dependencies for a package yum deplist nfs-utils List dependencies and packages providing them
----------------	---

provides	Find packages that provide the queried file yum provides "**bin/top" Show package that contains top command yum provides "**/README.top" Show package containing README.top file
-----------------	--

search	Search package names and descriptions for a term yum search samba Find packages with samba in name or description
---------------	--

updateinfo	Get information about available package updates yum updateinfo security Get info on available security updates
-------------------	---

INSTALL, REMOVE AND UPGRADE PACKAGES WITH YUM

SUBCOMMAND	DESCRIPTIONS AND TASKS
install	Install a package from a repository to your system <code>yum install vsftpd</code> Install the vsftpd package
update	Update one or all packages on your system <code>yum update</code> Update all packages with available updates <code>yum update httpd</code> Update the httpd package (if available) <code>yum update --security</code> Apply security-related package updates
update-to	Update one or all packages to a particular version
upgrade	Update packages taking obsoletes into account
localinstall	Install a package from a local file, http, or ftp <code>yum localinstall abc-1-1.i686.rpm</code> Install abc package from local directory <code>yum localinstall http://myrepo/abc-1-1.i686.rpm</code> Install abc from FTP site
downgrade	Downgrade a package to an earlier version <code>yum downgrade abc</code> Downgrade the abc package to an earlier version
reinstall	Reinstall the current version of a package <code>yum reinstall util-linux</code> Reinstall util-linux (to replace any deleted files)
swap	Remove one package and install another <code>yum swap ftp lftp</code> Remove ftp package and install lftp package
erase	Erase a package (and possibly dependencies) from your system <code>yum remove vsftpd</code> Remove the vsftpd package and dependencies
remove	Same as erase
autoremove	Same as erase, plus removes additional unneeded packages * <code>yum autoremove httpd</code> Remove httpd and other unneeded packages
groupinstall	Install all packages in the selected group <code>yum groupinstall "Web server"</code> Install Web Server packages

Week2 Lab 2c Manage processes

1. List processes

The `ps` command can be used to list processes.

```
ps [OPTION]...
```

The `ps` command supports three styles of options:

- Traditional UNIX style short options that use a single hyphen in front of a character
- GNU style long options that use two hyphens in front of a word
- BSD style options that use no hyphens and single character options

The `ps` command will display the processes that are running in the current terminal by default:

```
sysadmin@localhost:~$ ps
  PID TTY          TIME CMD
   80 ?            00:00:00 bash
   94 ?            00:00:00 ps
```

With traditional (non-BSD) options, the `-e` option will display every process. Typically, the `-f` option is also used as it provides full details of the command, including options and arguments:

```
sysadmin@localhost:~$ ps -ef
UID          PID  PPID  C STIME TTY          TIME CMD
root           1      0  0 17:16 ?            00:00:00 /sbin??? /init
syslog        33      1  0 17:16 ?            00:00:00 /usr/sbin/rsyslogd
root         38      1  0 17:16 ?            00:00:00 /usr/sbin/cron
root         40      1  0 17:16 ?            00:00:00 /usr/sbin/sshd
bind         57      1  0 17:16 ?            00:00:00 /usr/sbin/named -u bind
root         70      1  0 17:16 ?            00:00:00 /bin/login -f
sysadmin     80      70  0 17:16 ?            00:00:00 -bash
sysadmin     96      80  0 17:26 ?            00:00:00 ps -ef
```

While the `ps` command can display active processes, the `top` command provides the ability to monitor processes in real-time, as well as manage the processes. By default, the output of the `top` command will update every three seconds. For example, the following output demonstrates how the `top` command can be used to monitor processes currently running in the terminal, such as the three `cat` commands executed in the previous section:

```
sysadmin@localhost:~$ top
top - 16:47:34 up 51 days,  2:12,  1 user,  load average: 1.37, 1.56, 1.49
```

```
Tasks: 13 total, 4 running, 9 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.3 us, 37.2 sy, 0.2 ni, 62.1 id, 0.0 wa, 0.1 hi, 0.0 si, 0.0 st
KiB Mem: 16438128 total, 13108516 used, 3329612 free, 4276 buffers
KiB Swap: 0 total, 0 used, 0 free. 9808716 cached Mem
```

PID	USER	PR	NI	VRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
164	root	20	0	4364	696	616	R	87.4	0.1	1:23.32	cat
165	root	30	10	4364	696	620	T	9.3	0.1	0:49.13	cat
166	root	39	19	4364	772	696	T	1.7	0.1	0:41.75	cat
1	root	20	0	17960	2972	2724	S	0.0	0.0	0:00.02	init

2. Process Priority

When a process runs, it needs to have access to the CPU to perform actions. Not all processes have the same access to the CPU. For example, system processes typically have a higher *priority* when accessing the CPU.

The Linux kernel dynamically adjusts the priority of processes to try to make the operating system seem responsive to the user and efficient at performing tasks. A user can influence the priority that will be assigned to a process by setting a value of something called *nice*ness.

-20

0

19

Highest priority

Default niceness

Lowest priority

The higher you set the niceness value, the lower the priority that will be assigned to a process. The default value of niceness for processes is zero; most user processes run at this nice value. Only a user with administrative (root) access can set negative niceness values or alter the niceness of an existing process to be a lower niceness value.

To set the initial niceness of a command, use the `nice` command as a prefix to the command to execute.

The `-n` option indicates the desired niceness value. For example, to execute a command at the lowest priority possible, execute the following command:

```
sysadmin@localhost:~$ nice -n 19 cat /dev/zero > /dev/null
^Z
[1]+  Stopped                  nice -n 19 cat /dev/zero > /dev/
```

3. Background Processes

When a command may take some time to execute, it may be better to have that command execute in the *background*. When executed in the background, a child process releases control back to the parent process (the shell, in this case) immediately, allowing the user to execute other commands.

To have a command execute as a background process, add the ampersand & character after the command.

```
COMMAND &
```

While there are still background processes being run in the terminal, they can be displayed by using the `jobs` command. It is important to point out that the `jobs` command will only show background processes in the current terminal. If background processes are running in another terminal, they will not be shown by running the `jobs` command in the current terminal. The following is an example of using the `jobs` command:

```
sysadmin@localhost:~$ sleep 1000 &
[1] 106
sysadmin@localhost:~$ sleep 2000 &
[2] 107
sysadmin@localhost:~$ jobs
[1]-  Running                  sleep 1000 &
[2]+  Running                  sleep 2000 &
```

To terminate the processes, send them to the foreground by using the `fg` command with the job number of the process to terminate, and while the process is running in the foreground, use **Ctrl+C**, a signal which stops the process:

```
sysadmin@localhost:~$ fg 1
sleep 1000
^C
sysadmin@localhost:~$ fg 2
sleep 2000
^C
sysadmin@localhost:~$ jobs
sysadmin@localhost:~$
```

Ctrl+Z, is used to pause a process instead of stop the process:

```
^Z
[1]+  Stopped                  sleep 1000
```

The `bg` command resumes jobs without bringing them to the foreground. If no argument is provided, it will use the most recently suspended job.

```
sysadmin@localhost:~$ bg
[1]+ sleep 1000 &
```

Week2 Lab2d Disk Partitioning

The hard drive is a storage container that is capable of storing all of the system's files. However, there are benefits to breaking this large container into smaller containers, we call this process as making partitions.

There are three steps in the process of making and using partitions:

1. Divide the hard drive into partitions.
2. Create and format filesystems inside the partitions.
3. Mount the filesystem into the directory tree.

1. Divide the hard drive into partitions

In order to distinguish one partition from another, each partition is given a unique name. Recall that everything in Linux is treated as a file, so the names of devices, such as drives and partitions, are stored in the `/dev` directory.

There are two different types of drive interfaces:

Device Type	Name	Example
Drives that start with <code>sd</code> are either SATA (Serial ATA), SCSI (Small Computer System Interface) or USB drives.	<code>/dev/sd*</code>	<code>/dev/sda</code> <code>/dev/sdb</code>
Drives that start with <code>hd</code> are PATA (Parallel ATA), also known as IDE (Integrated Drive Electronics) drives.	<code>/dev/hd*</code>	<code>/dev/hda</code> <code>/dev/hdb</code>

Partitions are given names based on the drive they reside on. A number is added to the end of the drive name to distinguish one partition from another. For example, the partitions located on drive `sda` would be named `sda1`, `sda2`, etc. Partitions located on drive `sdb` would be `sdb1`, `sdb2`, etc. The following output shows the device files for a SATA hard drive with nine partitions:

```
sysadmin@localhost:~$ cd /dev
sysadmin@localhost:/dev$ ls sd*
sda  sda1  sda2  sda3  sda4  sda6  sda7  sda9
```

Similarly, partitions on drive `hda` would be `hda1`, `hda2`, etc., while partitions on `hdb` would be `hdb1`, `hdb2`, etc.

`fdisk` command can be used to create, modify, and list the partitions on a hard drive.

The `fdisk` program can be used in two ways: *interactive* and *non-interactive*. The interactive mode is used to modify the partitions, and the non-interactive mode is used to list partitions. In either mode, the `fdisk` program requires root privileges to run.

```
root@localhost:~# fdisk -l /dev/sda
Disk /dev/sda: 11.3 GB, 11261706240 bytes
255 heads, 63 sectors/track, 1369 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000ee7d2

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           1           64       512000   83   Linux
Partition 1 does not end on cylinder boundary.
/dev/sda2                64        1306       9972736   Linux LVM
/dev/sda3          1306        1319        102400   82   Linux swap / Solaris
Partition 3 does not end on cylinder boundary.
```

2. Create and format filesystems inside the partitions.

In order to place files and directories on a partition, a filesystem needs to be created. This is accomplished by formatting the partition. A filesystem provides the operating system with a way to organize the files and directories. Moreover, a filesystem stores information about files, such as the permissions of the files, the owner of the files, and the file type.

Some of the more common filesystem types include:

Type	Name	Advantages	Disadvantages
ext2	Second Extended Filesystem	Works well with small and solid-state disk filesystems	No journaling capability, making it susceptible to data loss in the event of power loss.
ext3	Third Extended Filesystem	Can be upgraded from existing ext2 filesystem without data loss. This filesystem performs journaling, which allows for data recovery after a crash.	Writes more to disk than ext2 because of journaling, making it slower. Does not support very large filesystems.

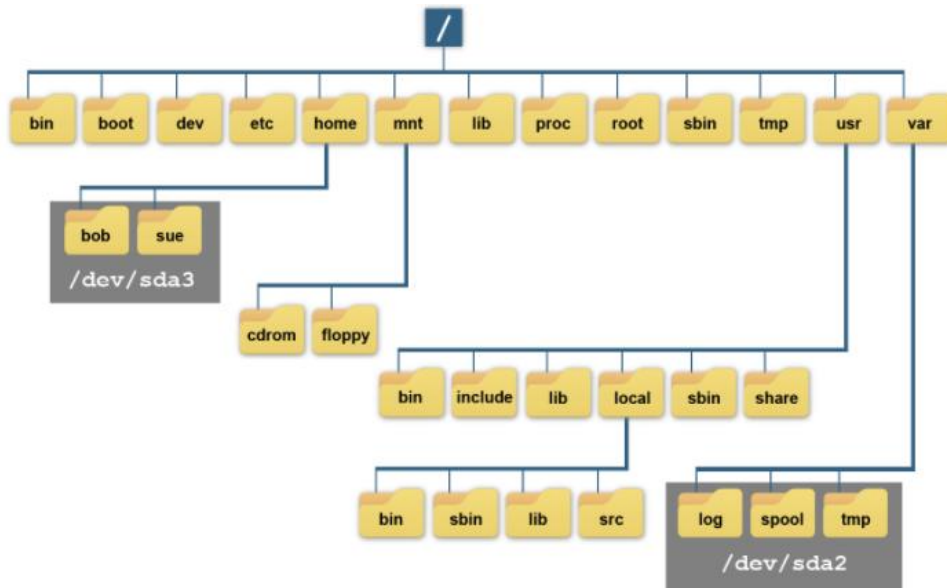
Type	Name	Advantages	Disadvantages
ext4	Fourth Extended Filesystem	Support for very large disk volumes and file sizes. Can operate with or without a journal. Backwards compatible with ext3 and ext2.	Not a huge improvement over ext3. No dynamic inode creation.
xfs	Extents Filesystem	Works very efficiently with large files. Compatible with the IRIX operating system from SGI. Announced to be the default filesystem for RHEL 7.	The filesystem cannot be shrunk.
vfat	File Allocation Table	Supported by almost all operating systems. Commonly used for removable media.	Unable to support very large disks or files. Microsoft's intellectual property claims.
iso	ISO 9660	The International Organization for Standardization standard for optical disc media that is supported by almost all operating systems.	Multiple levels and extensions complicate compatibility. Not designed for rewritable media.
udf	Universal Disc Format	Designed to replace ISO 9660 and adopted as the standard format for DVDs by the DVD Consortium.	Write support is limited to support revision 2.01 of the standard.

The `mkfs` command can be used to create a filesystem. To make a vfat type filesystem, which is compatible with multiple operating systems, including Microsoft Windows, execute a command like the following:

```
root@localhost:~# mkfs -t vfat /dev/sdb1
```

3. Mount the filesystem into the directory tree.

Linux users do not access the files that are stored on partitions by directly using device file names `/dev/sda1`, `/dev/sda2`, etc, it uses a directory tree.



The process of placing a filesystem under a mount point is called *mounting*. This is accomplished either automatically at boot or manually with the `mount` command.

Director y	Purpose	Suggested Size
/	The root filesystem holds the files essential to the operation of the system. It must contain the following directories or symbolic links: bin, boot, dev, etc, lib, media, mnt, opt, sbin, srv, tmp, usr, and var.	500MiB-50GiB+ Depends on what is mounted separately
/boot	The /boot directory contains the Linux kernel and the boot loader files.	500MiB-2GB
/home	The /home directory is where user home directories are normally created.	500MiB+ per user
/tmp	The /tmp directory is used to create temporary files for the system and users. If this directory is too small, it may prevent applications from functioning correctly.	Minimum of 5 GB + 500MiB+ per user actively logged in
/opt	The /opt directory is where third-party software is often installed. Some examples include Google Chrome and Google Earth.	100MiB+ Depends on how many packages are installed

Director y	Purpose	Suggested Size
swap	Swap is virtual memory that is not mounted on a directory. This virtual memory is used when the actual memory of the system is low. On systems with large amounts of memory, swap is less important.	Up to 2 times the physical memory of the system
/usr	The /usr directory contains the bulk of the operating system's files, including most of the commands and system software.	2GiB-10GiB+
/usr/local	This directory is used for locally installed software that should not be upgraded or updated with the operating system.	100MiB+ Size depends on local needs
/var	There are many directories that may have heavy activity under /var for services like mail, ftp, http, and printing.	100MiB+ Depending on the volume of activity
/boot/efi	The /boot/efi directory contains the Linux kernel and the boot loader files. It is typically set up automatically by the installer.	100MB-250MB

Mounting of partitions and checking on existing mounts is accomplished with the `mount` command. When called with no arguments, the `mount` command shows the currently mounted devices. This can be performed by regular users, not just the root user.

In order to properly mount a filesystem, at least two parameters must be provided to the `mount` command: the filesystem identifier and the directory path name to mount it on. If no other options are needed, the `mount` command will mount the filesystem on the specified directory mount point.

To unmount a filesystem, use the `umount` command with either the filesystem or the directory given as an argument.

Mounting Filesystems Automatically On Boot

The `/etc/fstab` file is used to configure what filesystems will be mounted automatically at boot. It can also be used to activate swap devices automatically. Root privileges are required to make changes to the `/etc/fstab` file. Any changes made to this file should be performed with care, as mistakes may prevent the system from booting normally. In fact, it would be an excellent idea to create a backup of the `/etc/fstab` file before making changes.

A system with three filesystem partitions and one swap partition might have an `/etc/fstab` file that looks like the following:

```
#
# /etc/fstab
# Created by anaconda on Fri Jan 17 10:31:51 2014
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=3db6ba40-67d2-403d-9c0a-9a901697cd8d /          ext4      defaults      1 1
UUID=09d641d5-bc5a-4065-8d80-8ae797dfa7f3 /boot      ext4      defaults      1 2
UUID=5ee634a5-c360-4211-a41b-9aa40d78a804 /home      ext4      defaults      1 2
UUID=34819281-65e3-4c78-ba2d-16952684c9cb swap       swap      defaults      0 0
tmpfs      /dev/shm   tmpfs      defaults      0 0
devpts     /dev/pts   devpts     gid=5,mode=620 0 0
sysfs      /sys       sysfs      defaults      0 0
proc       /proc      proc       default
```

Device Identifier/Device Name /Device Label field:

This could be the name of a device like `/dev/sdb1`, or universally unique identifier (UUID) . To determine what the UUID value is for a device, use the `blkid` command. The UUID is automatically created when the filesystem is created. Another option is to use filesystem *labels*. The volume label can be set using the `e2label` command.

Mount point field:

where the partition is to be mounted.

Filesystem field:

It is the type of the file system, for example, `ext2`,`ext3`,`ext4`,`vfat`

Mount Option field:

The fourth field in the `/etc/fstab` file is a comma-separated list of the mount options. This field is used to pass parameters to the filesystem driver, such as to make the device read-only or to adjust timeouts. For example, `ro`, `rw`, `suid`, `exec`, `auto`,`defaults`

Dump field:

The purpose of this field is to tell an administrator which filesystems should be backed up when using the `dump` command. This is a bit out of date, as the `dump` command is rarely used by system administrators on modern Linux distributions.

Filesystem Check field:

The sixth field is for determining the order in which the filesystems will be checked by the `fsck` (File System Check) utility during system boot. This utility is designed to find and fix filesystem problems. The root filesystem should always have a 1 in this field to indicate that it will be checked by the `fsck` program first. All other local filesystems (ext2/ext3/ext4) should have a value of 2 specified for this field, so they will be checked after the root filesystem.

Creating a Swap Partition

The steps to creating a swap partition are:

1. Create a partition with an Id of 82 using `fdisk` as previously described:

```
Command (m for help): n
Command Action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 3
First sector (20971520-21995519, default 20971520):
Using default value 20971520
Last sector, +sectors, or +size{K,M,G} (20971520-21995519, default 21995519): +100

Command (m for help): t
Partition number (1-6): 3
Hex code (type L to list codes): 82
Changed system type of partition 3 to 82 (Linux swap / Solaris)
```

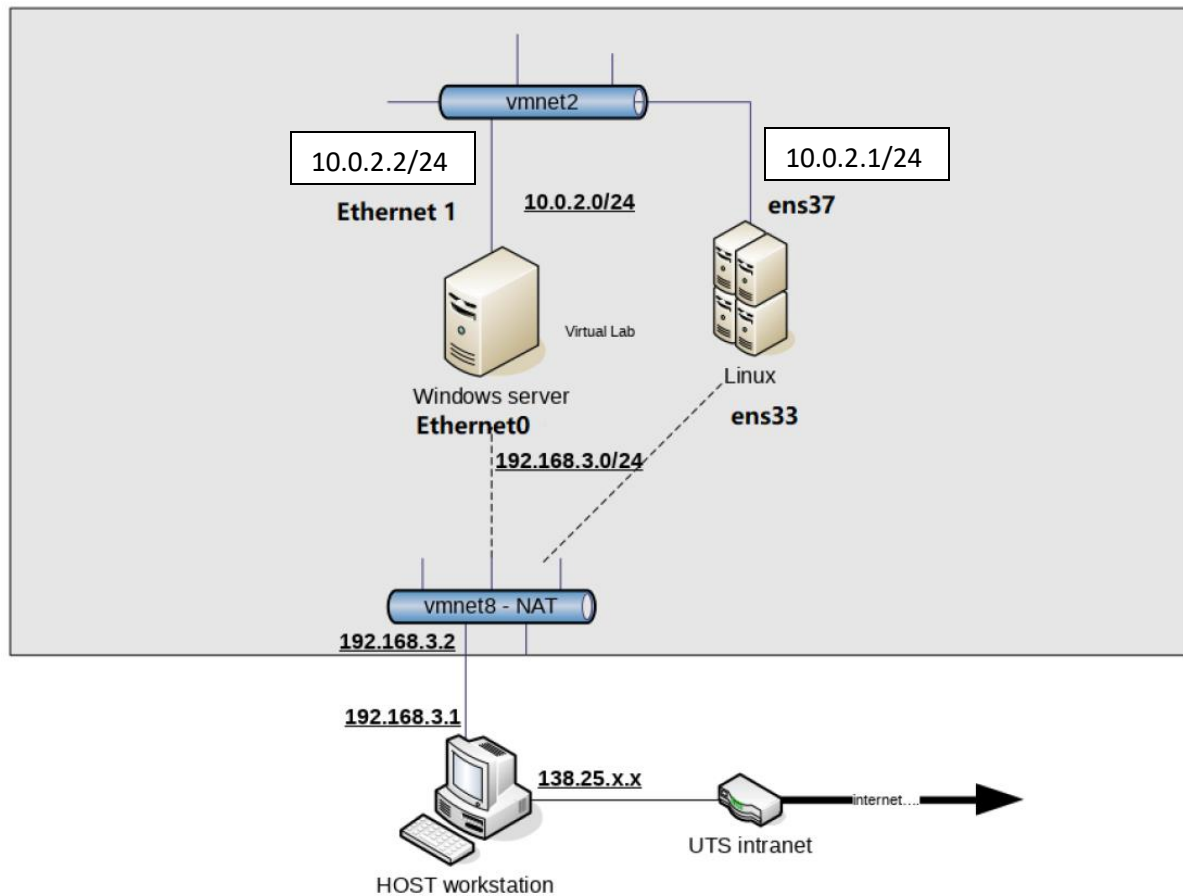
2. Convert the partition to swap space with the `mkswap` command.

```
root@localhost:~# mkswap /dev/sda3
Setting up swspace version1, size = 102396 KiB
no label, UUID=59aaf06e-7109-471f-88a5-e81dd7c82d76
```

3. Enable the partition as current swap space with the `swapon` command:

```
root@localhost:~# swapon /dev/sda3
```

Lab 3a Static Networking



IP Address Configuration

The `ifconfig` command stands for *interface configuration* and is used to display network configuration information. The `ifconfig` command can also be used to modify network settings temporarily.

The `ifconfig` command can be used without options or arguments to display all interfaces on the network, or with options and an interface name as an argument:

```
ifconfig [INTERFACE] [OPTIONS]
```

To assign an IP address to an interface, execute the following command:

```
sysadmin@localhost:~#ifconfig ens37 192.168.1.1 netmask 255.255.255.0
```

The `ifconfig` command is becoming obsolete in some Linux distributions (deprecated) and is being replaced with a form of the `ip` command, it can almost be a one-stop shop for configuration and control of a system's networking. The format for the `ip` command is as follows:

```
ip [OPTIONS] OBJECT COMMAND
```

Refer to the cheat sheet for commands example

The route Command

Recall that a router (or gateway) is a machine that allows hosts from one network to communicate with another network. To view a table that describes where network packages are sent, use the `route` command:

```
root@localhost:~# route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0    *               255.255.255.0   U        0      0      0 eth0
default        192.168.1.1    0.0.0.0         UG       0      0      0 eth0
```

The first highlighted line in the preceding example indicates that any network package sent to a machine in the `192.168.1` network is not sent to a gateway machine (the `*` indicates *no gateway*). The second highlighted line indicates that all other network packets are sent to the host with the IP address of `192.168.1.1` (the router).

Some users prefer to display this information with numeric data only, by using the `-n` option to the `route` command. For example, look at the following and focus on where the output used to display default:

```
root@localhost:~# route -n
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Ifa
192.168.1.0    0.0.0.0         255.255.255.0   U        0      0      0 eth0
0.0.0.0        192.168.1.1    0.0.0.0         UG       0      0      0 eth0
```

The `0.0.0.0` refers to *all other machines*, and is the same as *default*.

The fields in the output are as follows:

Option	Meaning
--------	---------

Destination	Destination host or network
-------------	-----------------------------

Option	Meaning
Gateway	IP address of the gateway 0.0.0.0 means “no route, broadcast on this network”
Genmask	Subnet mask for the destination network Set to 0.0.0.0 for the default route
Flags	Values include C (cache entry), G (use gateway) and U (up)
Metric	Number of hops to the target, used as a measure of distance
Ref	Number of references to this route; this is not used in Linux
Use	Count of lookups done for this route
Iface	Interface to use for sending packets for this route

To add this route to the `eth0` interface, the administrator could execute the following `sudo` command (or as root):

```
sysadmin@localhost:~$ sudo route add -net 192.56.78.0 netmask 255.255.255.0 gw 192.168.1.1 dev eth0
```

To add a default gateway, execute the following command:

```
sysadmin@localhost:~$ sudo route add default gw 192.168.1.1
```

To delete a route from the routing table, an administrator can execute a command like the one that added it, except using `del` instead of `add`. For example, to delete the route and default gateway that was added earlier, an administrator could execute the following `sudo` commands (or as root):

```
sysadmin@localhost:~$ sudo route del -net 192.56.78.0 netmask 255.255.255.0 gw 192.168.1.1 dev eth0
```

```
sysadmin@localhost:~$ sudo route del default gw 192.168.1.1
```

The `route` command is becoming obsolete in some Linux distributions (deprecated) and is being replaced with a form of the `ip` command, specifically `ip route show`. Note that the same information highlighted above can also be found using this command:

```
root@localhost:~# ip route show
```



```
default via 192.168.1.254 dev eth0 proto static
2.168.1.0/24 dev eth0 proto kernel scope link src 192.1
```

19

The netstat Command

The `netstat` command is a powerful tool that provides a large amount of network information. It can be used to display information about network connections as well as display the routing table similar to the `route` command.

To use the `netstat` command to display routing information, use the `-r` option:

```
root@localhost:~# netstat -r

Kernel IP routing table

Destination  Gateway          Genmask          Flags   MSS Window  irtt Iface
192.168.1.0  *                255.255.255.0    U        0  0        0 eth0
default      192.168.1.1      0.0.0.0          UG        0  0        0 eth0
```

The `netstat` command is also commonly used to display open *ports*. A port is a unique number that is associated with a service provided by a host. If the port is open, then the service is available for other hosts.

For example, you can log into a host from another host using a service called *SSH*. The SSH service is assigned port #22. So, if port #22 is open, then the service is available to other hosts.

It is important to note that the host also needs to have the services running itself; this means that the service (in this case the ssh daemon) that allows remote users to log in needs to be started (which it typically is, for most Linux distributions).

To see a list of all currently open ports, use the following command:

```
root@localhost:~# netstat -tln

Active Internet connections (only servers)

Proto Recv-Q Send-Q Local Address           Foreign Address         State       Pr
tcp    0      0 192.168.1.2:53          0.0.0.0:*               LISTEN      1
tcp    0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      1
tcp    0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1
tcp    0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      1
tcp6   0      0 :::53                   :::*                    LISTEN      1
tcp6   0      0 :::22                   :::*                    LISTEN      1
```

As you can see from the output above, port #22 is *listening*, which means it is open.

In the previous example, `-t` stands for *TCP* (recall this protocol from earlier in this chapter), `-l` stands for *listening* (which ports are listening) and `-n` stands for *show numbers, not names*.

Sometimes showing the names can be more useful. This can be achieved by dropping the `-n` option:

```
root@localhost:~# netstat -tl
```

Active Internet connections (only servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	to
p	0	0	cserver.example.:domain	*:*	LISTEN	
tcp	0	0	localhost:domain	*:*	LISTEN	
tcp	0	0	*:ssh	*:*	LISTEN	
tcp	0	0	localhost:953	*:*	LISTEN	
tcp6	0	0	[::]:domain	[::]:*	LISTEN	
tcp6	0	0	[::]:ssh	[::]:*	LISTEN	
tcp6	0	0	localhost:953	[::]:*	LISTEN	

Primary IPv4 Configuration File

On a CentOS system, the primary configuration file for an IPv4 network interface is the `/etc/sysconfig/network-scripts/ifcfg-ensxx` file. The following shows what this file looks like when configured for a static IP address. (On a legacy Red Hat-derived system, the `/etc/sysconfig/network` file contains host and routing details for all configured network interfaces.)

```
root@localhost:~# cat /etc/sysconfig/network-scripts/ifcfg-eth0
```

```
DEVICE="ens33"
BOOTPROTO=none
NM_CONTROLLED="yes"
ONBOOT=yes
TYPE="Ethernet"
UUID="98cf38bf-d91c-49b3-bb1b-f48ae7f2d3b5"
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="ens33"
IPADDR=192.168.1.1
PREFIX=24
#NETMASK=255.255.255.0
GATEWAY=192.168.1.1
DNS1=192.168.1.2
```

```
HWADDR=00:50:56:90:18:18
```

```
LAST_CONNECT=1376319928
```

If the device were configured to be a DHCP client, the `BOOTPROTO` value would be set to `dhcp`, and the `IPADDR`, `GATEWAY` and `DNS1` values would not be set.

The widely accepted method of making changes to a network interface is to take the interface down using a command such as `ifdown ens33`, make the desired changes to the configuration file, and then bring the interface back up and into service with a command such as `ifup ens33`.

Setting the Hostname

The *hostname* is used to identify the system by applications such as web servers. On Debian-derived and modern Red Hat-derived systems, the `/etc/hostname` file contains this information, while legacy Red Hat-derived systems store this information in the `/etc/sysconfig/network` file. This file is read at boot time to set the hostname.

The `hostname` command is used to set and view the system's host and domain name. It is the system administrator's responsibility to assign an appropriate hostname. It cannot be longer than 64 characters and can contain alphanumeric `[a-z]` `[0-9]`, period `.` characters, and hyphen `-` characters only.

To view the currently assigned hostname of the system, execute the `hostname` or *short name cut at the first dot* `hostname -s`, or *full domain name* `hostname -f` command:

```
sysadmin@localhost:~$ hostname -s
localhost
```

The name displayed above is returned by the `gethostname()` application programming interface (API).

To view the fully-qualified domain name, execute the following command:

```
sysadmin@localhost:~$ hostname -f
test.example.com
```

To set the hostname of the system, the root user can execute the following command:

```
root@localhost:~# hostname example.com
root@localhost:~# hostname
example.com
```

Note that setting the hostname using the `hostname` command results in a change that is only persistent until the next system boot.

The `/etc/hosts` file is used for mapping hostnames with IP addresses. It is a flat-file with one record on each line. The format of the file is:

<i>IP Address</i>	<i>Host Name</i>	<i>Alias</i>
-------------------	------------------	--------------

A sample `/etc/hosts` file will look like the following:

127.0.0.1	localhost	
192.168.4.8	apps.sample.com	apps
192.168.4.12	vm1.sample.com	vm1

The *Alias* field is used for mapping short names or labels to a host.

The functionality of the `/etc/hosts` file has been relegated by DNS but is still used in the following situations:

- **Bootstrapping:** This file is referred to during system startup since the DNS service is not started at this point.
- **Isolated Nodes:** If a node is not connected to the internet, it is unlikely to use DNS. The `/etc/hosts` file is useful for such nodes.
- **NIS:** The records in the hosts file are used as input for the NIS (Network Information Services) database.

Systemd systems use an alternative to the `hostname` command, the `hostnamectl` command. Similar to the `hostname` command, the `hostnamectl` command can also be used to query and set system hostnames, but the `hostnamectl` command provides additional categories for hostnames; *static*, *pretty*, and *transient* which are described below:

- **Static:** A *static hostname* is limited to [a-z], [0-9], hyphen -, and period . characters (no spaces i.e., `localhost` or `ndg-server`). This hostname is stored in the `/etc/hostname` file. Static hostnames can be set by a user.
- **Pretty:** Hostname can be in a human-readable format using any valid UTF-8 characters and can include special characters (i.e., `Sarah's Laptop` or `Joe's Home PC`).
- **Transient:** The transient hostname is a dynamic hostname usually set by the kernel to `localhost` by default. A dynamic hostname can be modified if needed. The transient hostname can be modified by DHCP or mDNS at runtime.

Hostnames can be up to 64 characters, but it is recommended that static and transient hostnames are limited to only 7 bit ASCII lowercase characters with no spaces or dots and conforming to strings acceptable for DNS domain names.

To demonstrate, in order to view the current hostname, simply use the `hostnamectl` command by itself or with the `status` subcommand:

```
sysadmin@localhost:~$ hostnamectl status
Static hostname: localhost
Icon name: computer-vm
Chassis: vm
Machine ID: 5b91eb5e50594030b48b28f103cf5cd6
Boot ID: c04fc73d939a4fc18b279cd46d08f8d7
```

```
Virtualization: kvm
Operating System: Ubuntu 18.04.2 LTS
Kernel: Linux 4.15.0-45-generic
Architecture: x86-64
```

To change the local machine hostname, use the `hostnamectl` command with the `set-hostname` subcommand, this must be done with elevated permissions:

```
sysadmin@localhost:~$ sudo hostnamectl set-hostname student
[sudo] password for sysadmin:
sysadmin@localhost:~$ hostnamectl status

Static hostname: student
Icon name: computer-vm
Chassis: vm
Machine ID: 5b91eb5e50594030b48b28f103cf5cd6
Boot ID: c04fc73d939a4fc18b279cd46d08f8d7
Virtualization: kvm
Operating System: Ubuntu 18.04.2 LTS
Kernel: Linux 4.15.0-45-generic
Architecture: x86-64
```

Configuring DNS

Name resolution on a Linux host is accomplished by 3 critical files: the `/etc/hosts`, `/etc/resolv.conf` and `/etc/nsswitch.conf` files. Together, they describe the location of name service information, the order in which to check resources, and where to go for that information.

Files	Explanation
<code>/etc/hosts</code>	This file contains a table of hostnames to IP addresses. It can be used to supplement a DNS server. <div><pre>sysadmin@localhost:~\$ cat /etc/hosts 127.0.0.1 localhost</pre></div>
<code>/etc/resolv.conf</code>	This file contains the IP addresses of the name servers the system should consult in any attempt to resolve names to IP addresses.

Files	Explanation
-------	-------------

These servers are often DNS servers. It also can contain additional keywords and values that can affect the resolution process.

```
sysadmin@localhost:~$ cat /etc/resolv.conf
nameserver 127.0.0.11
```

/etc/nsswitch.conf	This file can be used to modify where hostname lookups occur. It contains a particular entry that describes in what order name resolution sources are consulted.
--------------------	--

```
sysadmin@localhost:~$ cat /etc/nsswitch.conf
# /etc/nsswitch.conf
#
Output Omitted...
hosts:          files dns
Output Omitted...
```

The `/etc/hosts` file is searched first, the DNS server second:

```
hosts: files dns
```

The `/etc/resolv.conf` file is the configuration file for DNS resolvers. The information in this file is normally set up by network initialization scripts.

A sample `/etc/resolv.conf` file looks like the following:

```
# /etc/resolv.conf
domain      sample.com
search      sample.com
# central nameserver
nameserver  191.74.10.12

sortlist 191.74.10.0 191.74.40.0
```

The format of the `/etc/resolv.conf` is:

```
directive    value1, value2...
```

The configuration directives used in this file are:

Option	Meaning
<code>nameserver</code>	IP address of the name server that the resolver will use Maximum of 3 servers can be listed
<code>domain</code>	Domain name to be used locally
<code>search</code>	Search list to be used for hostname lookup
<code>sortlist</code>	Allow addresses to be sorted The list is specified by IP addresses and optionally the netmask

The `nameserver` setting is often set to the IP address of the DNS server. The following example uses the `host` command, which works with DNS to associate a hostname with an IP address. Note that the example server is associated with the IP address `192.168.1.2` by the DNS server:

```
sysadmin@localhost:~$ host example.com
example.com has address 192.168.1.2
```

It is also common to have multiple `nameserver` settings, in the event that one DNS server isn't responding.

NetworkManager

NetworkManager provides automatic detection and configuration of network interfaces on a Linux system. The NetworkManager supports some tools for users to interact with it, which are:

`nmcli` – a command-line tool used to configure networking.

`nmtui` – a simple curses-based text user interface, which is also used to configure and manage network interface connections.

The `nmcli` command uses the following syntax:

```
nmcli [OPTIONS] OBJECT [COMMAND] [ARGUMENTS...]
```

The `OPTIONS` for the `nmcli` command can be found by visiting the `nmcli` man page or by using the `nmcli -help` command. Commonly used options include the `terse -t` option that displays concise output and the `pretty -p` option, which makes the output easily readable by printing headers and aligning values.

The `OBJECT` field can be one of the following:

Object	Meaning
<code>general</code>	Display information about or modify the status of NetworkManager.
<code>networking</code>	Display information about or modify the network managed by NetworkManager.
<code>connection</code>	Display information about or modify connections managed by NetworkManager.
<code>device</code>	Display information about or modify devices managed by NetworkManager.
<code>radio</code>	Display the status of, and enable or disable, the radio switches.

To demonstrate, the following command can be used to show the existing network device:

The output in the examples below may not match the output in our virtual environment.

```
[sysadmin@centos ~]$ nmcli device
DEVICE    TYPE        STATE        CONNECTION
ens3      ethernet    connected    eth0
```

When using the `nmcli` command, the object can be abbreviated. For example, the `nmcli device` command used in the example above can be shortened to `nmcli dev` or `nmcli d`:

```
[sysadmin@centos ~]$ nmcli dev
DEVICE    TYPE        STATE        CONNECTION
ens3      ethernet    connected    eth0

[sysadmin@centos ~]$ nmcli d
DEVICE    TYPE        STATE        CONNECTION
ens3      ethernet    connected    eth0
```

The `nmcli` command is useful for displaying additional information about a specific connection. In the example below, the `nmcli` command is used with the `con` object, the `show` subcommand and the `pretty -p` option, to display information about the `eth0` connection:

```
[sysadmin@centos ~]$ nmcli -p con show eth0
```

The `nmcli` command can also be used to create a new connection. To add a connection, the following syntax can be used:


```
nmcli con add {OPTIONS} [IP]/[NETMASK] [GATEWAY]
```

For example, to add a connection named `eth1`, define the connection as Ethernet and specify the IP address, network mask, and gateway, the following command can be used:

```
[sysadmin@centos ~]$ nmcli con add con-name eth1 ifname eth1 type ethernet \ip4 10.0.2.18/24 gw4 10.0.2.2
```

The `nmcli` command above uses the `con` object with the `add` subcommand followed by a series of options, which are summarized below:

Options	Meaning
<code>con-name</code>	Specifies the name of the network connection. In the example above, the <code>con-name eth1</code> option adds a connection named <code>eth1</code> .
<code>ifname</code>	Name of the interface (device) that is used for the connection.
<code>type</code>	Specifies the connection type. Various types of connections exist such as ethernet, wifi, VLAN, bridge, and more.
<code>ip4</code>	Specify an IPv4 IP address and netmask for the connection.
<code>gw4</code>	Specify the gateway used for the connection.

The `con show` command can be used to view the new connection:

```
[sysadmin@centos ~]$ nmcli -p con show eth1
```

Week 3 Lab 3b Establishing and maintaining time and date settings

Linux-based systems have two types of clocks:

- **System Clock:** This is a clock maintained by the kernel and is *interrupt-driven*. The value of this clock is initialized from the hardware clock at boot time. The system time is calculated as the number of seconds since January 1st 1970 00:00:00. (This reference time is known as *epoch time* or sometimes *UNIX time*.) The system clock contains the current time as well as time zone information.
- **Hardware Clock:** This is a battery-powered clock that keeps time even when the system is shut down. When the system boots, the system clock is set using the value of the hardware clock. When the system is shut down, the hardware clock is set to the value of the system clock. This ensures that both the clocks are synchronized. The hardware clock is also known as the *real time clock (RTC)* or the *CMOS/BIOS clock*. The hardware clock stores the following values: year, month, day, hour, minute, and seconds.

Maintaining the Hardware Clock

The `hwclock` (hardware clock) command is used by the root user to update and query the hardware clock. The command accesses the hardware clock by performing Input/Output (I/O) via the `/dev/rtc` device file. Some systems may have more than one `rtc` file, so listing the `/dev/rtc*` files may show multiple files like `/dev/rtc0` and `/dev/rtc1`.

- To view the time of the hardware clock, execute the following command as root:

```
root@localhost:~# hwclock
2019-11-21 02:18:18.577020+0000
```

- To set the value of the hardware clock, execute the following command as root:

```
root@localhost:~# hwclock --set --date "1/1/2025 18:30:50"
root@localhost:~# hwclock
2025-01-01 18:30:50.045616+0000
```

Maintaining the System Clock

The `date` command is used to display and set the system date and time. To view the current date and time, execute the following command:

```
root@localhost:~# date
Tue Dec 17 18:50:20 UTC 2024
```

The system date can be displayed in different formats to suit a user's needs. For example, to display only the month, day, and year, execute the following command:

```
root@localhost:~# date "+%m/%d/%y"
12/17/24
```

The following table lists the common format options that the `date` command supports:

Specifier	Meaning
<code>%d</code>	Day of month (e.g., 30)
<code>%H</code>	Hour (0–23)
<code>%I</code>	Hour (1–12)
<code>%m</code>	Month (1–12)
<code>%M</code>	Minute (0–59)
<code>%S</code>	Seconds (0–60)
<code>%T</code>	Time (%H:%M:%S)
<code>%u</code>	Day of week (1–7, 1=Monday)
<code>%Y</code>	Year
<code>%F</code>	Full date; same as %Y-%m-%d

The man page for the `date` command provides a complete list of all the format options.

Displaying and Setting the Time Zone

The `/etc/localtime` File

On CentOS the `/etc/localtime` file is used to configure the time zone of the system. The time zone data for different regions is maintained in the `/usr/share/zoneinfo` directory. To set up a particular time zone, a symbolic link is created from the `/etc/localtime` file to the corresponding file in the `/usr/share/zoneinfo` directory.

To view the contents of the `/usr/share/zoneinfo` directory, execute the following command:

```

root@localhost:~# ls /usr/share/zoneinfo
Africa          Cuba          GMT+0          Kwajalein     Poland        WET            Am
erica          EET          GMT-0          Libya         Portugal      Zulu           Anta
rctica        EST          GMT0           MET           ROC           iso3166.tab    Arctic
              EST5EDT      Greenwich      MST           ROK           leap-seconds.list Asia
              Egypt       HST           MST7MDT       Singapore     localtime      Atlantic
              Eire        Hongkong      Mexico         SystemV       posix          Australia
Etc           Iceland      NZ            Turkey        posixrules    Brazil         Eu
rope          Indian       NZ-CHAT       UCT           right         CET           Fact
ory           Iran         Navajo        US            zone.tab      CST6CDT       GB
              Israel       PRC           UTC           zone1970.tab  Canada        GB-Eire
              Jamaica     PST8PDT       Universal     Chile         GMT           J
apan          Pacific      W-SU

```

To set the time zone to the America Tijuana time zone (PST), execute the following `ln` command as the root user, then verify the change with `date`:

```

root@localhost:~# date
e Dec 17 21:00:59 UTC 2024
root@localhost:~# ln -sf /usr/share/zoneinfo/America/Tijuana /etc/localtime
root@localhost:~# date
17 13:01:32 PST 2024

```

Understanding chronyd

Chrony is a set of programs that are used to ensure that the clock on a system is accurate. `chrony` lends itself to working well in environments with intermittent network connectivity, such as on a laptop or virtual system that may be created through an automated process.

The daemon portion of `chrony` is the command `chronyd`. The daemon synchronizes the system with time retrieved from NTP servers. Along with synchronizing time on the system it is running, `chronyd` can also operate as an NTP server providing time service to other systems that are allowed network access.

To control `chronyd`, you use the `chronyc` program to interface with `chronyd` via the command line. The `chronyc` command can be used in two different modes; *interactive* and *non-interactive* mode. If no argument is specified on the command line, the `chronyc` command will run in interactive mode and return a `chrony>` prompt where you can then enter `chronyc` commands.

```

root@localhost:~# chronyc

chrony version 3.2

Copyright © 1997-2003, 2007, 2009-2017 Richard P. Curnow and others

chrony comes with ABSOLUTELY NO WARRANTY. This is free software, and you are
welcome to redistribute it under certain conditions. See the GNU General Pub
lic License version 2 for details.

```

```
chronyc>
```

Note that the `chronyc` command should be run as the root user since some of the `chronyc` commands may be restricted for regular users.

Common `chronyc` commands are listed below:

Argument	Description
<code>tracking</code>	Displays performance statistics about the system clock
<code>sources</code>	Displays the NTP sources being used for <code>chronyd</code>
<code>activity</code>	Displays the status of NTP sources
<code>settime</code> <code><TIME></code>	Allows you to manually set the time used for <code>chronyd</code> . The format for <code>settime</code> can be any of the below: hh:mm hh:mm:ss Month Day, YYYY hh:mm:ss

To demonstrate, if you want to display information and performance statistics about the system clock in interactive mode, use the `tracking` command at the `chronyc>` prompt:

```
chronyc> tracking
Reference ID      : 6C3D49F3 (hydrogen.constant.com)
Stratum          : 3
Ref time (UTC)   : Thu Nov 07 03:10:17 2019
System time      : 0.000021801 seconds slow of NTP time
Last offset      : -0.00021801 seconds
RMS offset       : 0.000106846 seconds
Frequency        : 26.657 ppm slow
Residual freq    : -0.007 ppm
Skew             : 0.093 ppm
Root delay       : 0.013896370 seconds
Root dispersion  : 0.017410904 seconds
Update interval  : 257.5 seconds
Leap status      : Normal
```

The **Ctrl+D** keys can be used to exit interactive mode.

To run `chronyc` in non-interactive mode, the following syntax can be used:

```
chronyc command
```

For example, to display a list of NTP sources being used for `chronyd` in non-interactive mode, use the following command:

```
root@localhost:~# chronyc sources
210 Number of sources = 4
MS Name/IP address          Stratum Poll Reach LastRx Last sample
=====
^_ au.kashra.pictures        2    6   377         4 -2991us[-2991us] +/- 125ms
^* hydrogen.constant.com     2    9   377        205  -756us[ -678us] +/-  33ms
^- lax1.justaguy.be          2    8   377         20 +1559us[+1559us] +/-  98ms
^+ li924-200.members.linode> 2   10   377        671 +1321us[+1394us] +/-  31ms
```

Upon examining the output above, we can see that one of the sources listed, the `hydrogen.constant.com` source, matches the source in the `Reference ID` field in the output of the `tracking` command used previously.

Week 4 Lab 4b Configuring a DHCP server on Linux

DHCP (Dynamic host configuration protocol) used to assign an IP address automatically to Mobile, Laptop, PC, and other network devices so they can communicate. It employs a connectionless service model, using the UDP (User Datagram Protocol). DHCP uses a well-known UDP port 67 for the DHCP Server and the UDP Port 68 for the client.

Depending on implementation, the DHCP server may have three methods of allocating IP addresses:

Dynamic allocation

A network administrator reserves a range of IP addresses for DHCP, and each DHCP client on the LAN is configured to request an IP address from the DHCP server during network initialization. The request-and-grant process uses a lease concept with a controllable time period, allowing the DHCP server to reclaim and then reallocate IP addresses that are not renewed.

Automatic allocation

The DHCP server permanently assigns an IP address to a requesting client from a range defined by an administrator. This is like dynamic allocation, but the DHCP server keeps a table of past IP address assignments, so that it can preferentially assign to a client the same IP address that the client previously had.

Manual allocation

This method is also variously called *static DHCP allocation*, *fixed address allocation*, *reservation*, and *MAC/IP address binding*. An administrator maps a unique identifier (a *client id* or MAC address)

for each client to an IP address, which is offered to the requesting client. DHCP servers may be configured to fall back to other methods if this fails.

Install and configure DHCP Server on Centos8:

Dynamic allocation

1. Check the vm has static IP address 10.0.2.1/24(for our lab). The DHCP server will automatically assign an IP address to the other devices in the network 10.0.2.1/24.
2. Install DHCP server: `yum install dhcp-server`
3. Configuring DHCP Server

The main configuration file of DHCP server is `/etc/dhcp/dhcpd.conf`. The sample of dhcpd.conf file is under `/usr/share/doc/dhcp-server/dhcpd.conf.example`

According to our lab design, add the following subnet to the configuration file, make sure you include the semi-colon at the end of each line.

```
subnet 10.0.2.0 netmask 255.255.255.0 {  
    range 10.0.2.128 10.0.2.254;  
    option routers 10.0.2.1;  
    option domain-name "online.uts.edu.au";  
    option domain-name-servers 10.0.2.1;  
    default-lease-time 60;  
    max-lease-time 600;  
}
```

4. Start the DHCP service: `systemctl start dhcpd.service`, can use `tail -f /var/log/messages` to DHCP process.
5. Set up Window Server vm Ethernet 1 to get IP address automatically.
6. Verify the lease `cat /var/lib/dhcpd/dhcpd.leases`

Manual allocation

7. Reserve IP address on DHCP Server:

If you have a MAC address of a device, you can also bind an IP address with them, for this open up the configuration file `vim /etc/dhcp/dhcpd.conf` and add these following lines at the end of the page to bind an IP address with the specific device.

```
host WinServer {  
    hardware ethernet 00:50:56:8c:20:fd;  
    fixed-address 10.0.2.20;  
}
```

This will bind the IP address **10.0.2.20** with the machine whose MAC address is **00:50:56:8c:20:fd**.(use your Window Server mac address). Remember to restart your dhcpd service after you modify the dhcp configuration file.

8. Verify the Windows server have the reserved IP address.

5a:

User Accounts

There are several text files in the `/etc` directory that contain the account data of the users and groups defined on the system.

The `/etc/passwd` File

The `/etc/passwd` file contains the details of special system accounts (i.e. `bin`, `lp`, `nobody`, etc.) used to run services, and user accounts on the system. Without an entry in the `/etc/passwd` file, users are unable to log in to the system. This file has read access for all users, but write access is limited to the root user.

Each row in this file describes one user account record. Each record in the `passwd` file contains the following seven fields separated by colons:

```
loginID:x:UID:GID:comment:home_directory:login_shell
```

The following example shows the last five lines of a typical `/etc/passwd` file:

```
sysadmin@localhost:~$ tail -5 /etc/passwd
syslog:x:101:103::/home/syslog:/bin/false
bind:x:102:105::/var/cache/bind:/bin/false
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
operator:x:1000:37::/root:/bin/sh
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

Each line contains information pertaining to a single user. The data is separated into fields by colon characters. The following describes each of the fields in detail, from left to right, using the last line of the output of the previous graphic:

- **Name**

```
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

The first field contains the name of the user or the *username*. This name is used when logging in to the system and when file ownership is viewed with the `ls -l` command. It is provided to make it easier for regular users to refer to the account, while the system typically utilizes the *user ID* internally.

- **Password Placeholder**

```
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

At one time, the password for the user was stored in this location, however, now the `x` in this field indicates to the system that the password is in the `/etc/shadow` file.

- **User ID**

```
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

Each account is assigned a *user ID (UID)*. Usernames are not directly used by the system, which typically defines the account by the UID instead. For example, files are owned by UIDs, not by usernames.

Regular user accounts have UID values of greater than 500 (on some systems 1000). The root user has UID value of 0. Typically from UID 1 to UID 499, are called system accounts, and they are designed to provide accounts for services that are running on the system.

- **Primary Group ID**

```
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

This field indicates that the user is a member of that group, which means the user has special permissions on any file that is owned by this group.

- **Comment**

```
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

This field can contain any information about the user, including their real name or other useful information.

- **Home Directory**

```
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

This field defines the location of the user's home directory. For regular users, this would typically be `/home/username`. For example, a username of `bob` would have a home directory of `/home/bob`.

The root user usually has a different place for the home directory, the `/root` directory.

- **Shell**

```
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

This field indicates the location of the user's login shell. By default, the user is placed in this shell whenever they log into a command line environment or open a terminal window. The bash shell `/bin/bash` is the most common shell for Linux users.

An efficient way to check if a specific user has been defined on a system is to search the `/etc/passwd` file using the `grep` command. For example, to see the account information for the user named `sysadmin`, use the following command:

```
sysadmin@localhost:~$ grep sysadmin /etc/passwd
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

/etc/shadow

The `/etc/shadow` file contains account information related to the user's password. However, regular users can't view the contents of the `/etc/shadow` file for security reasons. To view the contents of this file, log in as the administrator (the root account).

Each record in the `/etc/shadow` file contains the following eight fields, separated by colons:

```
loginID:password:lastchg:min:max:warn:inactive:expire
```

A typical `/etc/shadow` file would look similar to the following:

```
root@localhost:~# tail -5 /etc/shadow
syslog*:16874:0:99999:7:::
bind*:16874:0:99999:7:::
sshd*:16874:0:99999:7:::
operator!:16874:0:99999:7:::
sysadmin:$6$c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1l14OvL2mFSIfnc1aU2cQ/221QL5AX
5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050:
```

Again, each line is separated into fields by colon characters. The following describes each of the fields in detail, from left to right, using the last line of the output of the previous graphic:

- **Username**

```
sysadmin:$6$c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1l14OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::
```

This field contains the *username* of the account, which matches the account name in the `/etc/passwd` file.

- **Password**

- sysadmin: \$6\$c75ekQWF\$.GpiZpFnIXLzkALjDpZXmjxZcI1l14OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::

The *password* field contains the encrypted password for the account. This very long string is a one-way encryption, meaning that it can't be "reversed" to determine the original password.

While regular users have encrypted passwords in this field, system accounts have an asterisk * character in this field.

The password field in `/etc/shadow` for the new user is set to `!`, meaning the account is locked by default. Use the `passwd` command to assign a password for the new user account

- **Last Change**

- `sysadmin:6c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1114OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::`

This field contains a number that represents the last time the password was changed. The number 16874 is the number of days since January 1, 1970 (called the Epoch).

This value generates automatically when the user's password is modified. It is used by the *password aging* features provided by the rest of the fields of this file.

- **Minimum**

- `sysadmin:6c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1114OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::`

This field indicates the *minimum* number of days between password changes. It is one of the *password aging* fields; a non-zero value in this field indicates that after a user changes their password, the password can't be changed again for the specified number of days, 5 days in this example. This field is important when the *maximum* field is used.

A value of zero in this field means the user can always change their password.

- **Maximum**

- `sysadmin:6c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1114OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::`

This field indicates the *maximum* number of days the password is valid. It is used to force users to change their passwords on a regular basis. A value of 30 in this field means the user must change their password at least every 30 days to avoid having their account locked out.

Note that if the *minimum* field is set to 0, the user may be able to immediately set their password back to the original value, defeating the purpose of forcing the user to change their password every 30 days. So, if the *maximum* field is set, the *minimum* field is ordinarily set as well.

For example, a *minimum:maximum* of 5:30 means the user must change their password every 30 days and, after changing, the user must wait 5 days before they can change their password again.

If the *max* field is set to 99999, the maximum possible value, then the user essentially never has to change their password (because 99999 days is approximately 274 years).

- **Warn**

- `sysadmin:6c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1114OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::`

If the *maximum* field is set, the *warn* field indicates the number of days before password expiry that the system warns the user. For example, if the *warn* field is set to 7, then any time during the 7 days before the *maximum* time frame is reached, the user will be warned to change their password during the login processes.

The user is only warned at login, so some administrators have taken the approach of setting the *warn* field to a higher value to provide a greater chance of having a warning issued.

If the *maximum* time frame is set to 99999, then the *warn* field is basically useless.

- **Inactive**

- `sysadmin:6c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1l14OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::`

If the user ignores the warnings and exceeds the password timeframe, their account will be locked out. In that case, the *inactive* field provides the user with a "grace" period in which their password can be changed, but only during the login process.

If the *inactive* field is set to 60, the user has 60 days to change to a new password. If they fail to do so, then the administrator would be needed to reset the password for the user.

- **Expire**

- `sysadmin:6c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1l14OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::`

This field indicates the day the account will *expire*, represented by the number of days from January 1, 1970. An expired account is locked, not deleted, meaning the administrator can reset the password to unlock the account.

Accounts with expiration dates are commonly provided to temporary employees or contractors. The account automatically expires after the user's last day of work.

When an administrator sets this field, a tool is used to convert from a real date to an Epoch date. There are also several free converters available on the Internet.

- **Reserved**

- `sysadmin:6c75ekQWF$.GpiZpFnIXLzkALjDpZXmjxZcI1l14OvL2mFSIfnc1aU2cQ/221QL5AX5RjKXpXPJRQ0uVN35TY3/..c7v0.n0:16874:5:30:7:60:15050::`

Currently not used, this field is reserved for future use.

Group Accounts

Your level of access to a system is not determined solely by your user account. Each user can be a member of one or more *groups*, which can also affect the level of access to the system.

The `/etc/passwd` file defines the primary group membership for a user. Supplemental group membership (or secondary group membership) and the groups themselves are defined in the `/etc/group` file.

The `/etc/group` file is another colon-delimited file. The following describes the fields in more detail, using a line that describes a typical group account.

- **Group Name**

```
mail:x:12:mail,postfix
```

This field contains the *group name*. As with usernames, names are more natural for people to remember than numbers. The system typically uses group IDs rather than group names.

- **Password Placeholder**

```
mail:x:12:mail,postfix
```

While there are passwords for groups, they are rarely used in Linux. If the administrator makes a group password, it would be stored in the `/etc/gshadow` file. The `x` in this field is used to indicate that the password is not stored in this file.

- **GID**

```
mail:x:12:mail,postfix
```

Each group is associated with a unique *group ID (GID)* which is placed in this field.

- **User List**

```
mail:x:12:mail,postfix
```

This last field is used to indicate who is a member of the group. While primary group membership is defined in the `/etc/passwd` file, users who are assigned to additional groups would have their username placed in this field of the `/etc/group` file. In this case, the `mail` and `postfix` users are secondary members of the `mail` group.

It is very common for a username to also appear as a group name. It is also common for a user to belong to a group with the same name.

Viewing User Information

The `id` command is used to print user and group information for a specified user.

```
id [options] username
```

When switching between different user accounts, it can be confusing as to which account is currently logged in. When executed without an argument, the `id` command outputs information about the current user, allowing you to confirm your identity on the system.

```
sysadmin@localhost:~$ id
uid=1001(sysadmin) gid=1001(sysadmin) groups=1001(sysadmin),4(adm),27(sudo)
```

The output of the `id` command always lists the user account information first, using the *user ID* and *username* first:

```
uid=1001(sysadmin) gid=1001(sysadmin) groups=1001(sysadmin),4(adm),27(sudo)
```

After the username the *primary group* is listed, denoted by both the *group ID* and *group name*:

```
uid=1001(sysadmin) gid=1001(sysadmin) groups=1001(sysadmin),4(adm),27(sudo)
```

Other information listed includes the groups the user belongs to, again denoted by group IDs followed by the group names. The user shown belongs to three groups:

```
uid=1001(sysadmin) gid=1001(sysadmin) groups=1001(sysadmin),4(adm),27(sudo)
```

If the command is given a username as an argument, such as `root`, it displays information about the specified account:

```
sysadmin@localhost:~$ id root
uid=0(root) gid=0(root) groups=0(root)
```

To print only the user's primary group, use the `-g` option:

```
sysadmin@localhost:~$ id -g
1001
```

The `id` command can also be used to verify the user's secondary group memberships using the `-G` option. This will print all the groups that a user belongs to, both primary and secondary.

```
sysadmin@localhost:~$ id -G
1001 4 27
```

The output of the previous example aligns with the contents of the `/etc/group` file, as a search for `sysadmin` reveals:

```
sysadmin@localhost:~$ cat /etc/group | grep sysadmin
adm:x:4:syslog,sysadmin
sudo:x:27:sysadmin
sysadmin:x:1001:
```

Creating A Group

The `groupadd` command can be executed by the root user to create a new group. The command requires only the name of the group to be created. The `-g` option can be used to specify a group id for the new group:

```
root@localhost:~# groupadd -g 1005 research
root@localhost:~# grep research /etc/group
```

```
research:x:1005:
```

If the `-g` option is not provided, the `groupadd` command will automatically provide a GID for the new group. To accomplish this, the `groupadd` command looks at the `/etc/group` file and uses a number that is one value higher than the current highest GID number. The execution of the following commands illustrates this:

```
root@localhost:~# groupadd development
root@localhost:~# grep development /etc/group
development:x:1006:
```

To change the group name, execute the following command:

```
root@localhost:~# groupmod programmers -n c-programmers
root@localhost:~# tail -1 /etc/group
c-programmers:x:1990:
```

The `groupdel` command is used to delete a group that is no longer needed. For example, to delete the group named `programmers`, execute the following command:

```
root@localhost:~# groupdel programmers
```

An error message is displayed when the group being deleted is the primary group of any existing user:

```
root@localhost:~# groupdel sysadmin
groupdel: cannot remove the primary group of user `sysadmi
```

Creating a User

Creating a new user is a two-step process:

1. Create a new account.
2. Set a password for the new account.

The `useradd` command, by itself, creates a new account using default options. To create a new user with options different than the defaults, use the `useradd` command with optional parameters:

```
useradd -s <Shell> -d <Home directory> -m -k <Skeleton directory> -g <Group> username
```

Option	Meaning
--------	---------

-s	User's default login shell. The default on most Linux systems is <code>/bin/bash</code> . Other popular alternatives are: <code>/bin/tsh</code> , <code>/bin/dash</code> , <code>/bin/zsh</code> , and <code>/bin/ksh</code> .
-d	The home directory for the new user. The default on most Linux systems is <code>/home/username</code> .
-m	Creates the home directory for the user if it does not exist.
-k	Copy initialization files from an alternative directory. The default is <code>/etc/skel</code> .
-g	Group name or number of the user. The default group varies depending on the Linux distribution.
-N	Avoids creation of a group with the same name as the user name. An alternative group should be specified with the <code>-g</code> option.

```
root@localhost:~# useradd -s /bin/bash -d /home/qa_user1 -m -g team qa_user1
root@localhost:~# tail -1 /etc/passwd
qa_user1:x:1006:1004::/home/qa_user1:/bin/bash
root@localhost:~# tail -1 /etc/shadow
qa_user1:!:18162:0:99999:7:::
root@localhost:~# grep team /etc/group
team:x:1004:
```

Another example of a `useradd` command using a few options looks like the following:

```
root@localhost:~# useradd -u 1009 -g users -G sales,research -m -c 'Jane Doe' jane
```

This example of the `useradd` command creates a user with a UID of 1009, a primary group of users, supplementary memberships in the sales and research groups, a comment of "Jane Doe", and an account name of jane.

After executing the previous command, the information about the jane user account is automatically added to the `/etc/passwd` and `/etc/shadow` files, while the information about supplemental group access is automatically added to the `/etc/group` and `/etc/gshadow` files:


```
root@localhost:~# grep jane /etc/passwd
jane:x:1009:100:Jane Doe:/home/jane:/bin/sh
root@localhost:~# grep jane /etc/shadow
jane:!:17883:0:99999:7:30::
root@localhost:~# grep jane /etc/group
research:x:1005:jane
sales:x:999:jane
root@localhost:~# grep jane /etc/gshadow
research:!:jane
sales:!:jane
```

Note that the account doesn't have a valid password yet!

In addition, if the `CREATE_MAIL_SPOOL` is set to `yes` in the `/etc/login.defs` file then the mail spool file `/var/spool/mail/jane` is created:

```
root@localhost:~# ls /var/spool/mail
jane root rpc sysadmin
```

Finally, because the `-m` option is used, the `/home/jane` directory is created with permissions only permitting the `jane` user access, and the contents of the `/etc/skel` directory would be copied into the directory:

```
root@localhost:~# ls /home
jane sysadmin
```

Skeleton Directory

By default, the contents of the `/etc/skel` directory are copied into the new user's home directory. The resulting files are also owned by the new user. By using the `-k` option with the `useradd` command, the contents of a different directory can be used to populate a new user's home directory. When specifying the skeleton directory with the `-k` option, the `-m` option must be used or else the `useradd` command will fail with an error.

Some of the commonly found files in this directory are:

- `.bashrc` - contains alias definitions, enables shell features, etc.
- `.profile` - contains variable definitions and one-time-only commands to set the user's environment
- `.bash_logout` - contains actions to take when logging out (i.e., clear the screen, etc.)

The following example uses `/home/sysadmin` as the skeleton directory:

```
root@localhost:~# useradd -mk /home/sysadmin jane
root@localhost:~# ls /home/jane
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
```

The `/etc/login.defs` file contains values that are applied by default to new users you create with the `useradd` command.

This file contains many comments and blank lines, so to only view lines that are not comments or blank lines (the real configuration settings), then you can use the following `grep` command:

```
root@localhost:~# grep -Ev '^#|^$' /etc/login.defs
MAIL_DIR /var/mail/spool
PASS_MAX_DAYS      99999
PASS_MIN_DAYS      0
PASS_MIN_LEN       5
PASS_WARN_AGE      7
UID_MIN             500
UID_MAX             60000
GID_MIN             500
GID_MAX             60000
CREATE_HOME         yes
UMASK               077
USERGROUPS_ENAB     yes
ENCRYPT_METHOD       SHA512
MD5_CRYPT_ENAB     no
```

The `-D` option to the `useradd` command allows you to view or change some of the default values used by the `useradd` command. The values shown by `useradd -D` can also be viewed or updated by manipulating the `/etc/default/useradd` file:

```
root@localhost:~# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

Setting a User Password

There are several ways for a user password to be changed. The user can execute the `passwd` command, the administrator can execute the `passwd` command providing the username as an argument, or graphical tools are also available.

The administrator can use the `passwd` command to either set the initial password or change the password for the account. For example, if the administrator had created the account `jane`, then executing `passwd jane` provides the administrator a prompt to set the password for the `jane` account. If completed successfully, then the `/etc/shadow` file will be updated with the user's new password.

While regular users must follow many password rules, the root user only needs to follow one rule: the password cannot be left blank. When the root user violates all other password rules that normally apply to regular users, it results in a warning being printed to the screen and the rule not being enforced:

```
root@localhost:~# passwd Jane
Enter new UNIX password:
BAD PASSWORD: it is WAY to short
BAD PASSWORD: is too simple
Retype new UNIX password:
```

If the user wants to view status information about their password, then they can execute the following command:

```
sysadmin@localhost:~$ passwd -S sysadmin

sysadmin P 04/24/2019 0 99999 7 -1
```

The root user can lock and unlock a user's account by using the `passwd -l username` and `passwd -u username` commands, respectively.

When an account is locked, the password entry in the `/etc/shadow` file is made invalid by prepending a `!` or `!!` to the encrypted password. To unlock the entry, the `!` or `!!` characters are removed.

```
root@localhost:~# grep sysadmin /etc/shadow
sysadmin:$6$8.BhF.gs$gVnGSxU/JM4mcKgdbvgIOPWGT5XN25cZwEyI4W3/Nvh0p14UrUrSuvqp
3SpPBBq9xYGpmbCAYPHfLgYLpnmL81:18010:0:99999:7:::

root@localhost:~# passwd -l sysadmin
passwd: password expiry information changed.

root@localhost:~# grep sysadmin /etc/shadow
sysadmin:!!$6$8.BhF.gs$gVnGSxU/JM4mcKgdbvgIOPWGT5XN25cZwEyI4W3/Nvh0p14UrUrSuvq
p3SpPBBq9xYGpmbCAYPHfLgYLpnmL81:18010:0:99999:7:::

root@localhost:~# passwd -u sysadmin
```

```
passwd: password expiry information changed.  
root@localhost:~# grep sysadmin /etc/shadow  
sysadmin:$6$8.BhF.gs$gVnGSxU/JM4mcKgdbvgIOPWGT5XN25cZwEyI4W3/
```

Managing Password Aging

The `chage` command provides many options for managing the password aging information found in the `/etc/shadow` file.

Here's a summary of the `chage` options:

Short Option	Long Option	Description
<code>-l</code>	<code>--list</code>	List the account aging information
<code>-d LAST_DAY</code>	<code>--lastday LAST_DAY</code>	Set the date of the last password change to <code>LAST_DAY</code>
<code>-E EXPIRE_DATE</code>	<code>--expiredate EXPIRE_DATE</code>	Set account to expire on <code>EXPIRE_DATE</code> in the <code>YYYY-MM-DD</code> format
<code>-h</code>	<code>--help</code>	Show the help for the <code>chage</code> command
<code>-I INACTIVE</code>	<code>--inactive INACTIVE</code>	Set account to permit login for <code>INACTIVE</code> days after password expires
<code>-m MIN_DAYS</code>	<code>--mindays MIN_DAYS</code>	Set the minimum number of days before the password can be changed to <code>MIN_DAYS</code>
<code>-M MAX_DAYS</code>	<code>--maxdays MAX_DAYS</code>	Set the maximum number of days before a password should be changed to <code>MAX_DAYS</code>
<code>-W WARN_DAYS</code>	<code>--warndays WARN_DAYS</code>	Set the number of days before a password expires to start displaying a warning to <code>WARN_DAYS</code>

A good example of the `chage` command would be to change the maximum number of days that an individual's password is valid to be 60 days:

```
root@localhost:~# chage -M 60 jane
root@localhost:~# grep jane /etc/shadow
```

Modifying a User

The `usermod` command offers many options for modifying an existing user account. Many of these options are also available with the `useradd` command at the time the account is created. The following chart provides a summary of the `usermod` options:

Short Option	Long Option	Description
<code>-c</code>	<code>COMMENT</code>	Sets the value of the comment field to <code>COMMENT</code> .
<code>-d HOME_DIR</code>	<code>--home HOME_DIR</code>	Sets <code>HOME_DIR</code> as a new home directory for the user.
<code>-e EXPIRE_DATE</code>	<code>--expiredate EXPIRE_DATE</code>	Set account expiration date to <code>EXPIRE_DATE</code> .
<code>-f INACTIVE</code>	<code>--inactive INACTIVE</code>	Set account to permit login for <code>INACTIVE</code> days after password expires.
<code>-g GROUP</code>	<code>--gid GROUP</code>	Set <code>GROUP</code> as the primary group.
<code>-G GROUPS</code>	<code>--groups GROUPS</code>	Set supplementary groups to a list specified in <code>GROUPS</code> .
<code>-a</code>	<code>--append</code>	Append the user's supplemental groups with those specified by the <code>-G</code> option.
<code>-h</code>	<code>--help</code>	Show the help for the <code>usermod</code> command.
<code>-l NEW_LOGIN</code>	<code>--login NEW_LOGIN</code>	Change the user's login name.

Short Option	Long Option	Description
<code>-L</code>	<code>--lock</code>	Lock the user account.
<code>-s SHELL</code>	<code>--shell SHELL</code>	Specify the login shell for the account.
<code>-u NEW_UID</code>	<code>--uid NEW_UID</code>	Specify the user's UID to be <code>NEW_UID</code> .
<code>-U</code>	<code>--unlock</code>	Unlock the user account.

Several of these options are worthy of discussion because of how they impact user management. It can be very problematic to change the user's UID with the `-u` option, as any files owned by the user will be orphaned. On the other hand, specifying a new login name for the user with `-l` does not cause the files to be orphaned.

Deleting a user with the `userdel` command can either orphan or remove the user's files on the system. Instead of deleting the account, another choice is to lock the account with the `-L` option for the `usermod` command. Locking an account prevents the account from being used, but ownership of the files remains.

There are some important things to know about managing the supplementary groups. If you use the `-G` option without the `-a` option, then you must list all the groups to which the user would belong. Using the `-G` option alone can lead to accidentally removing a user from all the former supplemental groups that the user belonged to.

If you use the `-a` option with `-G` then you only have to list the new groups to which the user would belong. For example, if the user `jane` currently belongs to the `sales` and `research` groups, then to add her account to the `development` group, execute the following command:

```
root@localhost:~# usermod -aG development jane
```

Deleting a User

The `userdel` command is used to delete users. When you delete a user account, you also need to decide whether to delete the user's home directory. The user's files may be important to the organization, and there may even be legal requirements to keep the data for a certain amount of time, so be careful not to make this decision lightly. Also, unless you've made backup copies of the data, once you've executed the command to delete the user and their files, there is no reversing the action.

To delete the user `jane` without deleting the user's home directory `/home/jane`, execute:

```
root@localhost:~# userdel jane
```

Beware that deleting a user without deleting their home directory means that the user's home directory files will be orphaned and these files will be owned solely by their former UID and GID.

To delete the user, home directory, and mail spool as well, use the `-r` option:

```
root@localhost:~# userdel -r jane
```

WARNING: The above command will only delete the user's files in their home directory and their mail spool. If the user owns other files outside of their home directory, then those files will continue to exist as orphaned files.

5b:

Switching Users

The `su` command allows you to run a shell as a different user. While switching to the `root` user is what the `su` command is used for most frequently, it can also switch to other users as well.

```
su [options] [username]
```

When switching users utilizing the *login shell* option is recommended, as the login shell fully configures the new shell with the settings of the new user, ensuring any commands executed run correctly. If this option is omitted, the new shell changes the UID but doesn't fully log in the user. The login shell option can be specified one of three ways:

```
su -  
su -l  
su --login
```

By default, if a username is not specified, the `su` command opens a new shell as the `root` user. The following two commands are equivalent ways to start a shell as the `root` user:

```
su - root  
su -
```

After pressing **Enter** to execute either one of these commands, the user must provide the password of the `root` user to start the new shell. If you don't know the password of the account that you are shifting to, then the `su` command will fail.

Notice in the example below the command prompt changes to reflect the current user.

```
sysadmin@localhost:~$ su -  
Password:  
root@localhost:~# id  
uid=0(root) gid=0(root) groups=0(root)
```

After using the shell started by the `su` command to perform the necessary administrative tasks, return to your original shell (and original user account) by using the `exit` command.

```
root@localhost:~# exit  
logout  
sysadmin@localhost:~$ id  
uid=1001(sysadmin) gid=1001(sysadmin) groups=1001(sysadmin),4(adm),27(sudo)
```

Executing Privileged Commands

A normal user on Linux will have many restrictions in the system. For example — the user won't be able to start a service, change a config file or execute a script on the system by default.

The `sudo` command allows users to execute commands as another user. Similar to the `su` command, the `root` user is assumed by default.

```
sudo [options] command
```

For the user to be able to use the `sudo` command, the user name need to be listed in Sudoers File.

The word Sudo is the abbreviation of the term “Super User Do”. The users with Sudo permission are called as Sudo Users. Management of all these sudo users is done through a file called as Sudoers File. We can use `visudo` to edit this file. The syntax to give a user an administration privilege:

User <space> OnHost = (Runas-User:Group) <space> Commands

Example: root ALL = (ALL:ALL) ALL

In the example below we are going to add the following field in the configuration file.

`sysadmin ALL=(root) /usr/bin/head /etc/shadow` Now user `sysadmin` can run the `head` command as the `root` user. It prompts for the password of the `sysadmin` user:

```
sysadmin@localhost:~$ sudo head /etc/shadow  
[sudo] password for sysadmin:  
root:$6$4Yga95H9$8HbxqsMEIBTZ0YomlMffYCV9VE1SQ4T2H3SHXw41M02SQtfAdDVE9mqGp2hr  
20q.ZuncJpLyWkYwQdKlSJyS8.:16464:0:99999:7:::  
daemon*:16463:0:99999:7:::  
bin*:16463:0:99999:7:::
```



```
sys:*:16463:0:99999:7:::
sync:*:16463:0:99999:7:::
games:*:16463:0:99999:7:::
man:*:16463:0:99999:7:::
lp:*:16463:0:99999:7:::
mail:*:16463:0:99999:7:::
news:*:16463:0:99999:7:::
```

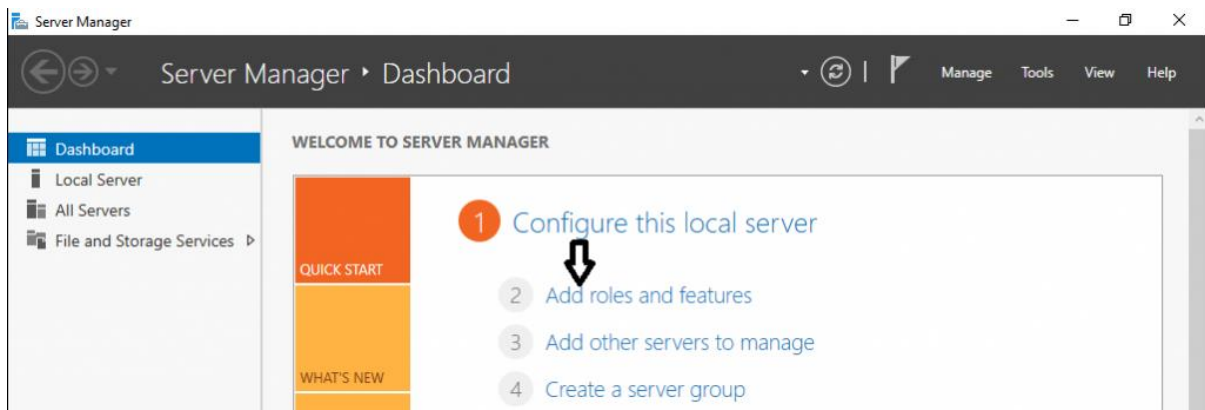
Install and Configure DNS on Windows Server 2019

Prerequisites

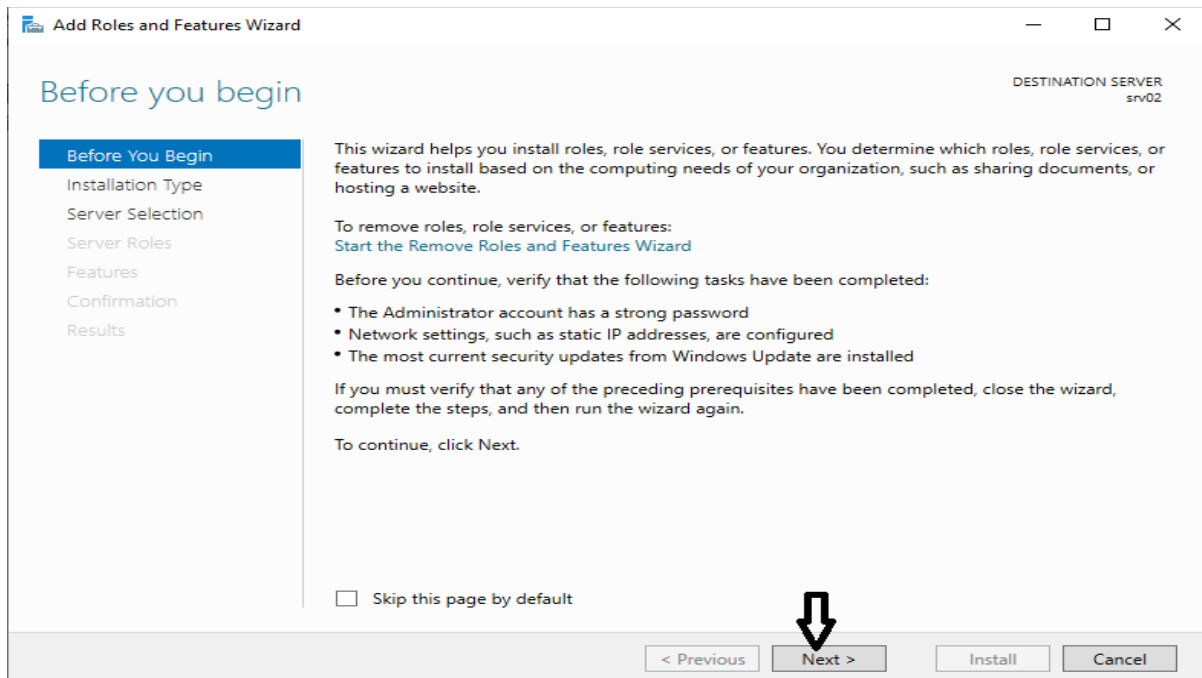
1. Server OS (Windows server 2019)
2. Administrator Account
3. A Static IP address is configured(We are going to use IP address 10.0.2.2 in our lab)

Install Windows DNS Server

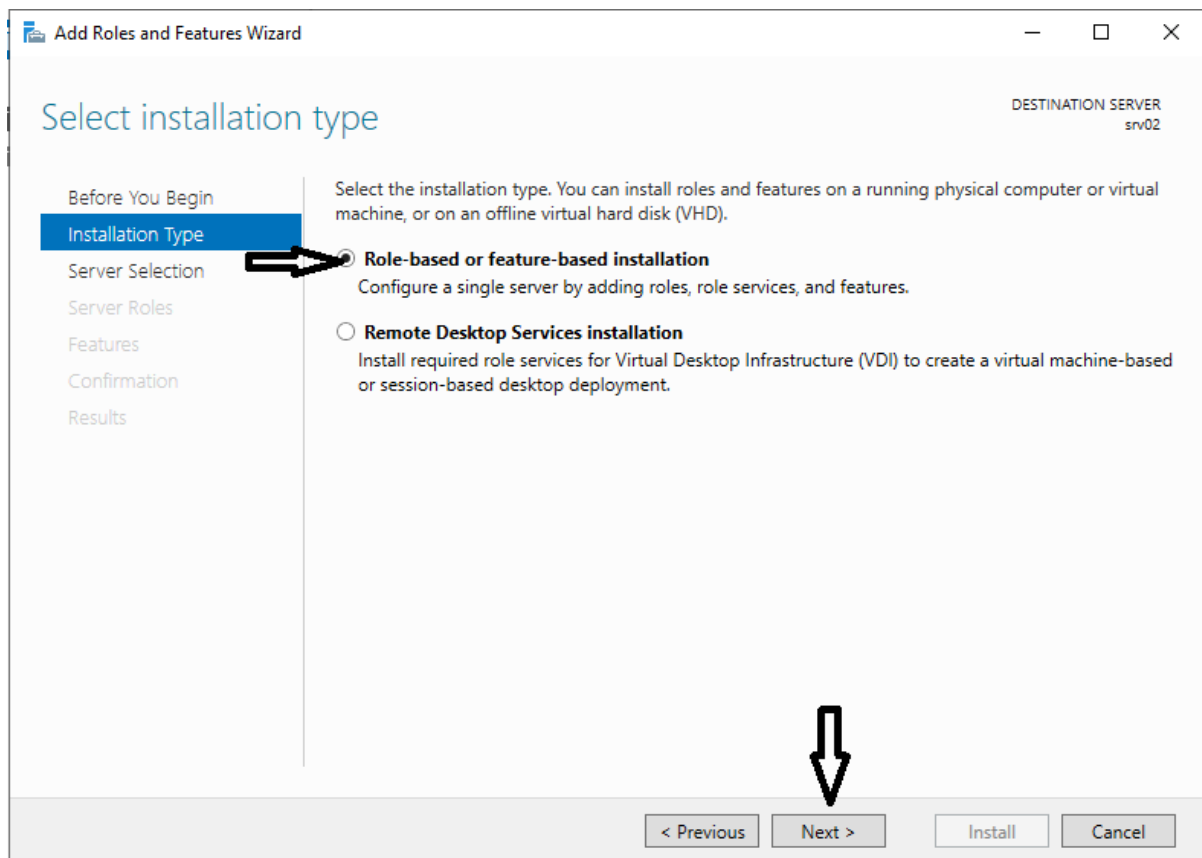
Step1: Open the server manager dashboard and Click add roles and features.



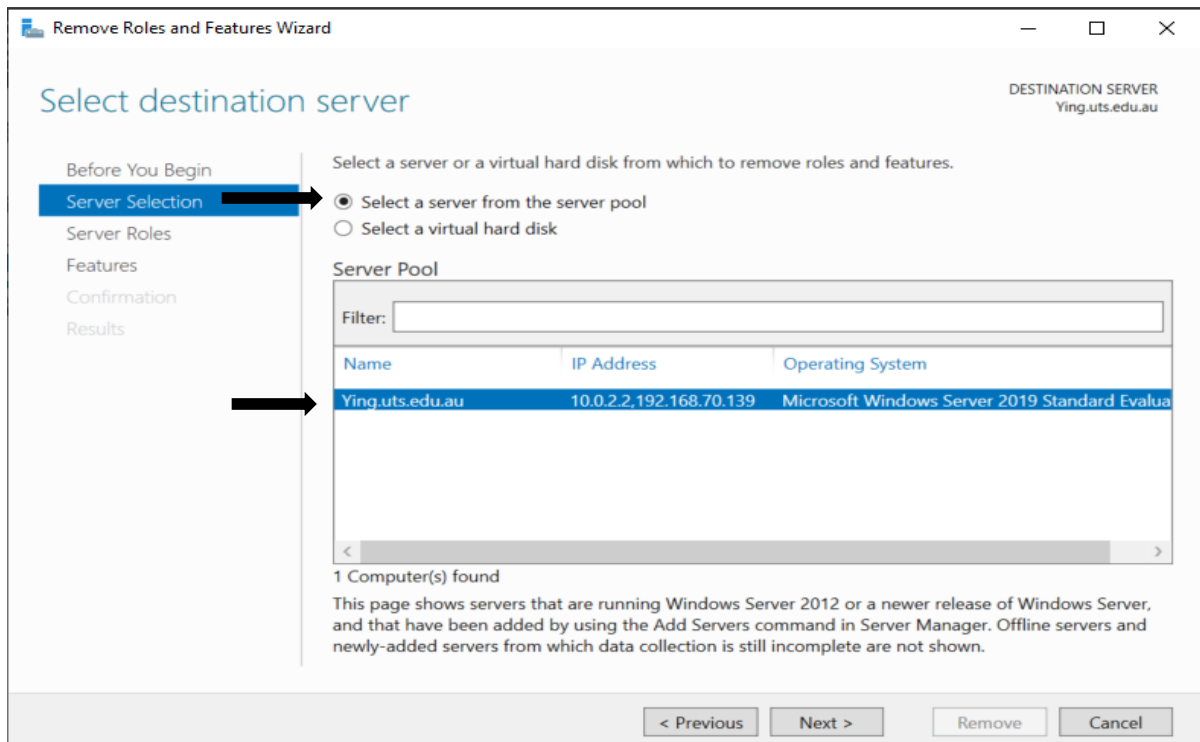
Step 2: Click on Next



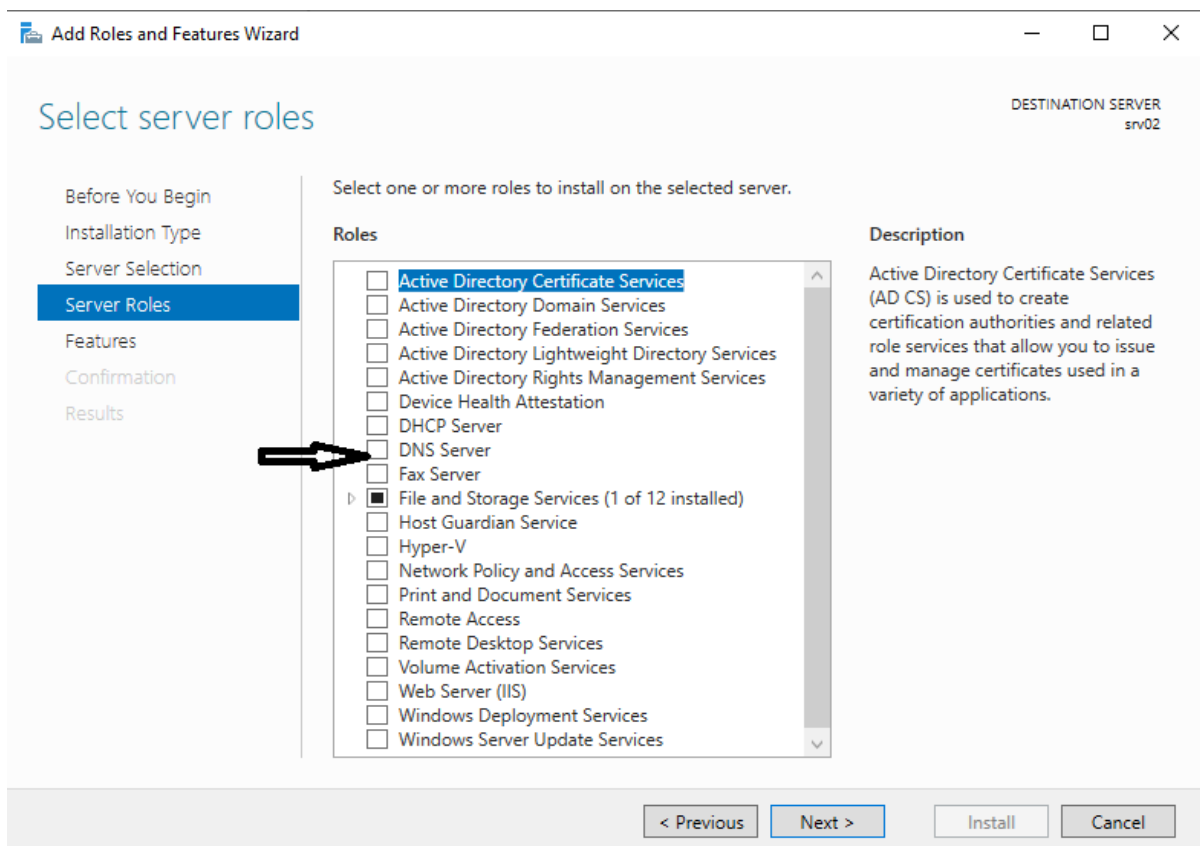
Step 3: Choose Role-based or feature-based installation and click Next.



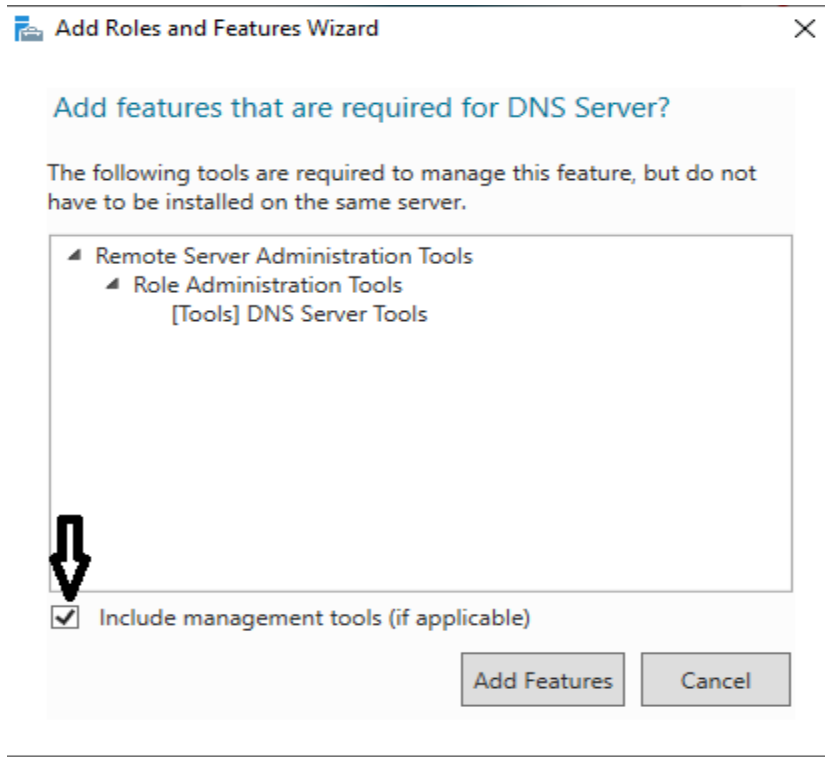
Step 4: Choose destination server for DNS role and click Next



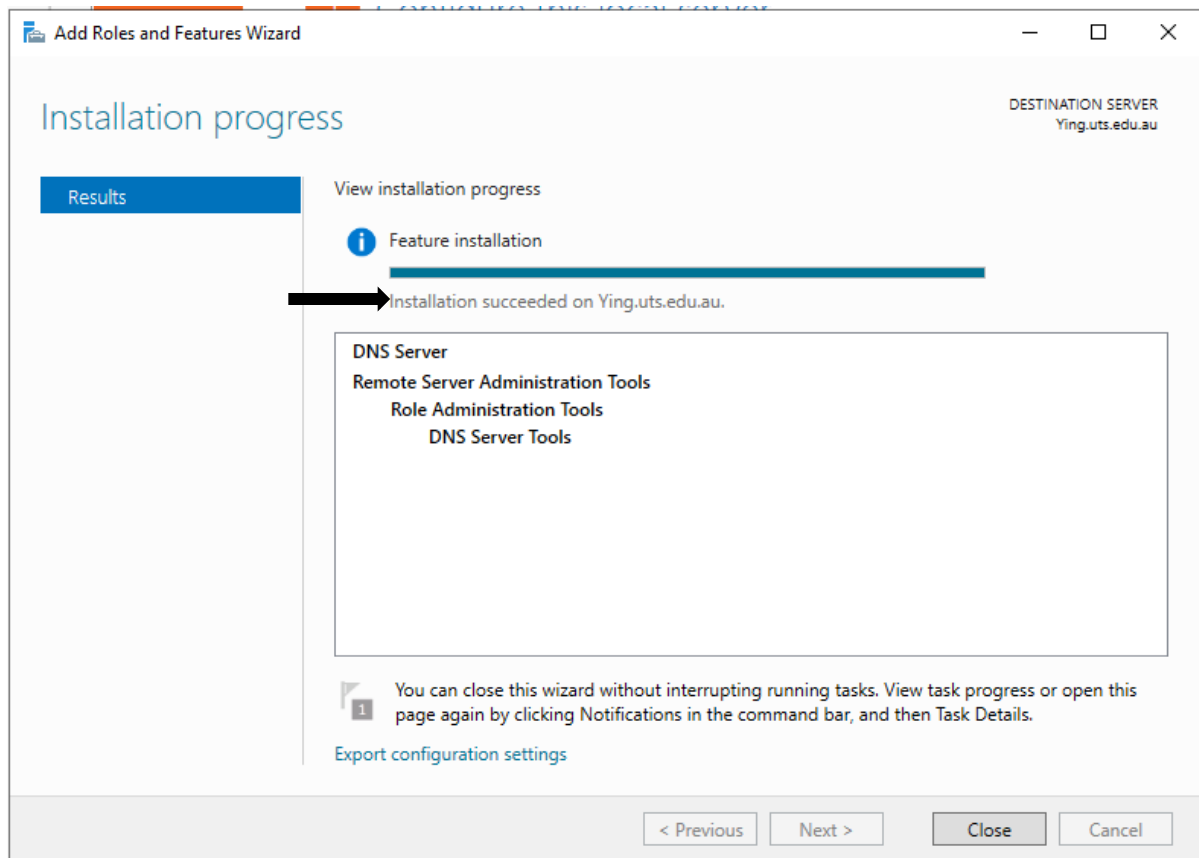
Step 5: Put the Check on the DNS server for DNS Server Role Installation



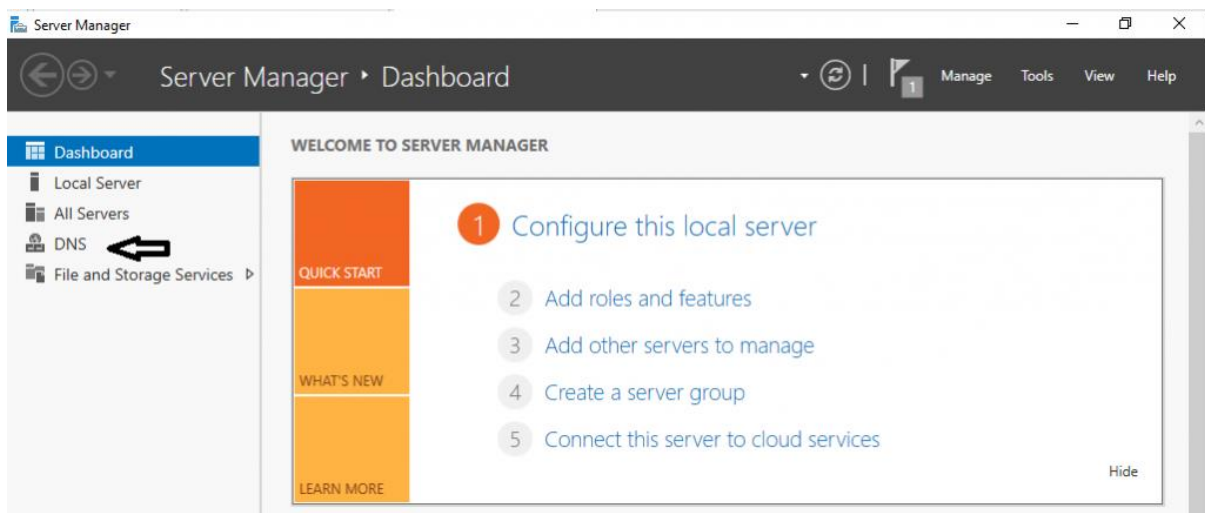
Step 6: Add features that are required for the DNS server including management tools.



Step 7: After adding of required features it installs the DNS server with a succeeded message.

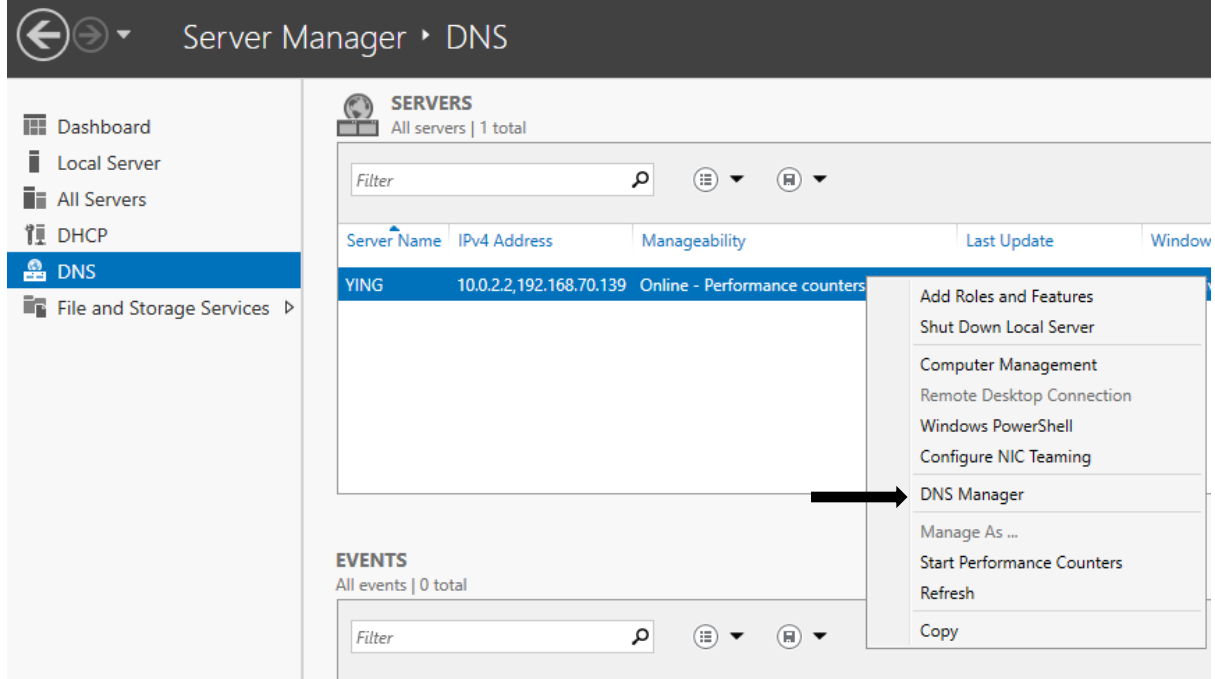


Step 8: After DNS Server Installation you can access the DNS server from Server Manager or from Administrative Tools option in windows Program.

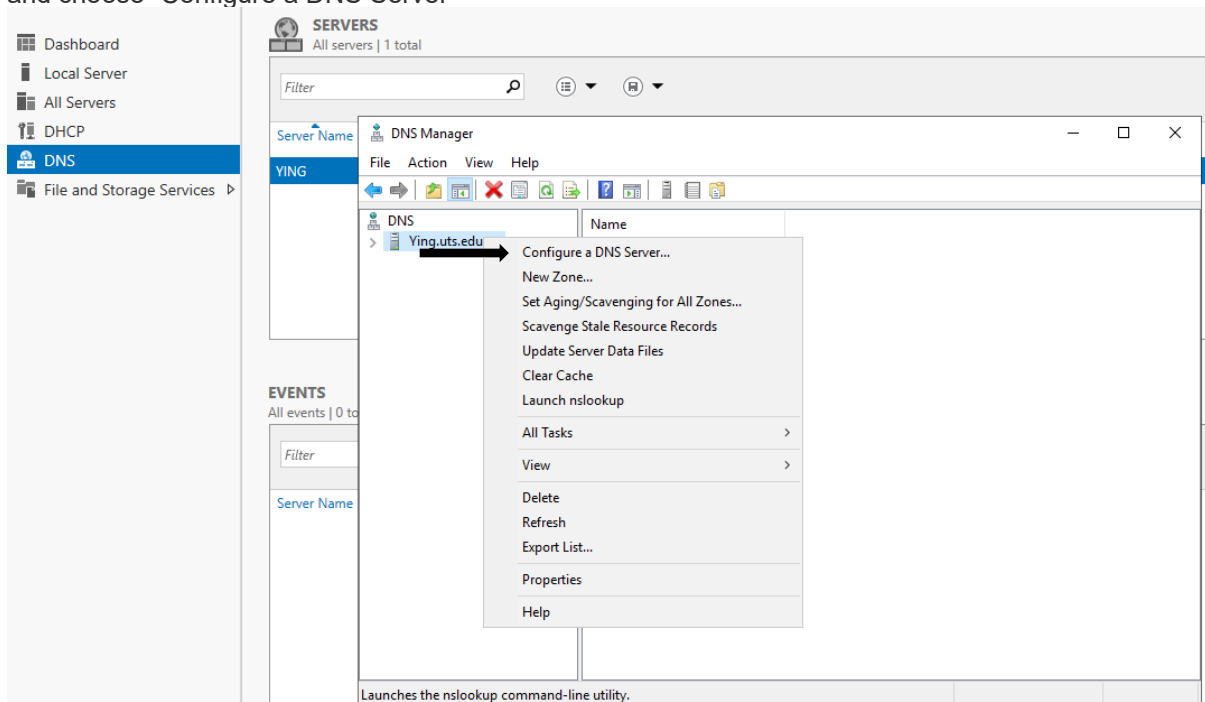


Creating Forward Lookup Zone

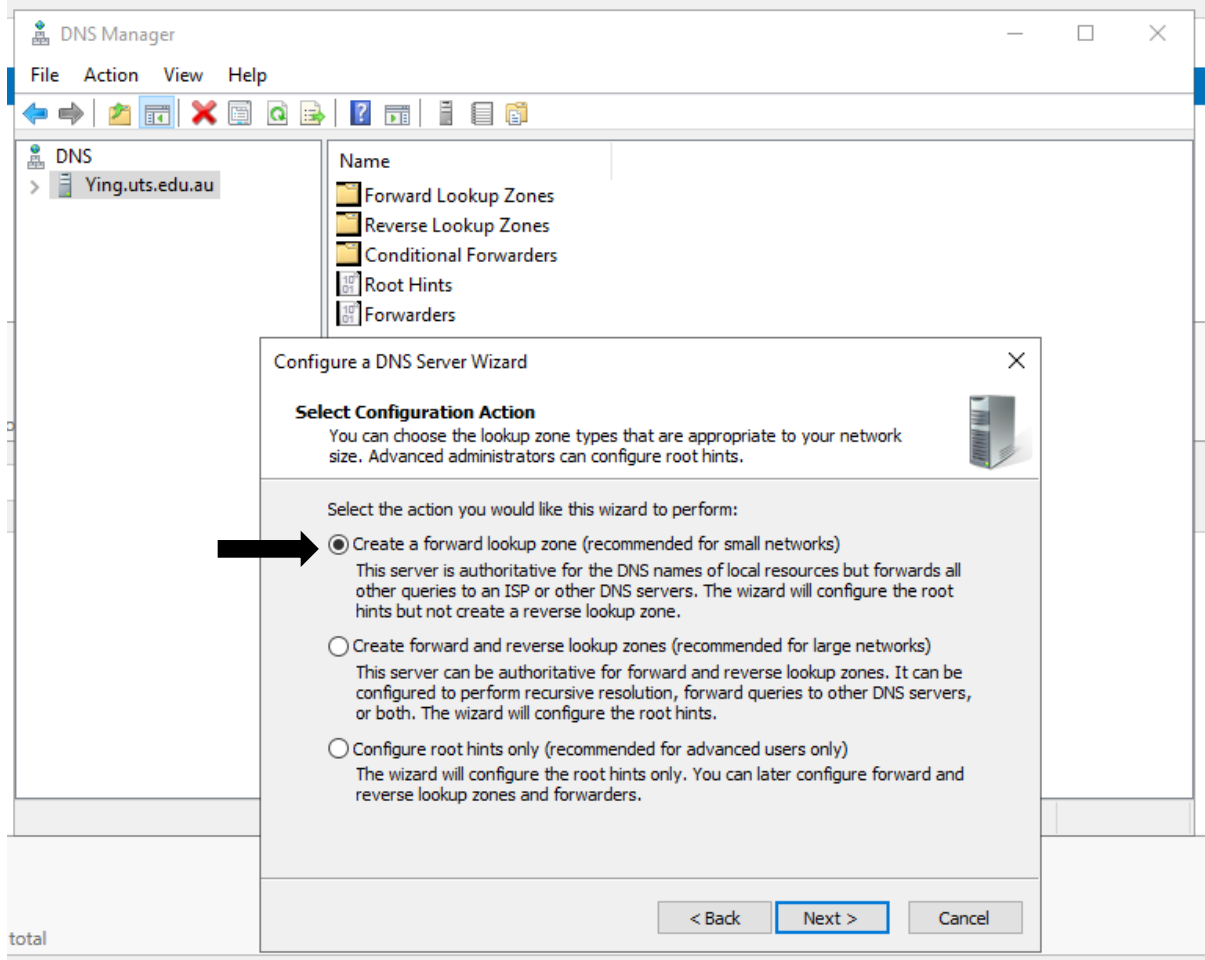
Step 1. Open DNS management console from server manager or from Windows Administrative Tools.



Step 2. In DNS manager console, look for your server's name on the left. Right-click on the server's name and choose "Configure a DNS Server"



Step3 Follow the DNS Server configuration wizard, We will choose “Create a forward lookup zone” .



We choose “This server maintains the zone”.

Configure a DNS Server Wizard

Primary Server Location
You can choose where the DNS data is maintained for your network resources.

Which DNS server maintains your primary forward lookup zone?

☒ This server maintains the zone
The wizard will help you create a primary forward lookup zone.

☐ An ISP maintains the zone, and a read-only secondary copy resides on this server
The wizard will help you create a secondary forward lookup zone.

< Back Next > Cancel

We will name our zone as "netserv.edu.au".

New Zone Wizard

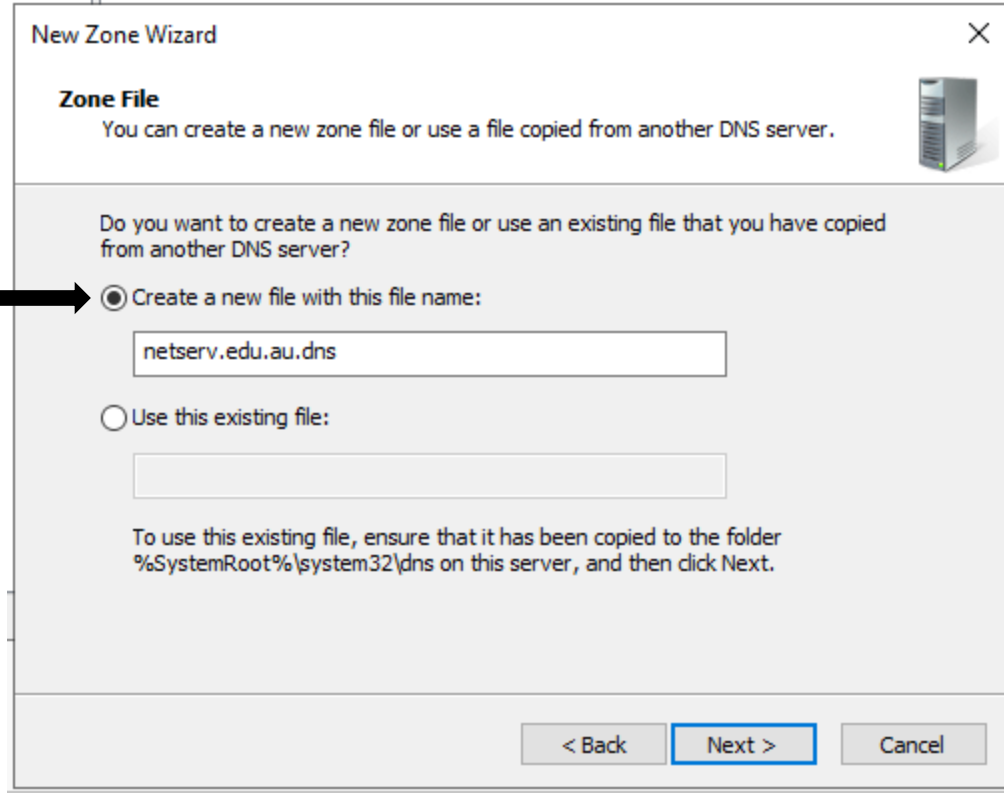
Zone Name
What is the name of the new zone?

The zone name specifies the portion of the DNS namespace for which this server is authoritative. It might be your organization's domain name (for example, microsoft.com) or a portion of the domain name (for example, newzone.microsoft.com). The zone name is not the name of the DNS server.

Zone name:
netserv.edu.au

< Back Next > Cancel

A new zone file "netserv.edu.au.dns" will be created in C:\Windows\System32\dns directory.



The image shows a Windows 'New Zone Wizard' dialog box. The title bar says 'New Zone Wizard' with a close button. Below the title bar, the section is titled 'Zone File' with a sub-header 'You can create a new zone file or use a file copied from another DNS server.' and a server icon. The main area asks 'Do you want to create a new zone file or use an existing file that you have copied from another DNS server?'. There are two radio buttons: 'Create a new file with this file name:' (which is selected and pointed to by a black arrow) and 'Use this existing file:'. The first option has a text box containing 'netserv.edu.au.dns'. The second option has an empty text box. Below these, a note says 'To use this existing file, ensure that it has been copied to the folder %SystemRoot%\system32\dns on this server, and then click Next.' At the bottom are three buttons: '< Back', 'Next >' (highlighted with a blue border), and 'Cancel'.

New Zone Wizard

Zone File
You can create a new zone file or use a file copied from another DNS server.

Do you want to create a new zone file or use an existing file that you have copied from another DNS server?

☒ Create a new file with this file name:

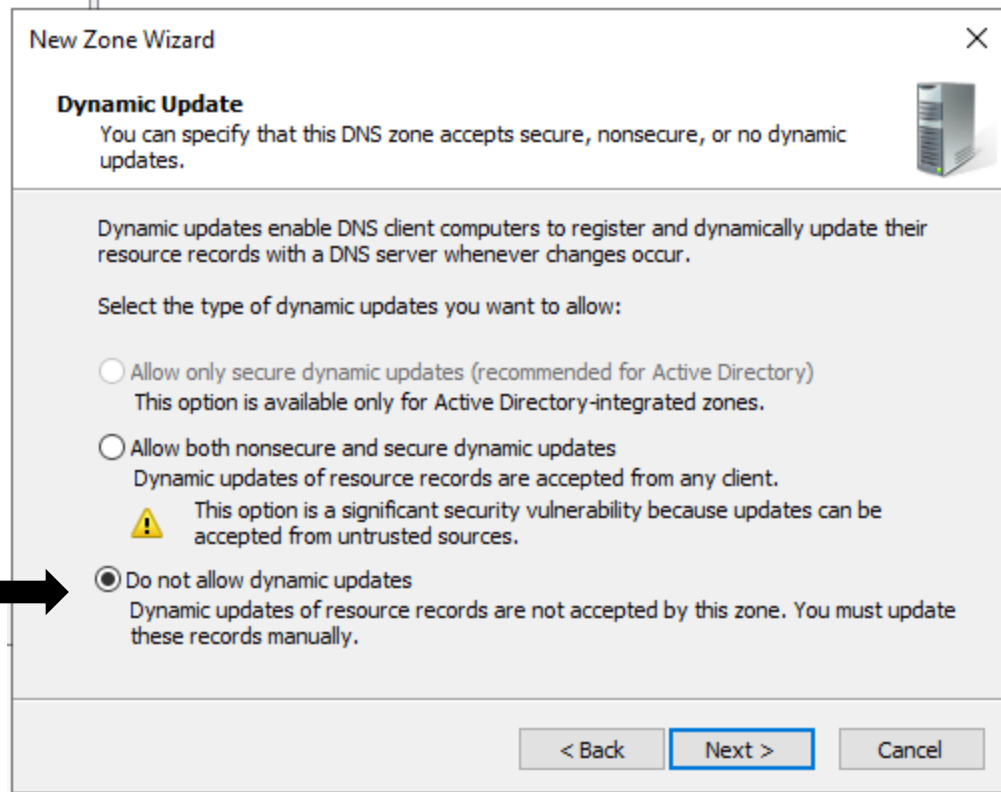
netserv.edu.au.dns

☐ Use this existing file:

To use this existing file, ensure that it has been copied to the folder %SystemRoot%\system32\dns on this server, and then click Next.

< Back Next > Cancel

Do not allow dynamic updates for our zone.



Use `ipconfig /all` to check your VM's DNS Server IP address for Ethernet0, configure this IP address as forwarder, my VM use 192.168.70.2, your DNS server IP address can be different.

Configure a DNS Server Wizard

Forwarders
Forwarders are DNS servers to which this server sends queries that it cannot answer.

Should this DNS server forward queries?

☒ Yes, it should forward queries to DNS servers with the following IP addresses:

IP Address	Server FQDN	Validated
192.168.70.2		

☐ No, it should not forward queries
If this server is not configured to use forwarders, it can still resolve names using root name servers.

< Back Next > Cancel

We want to see the green tick for this forwarder address.

Configure a DNS Server Wizard

Forwarders
Forwarders are DNS servers to which this server sends queries that it cannot answer.

Should this DNS server forward queries?

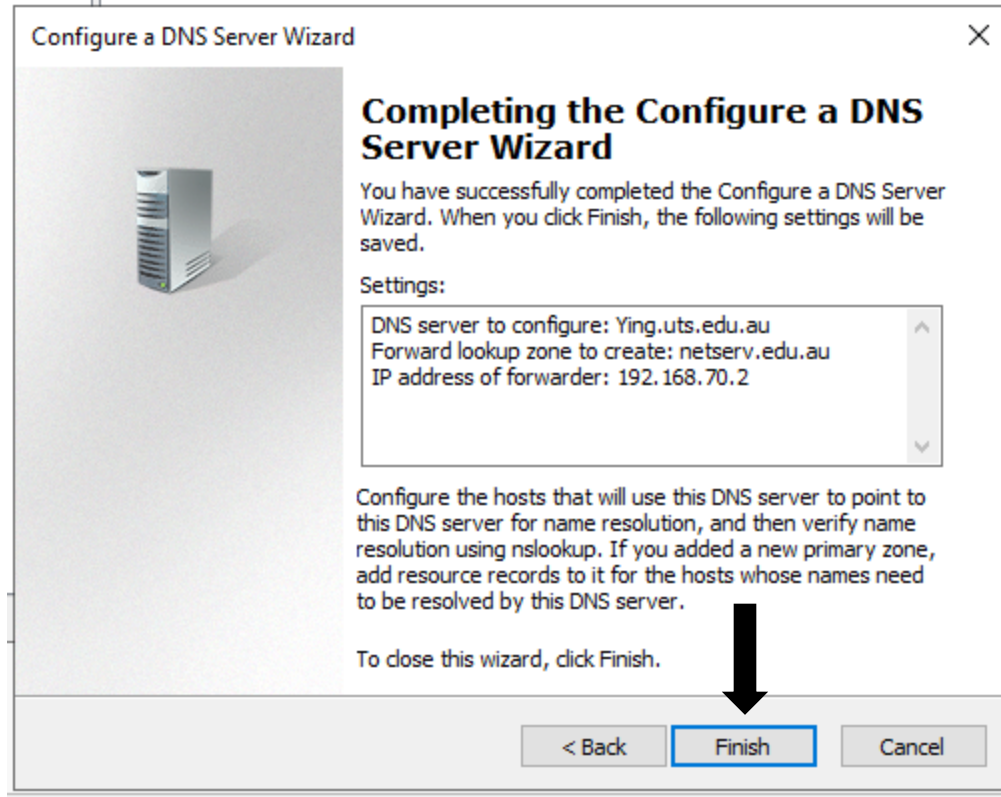
☒ Yes, it should forward queries to DNS servers with the following IP addresses:

IP Address	Server FQDN	Validated
<Click here to add an IP Address or DNS Name>		
✓ 192.168.70.2	au-wa-2481-ler-0...	OK

☐ No, it should not forward queries
If this server is not configured to use forwarders, it can still resolve names using root name servers.

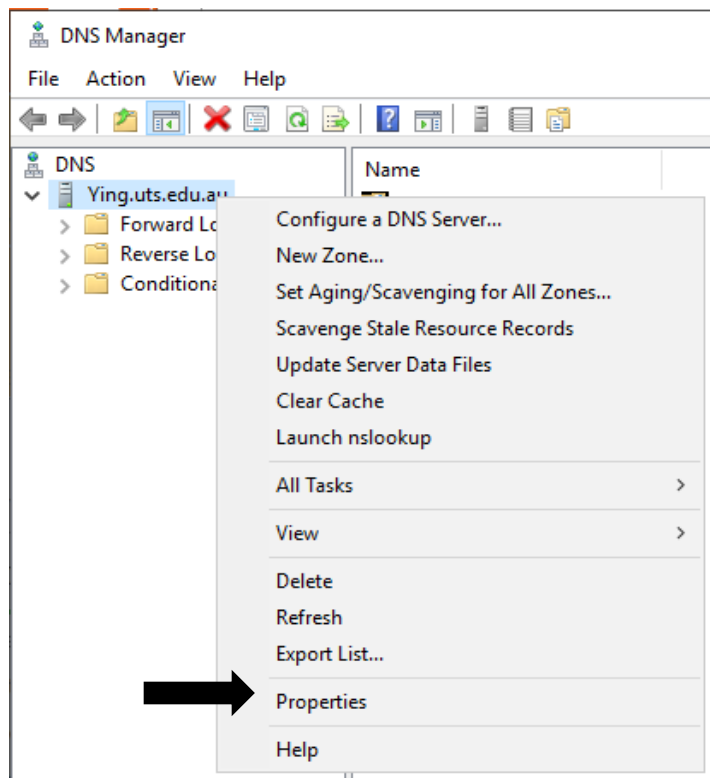
< Back Next > Cancel

Finish the DNS server configuration wizard.

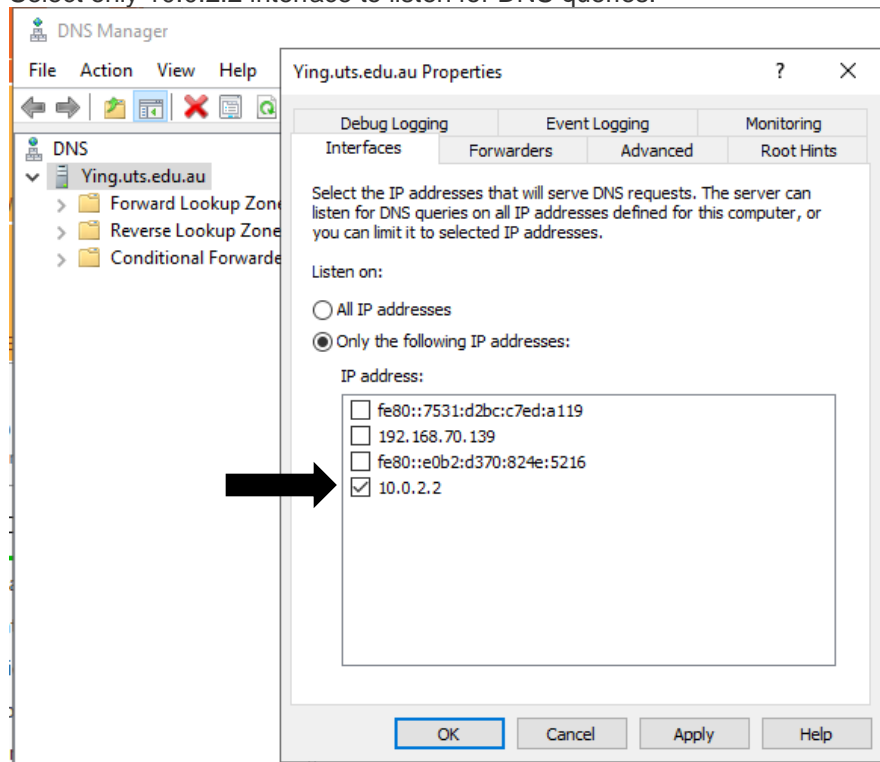


Step 4 Set up DNS server properties

Right click the DNS server's name, choose properties.

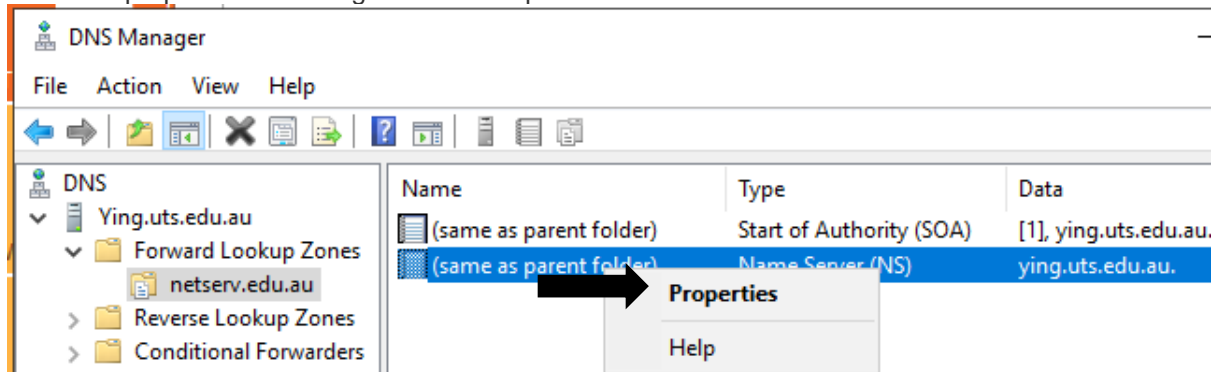


Select only 10.0.2.2 interface to listen for DNS queries.

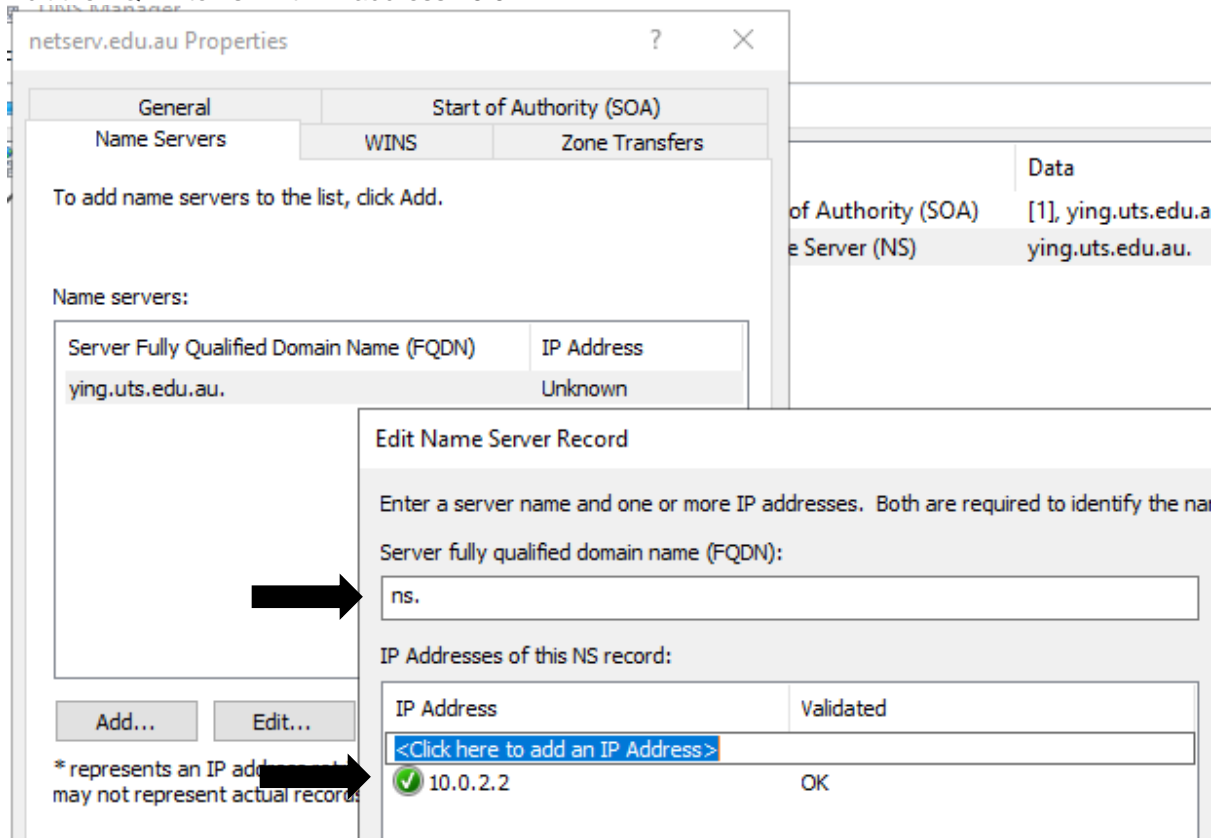


Step 5 Create the forward lookup zone file.

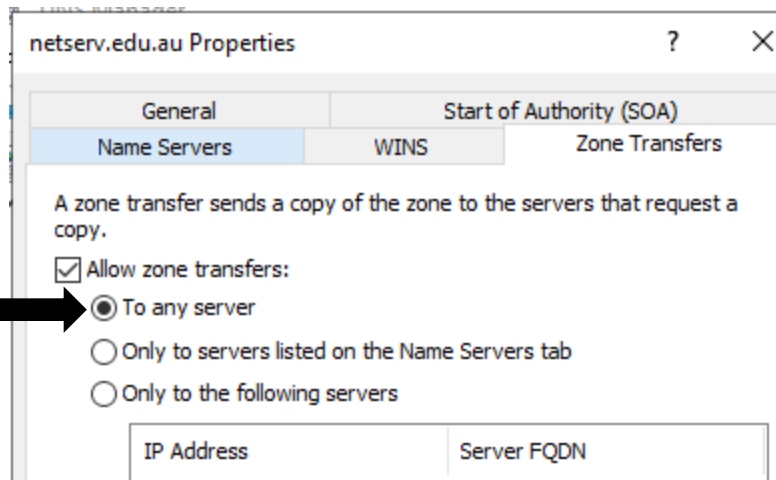
Click the forward lookup zones, then “netserv.edu.au”, we can see the zone file information, change the ns record properties according to our lab requirement.



Edit the FQDN to ns. With IP address 10.0.2.2



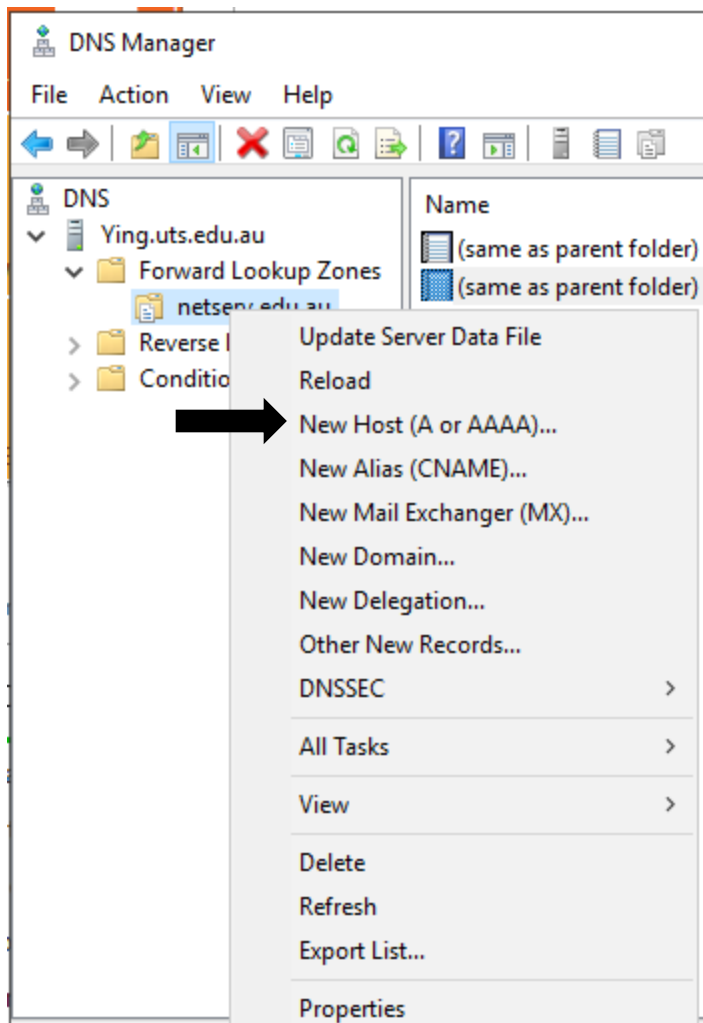
Zone transfer to any server



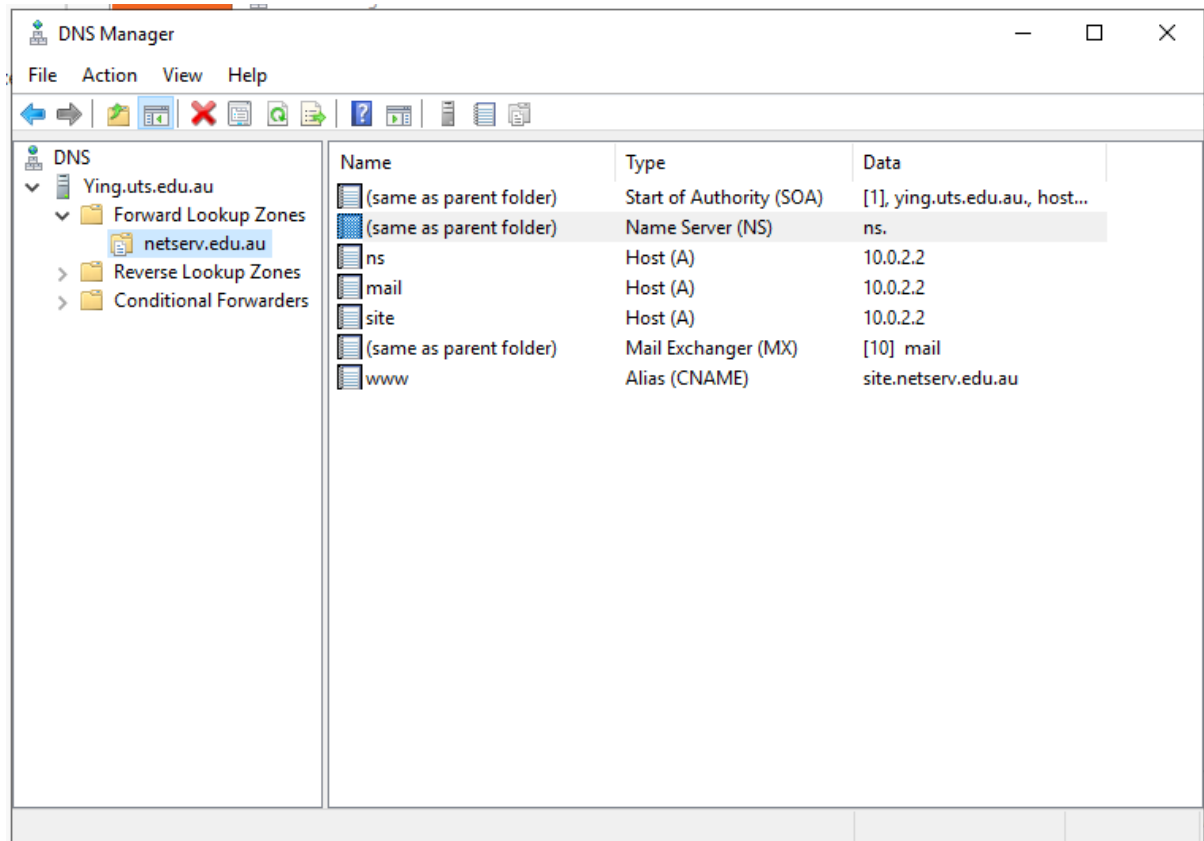
Step6 Add more entries to the Zone file

Right click the Zone name, then add the zone record according to the lab manual.

- New host (A): ns with IP address 10.0.2.2
- New host (A): mail with IP address 10.0.2.2
- New Mail Exchanger (MX): leave host blank and the FQDN should be mail
- New host (A): site with IP address 10.0.2.2
- New Alias (CNAME): www and FQDN site



Now the zone record look like the following



Right click the DNS domain, choose “**New delegation**” to Create a delegation for the it.netserv.edu.au domain.

Note about Delegation: For a DNS server to answer queries about any name, it must have a direct or indirect path to every zone in the namespace. These paths are created by means of delegation. A delegation is a record in a parent zone that lists a name server that is authoritative for the zone in the next level of the hierarchy. Delegations make it possible for servers in one zone to refer clients to servers in other zones.

New Delegation Wizard

Delegated Domain Name
Authority for the DNS domain you supply will be delegated to a different zone.

Specify the name of the DNS domain you want to delegate.

Delegated domain:

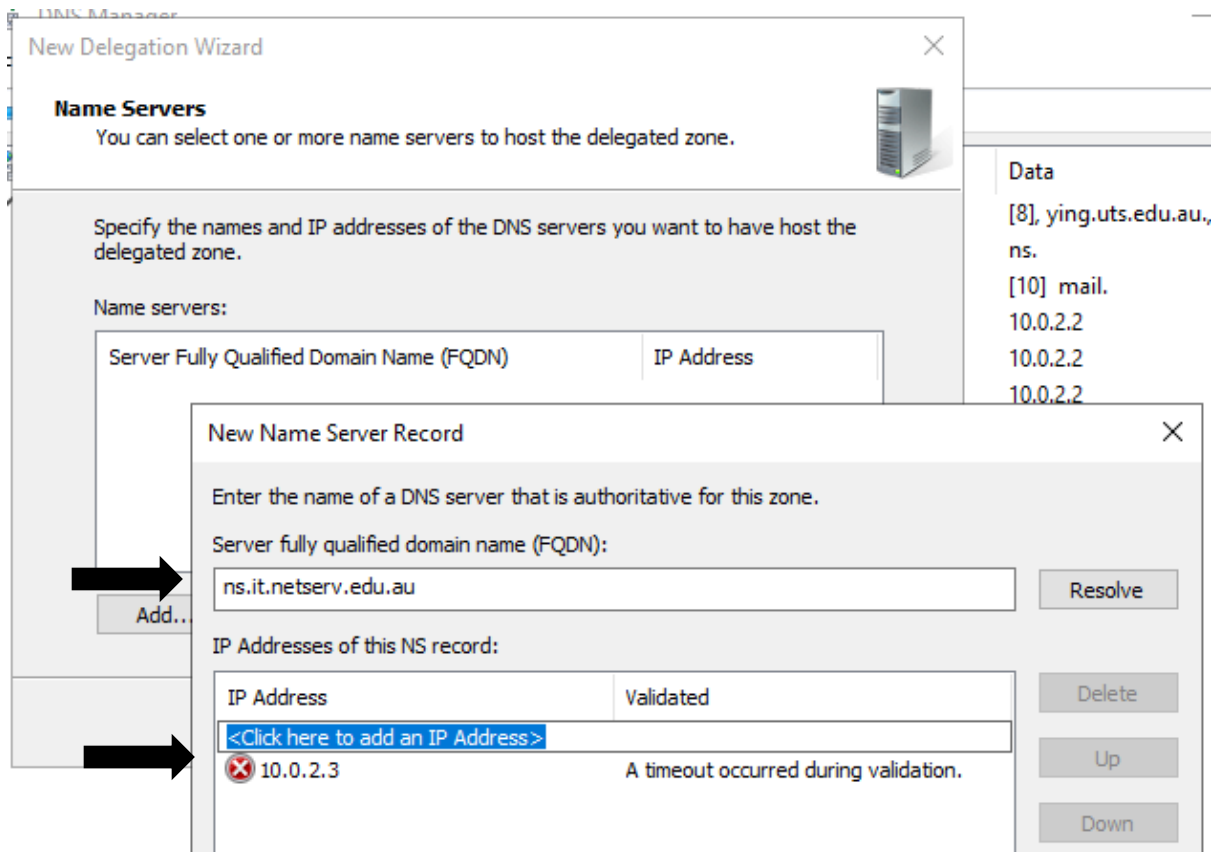
it

Fully qualified domain name (FQDN):

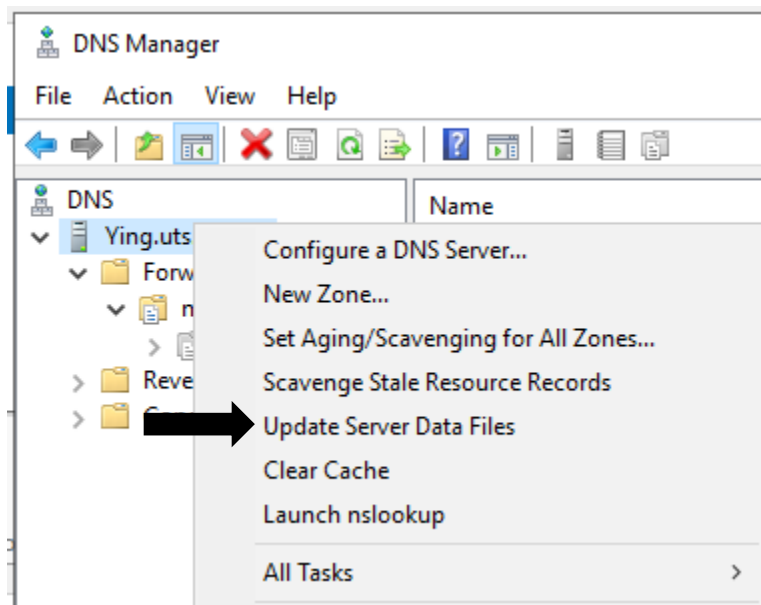
it.netserv.edu.au

< Back Next > Cancel

Click the "Add" to enter the name server record. We will use Linux server. The Linux machine is not configured yet, so you will see the validation timed out.



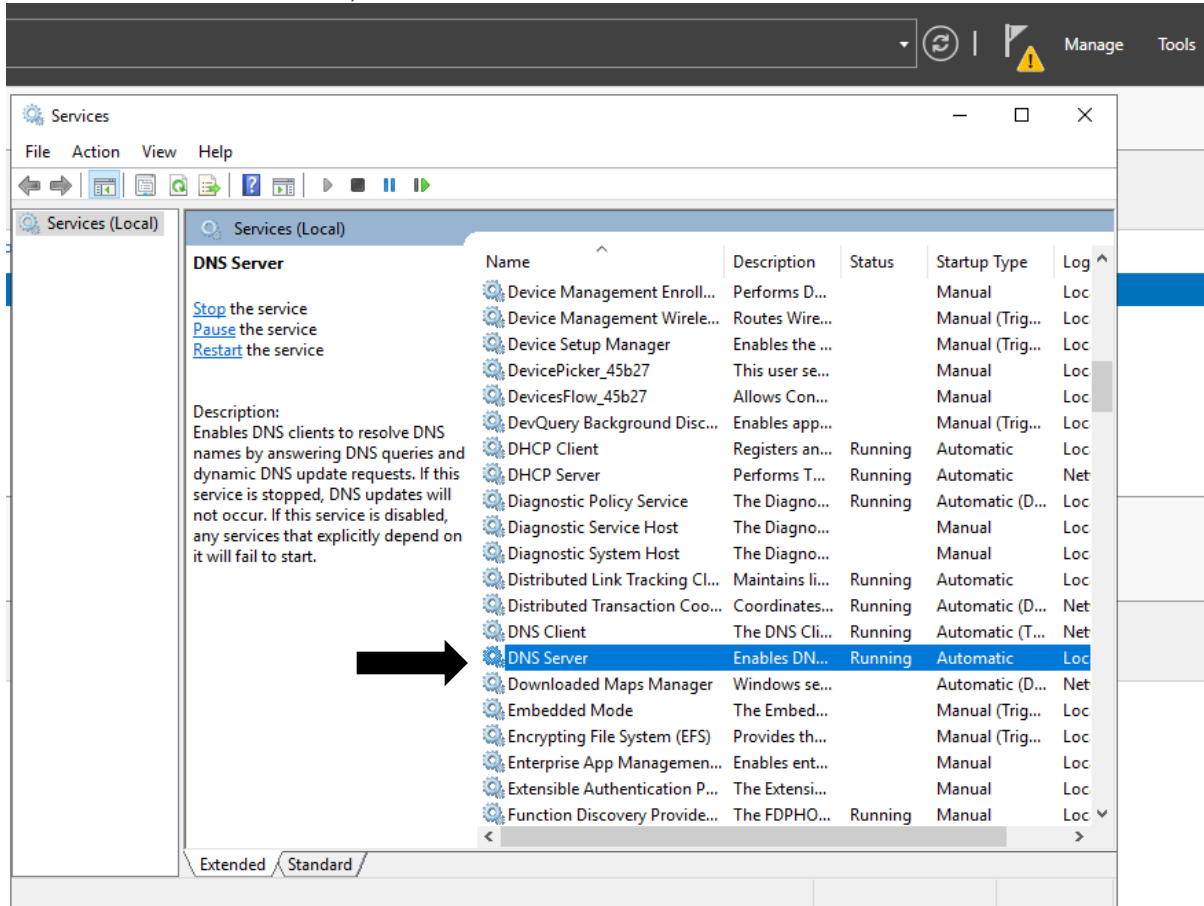
Now right click the server to “update Server Data File”.



Now restart the DNS server:

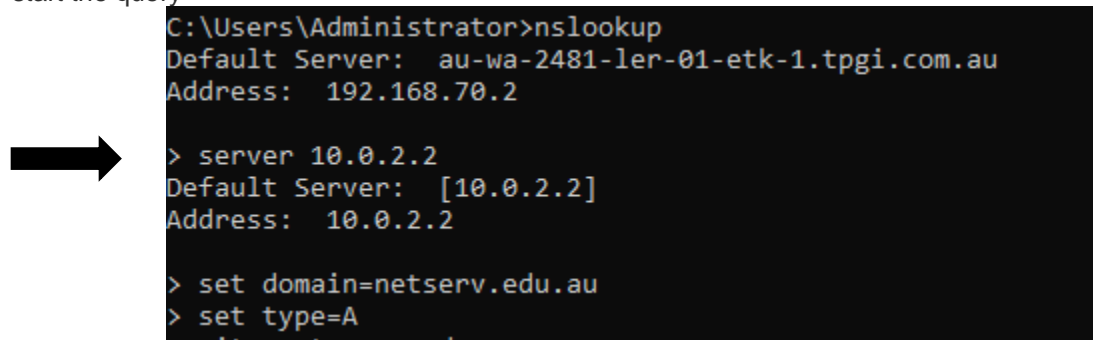


From “Tools” select “Service”, find the DNS Server to restart it.



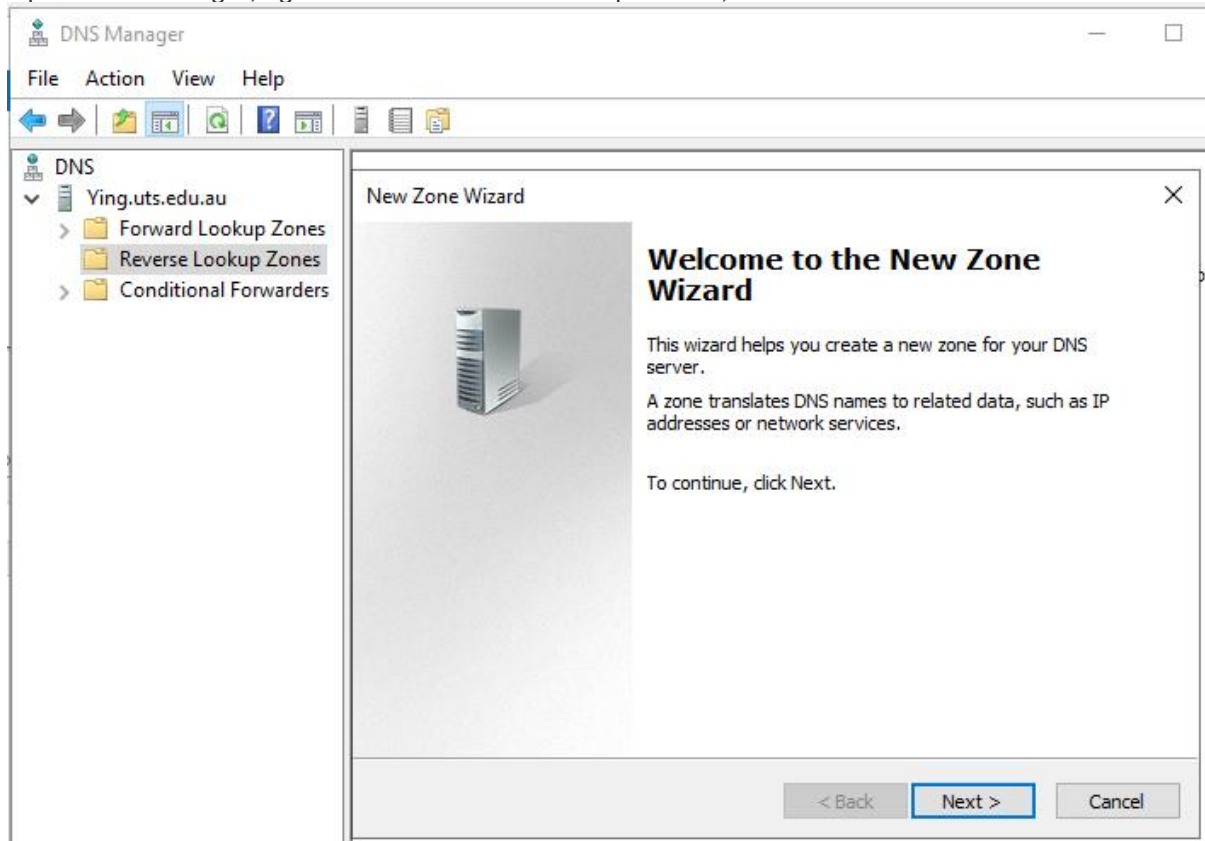
Step 7. Now we can test the DNS configuration.

Use nslookup to set the DNS server as 10.0.2.2, domain=netserv.edu.au, query type to A record, then start the query

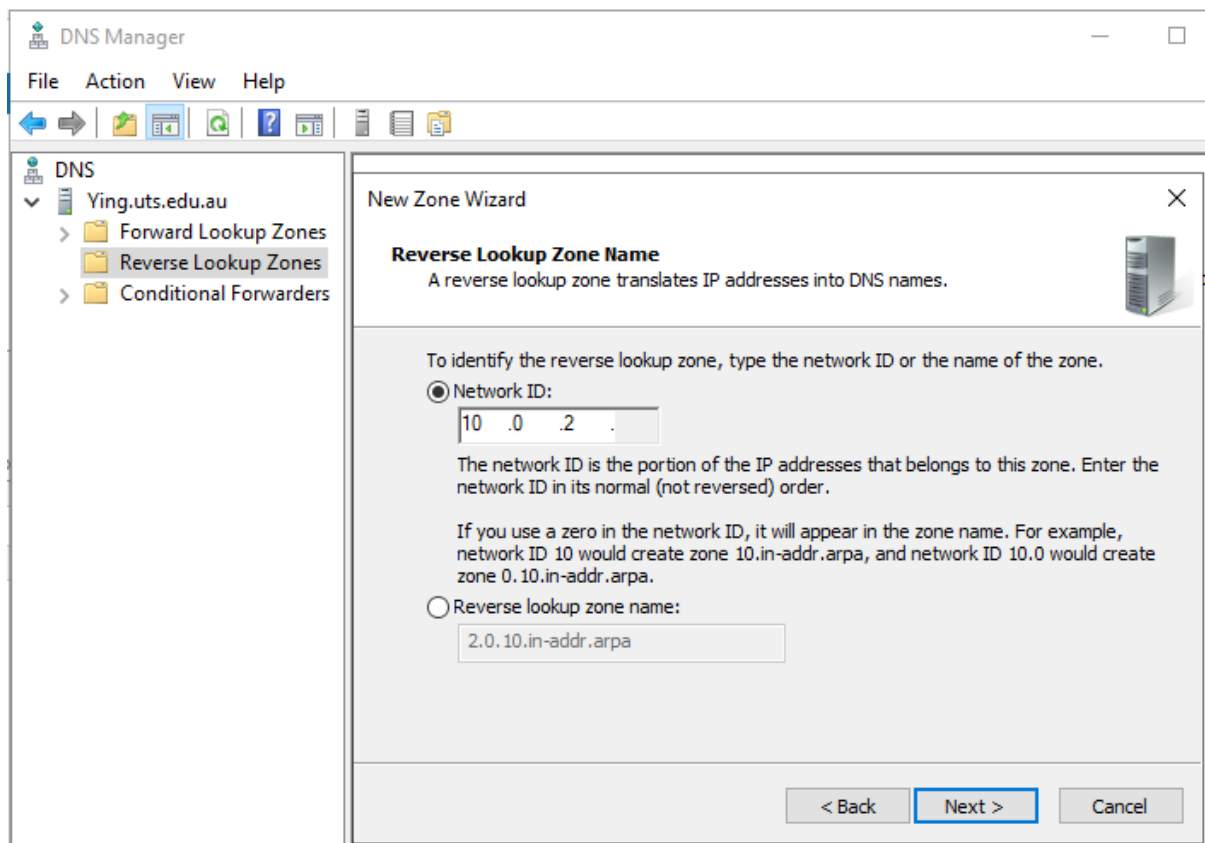


Step 8 add DNS Reverse Lookup Zone

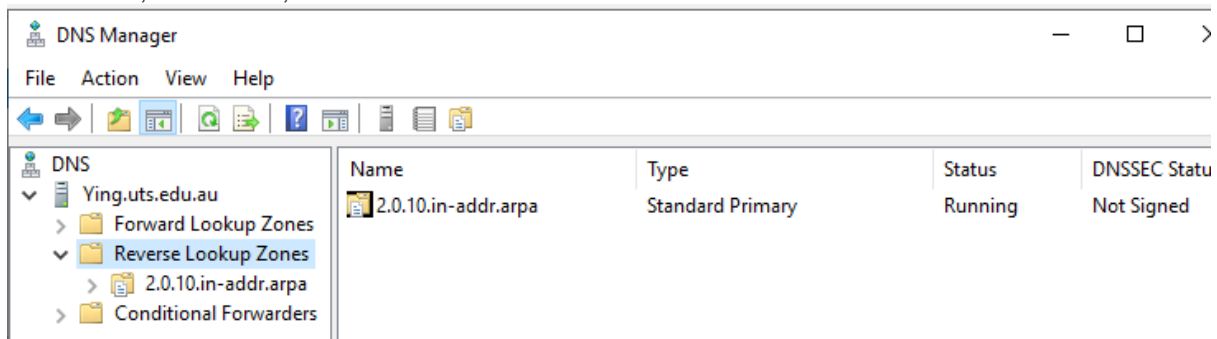
Open zone manager, right click the “Reverse Lookup Zones”, select a new zone.



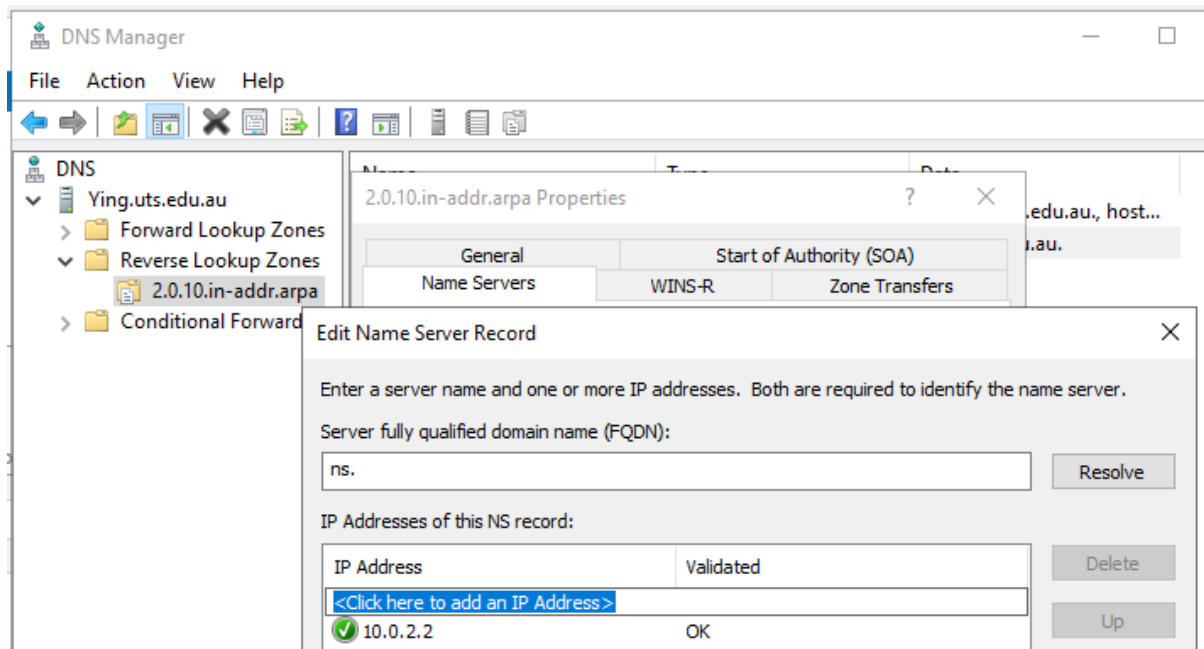
Then follow the configuration wizard, select “Primary zone”, “IPv4 Reverse lookup Zone”, enter the Network ID.



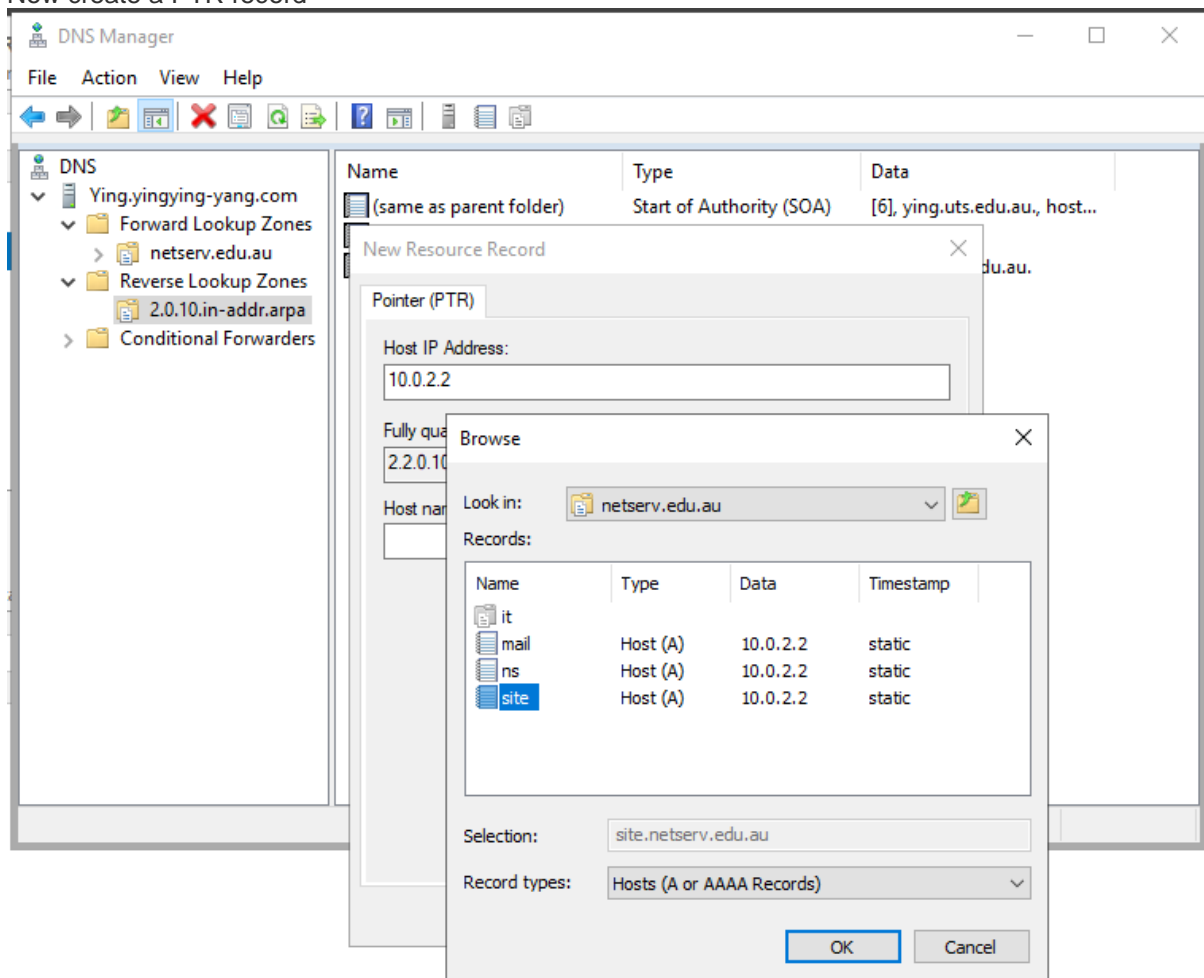
Click “next”, then “next”, “finish”



Select the reverse zone, edit the Name Server priority



Now create a PTR record



Set Up Private DNS Servers with BIND on CentOS 8

Prerequisites

- Access to the root user account (or access to an admin account with root privileges)
- Static IP address (we are going to use 10.0.2.3)

Step 1: Install BIND DNS Server

By default, the bind package is available in the CentOS 8 standard repository. You can install it by running the following command:

```
dnf install bind
```

Step 2: Configure BIND DNS Server

By default, the BIND server is listening on localhost only. So you will need to configure it to listen on all network interfaces. You can configure it by editing the file `/etc/named.conf`:

The following sections may be used in `/etc/named.conf`:

options — Assigns values to many assorted options, including the use of forwarders, the location of the named working directory, the names of the various files, and much more.

We are going to modify the following lines in this section to allow any host and add a forwarder:

```
options {  
    listen-on port 53 { any; };  
    listen-on-v6 port 53 { ::1; };  
    directory      "/var/named";  
    .....  
    allow-query    { any; };  
    forwarders {  
        10.0.2.2;  
    };
```

Note: for our forwarder to work we need to disable our Linux bind server from checking DNSSEC data.(DNS security extensions). In the `/etc/named.conf` file, look for the following lines, and update “yes” to “no” as shown below:


```
dnssec-enable no;
```

```
dnssec-validation no;
```

zone "<zone-name>" — Specifies particular zones for which this nameserver is authoritative. The zone statement is primarily used to specify the file containing the zone's configuration and pass certain options about that zone to named that override other global option statements used in /etc/named.conf.

We are going to add the following zone information in the /etc/named.conf file. A Forward Zone is used to resolve the hostname to IP address while a Reverse Zone is used to resolve the IP address to hostname. Generally, all normal DNS queries are forward lookup queries

```
//Forward Zone
zone "it.netserv.edu.au" IN {
    type master;
    file "it.netserv.edu.au.zone";
};

//Reverse Zone
zone "2.0.10.in-addr.arpa" {
    type master;
    file "2.0.10.in-addr.arpa.zone";
};
```

Save and close the file when you are finished.

Step 3: Create Forward and Reverse Zone Files

Zone files, which contain information about a particular namespace, are stored in the named working directory. By default, this is /var/named. Each zone file is named according to the file option data in the zone statement, usually in a way that relates to the domain in question and identifies the file as containing zone data, such as it.netserv.edu.au.zone.

Each zone file may contain directives and resource records. **Directives** tell the nameserver to do a certain thing or apply a special setting to the zone. **Resource records** define the parameters of the zone, assigning an identity within the zone's namespace to particular systems. Directives are optional, but resource records are required to provide nameservice to that zone. All directives and resource records should go on their own lines.

Let's create a forward and reverse zone files defined in the previous step. By default, all zone lookup files are located inside /var/named directory.

First, create a forward zone file with the following command:

```
vim /var/named/it.netserv.edu.au.zone
```

Add the following lines:

```
$TTL 1D
@      IN SOA  ns.it.netserv.edu.au. root.it.netserv.edu.au. (
                                0      ; serial
                                1D     ; refresh
                                1H     ; retry
                                1W     ; expire
                                3H )   ; minimum
```

```
      IN      NS      ns.it.netserv.edu.au.
      IN      MX      10      mail
      IN      A       10.0.2.3
```

```
site  IN      A       10.0.2.3
mail  IN      A       10.0.2.3
ns     IN      A       10.0.2.3
www    IN      CNAME   site
ftp    IN      CNAME   site
```

Create a reverse forward zone file with the following command:

```
vim /var/named/2.0.10.in-addr.arpa.zone
```

Add the following lines:

```
$TTL 1D
@      IN SOA  ns.it.netserv.edu.au. root.it.netserv.edu.au. (
                                0      ; serial
                                1D     ; refresh
                                1H     ; retry
                                1W     ; expire
                                3H )   ; minimum

      IN      NS      ns.it.netserv.edu.au.
2      IN      PTR     site.netserv.edu.au.
3      IN      PTR     site.it.netserv.edu.au.
```

Zone file resource records contain columns of data, separated by whitespace, that define the record. All zone file resource records are assigned a particular type, which designates the record's purpose. The following types of resource records are the most commonly used:

SOA — Start Of Authority record, proclaiming important authoritative information about the namespace to the nameserver.

Located after the directives, an SOA record is the first resource record in a zone file.

```
@      IN      SOA      <primary-name-server>      <hostmaster-email> (
                                <serial-number>
                                <time-to-refresh>
                                <time-to-retry>
```

```
<time-to-expire>
```

```
<minimum-TTL> )
```

Note: The serial number is incremented every time you change the zone file so that named will know that it should reload this zone.

A — Address record, which specifies an IP address to assign to a name.

```
<host>      IN      A      <IP-address>
```

If the <host> value is omitted, then an A record points to a default IP address for the top of the namespace. This system will be the target of all non-FQDN requests.

CNAME — Canonical name record, which tells the nameserver that one name is also known as another.

```
<alias-name>      IN      CNAME      <real-name>
```

MX — Mail eXchange record, which tells where mail sent to a particular namespace controlled by this zone should go.

```
      IN      MX      <preference-value>  <email-server-name>
```

The <preference-value> allows you to numerically rank the email servers you would prefer to receive email for this namespace.

NS — NameServer record, which announces the authoritative nameservers for a particular zone.

```
      IN      NS      <nameserver-name>
```

The <nameserver-name> should be a FQDN.

PTR — PoinTeR record, designed to point to another part of the namespace.

PTR records are primarily used for reverse name resolution, as they point IP addresses back to a particular name.

```
<last-IP-digit>      IN      PTR      <FQDN-of-system>
```

Note: remember to change the zone file ownership to “named” after modifying the zone file.

```
chgrp named it.netserv.edu.au.zone
chgrp named 2.0.10.in-addr.arpa.zone
```

Save and close the file when you are finished.

Step 4: Verify DNS Configuration

After configuring all zone files, you will need to verify the configuration files.

First, validate the main configuration file with the following command:

```
named-checkconf /etc/named.conf
```

If everything is fine, you don't see any error.

Next, verify the forward zone file with the following command:

```
named-checkzone it.netserv.ecud.au /var/named/it.netserv.edu.au.zone
```

You should get the following output:

```
zone it.netserv.ecud.au/IN: loaded serial 0
OK
```

Next, verify the reverse zone file with the following command:

```
named-checkzone 2.0.10.in-addr.arpa.zone /var/named/2.0.10.in-addr.arpa.zone
```

You should get the following output:

```
zone 1.168.192.in-addr.arpa/IN: loaded serial 3
OK
```

Finally, start the BIND service and enable it to start at system reboot:

```
systemctl start named
systemctl enable named
```

Step 5: Configure Firewall

Next, you will need to create a firewall rule for port 53 to allow DNS queries from client machines. You can create it with the following command:

```
firewall-cmd --permanent --add-port=53/udp
```

Next, reload the firewall service to apply the changes:

```
firewall-cmd --reload
```

Step 5: Verify DNS Server

At this point, the BIND DNS server is installed and configured. It's time to check whether it is working or not.

```
dig @localhost site.it.netserv.edu.au
dig @localhost it.netserv.edu.au soa
```

```
dig @localhost -x 10.0.2.2
```

Congratulations! you have successfully set up a private DNS server with BIND on CentOS 8.

Week6 Lab 6b Configure DNS Client

<p>/etc/hosts define local DNS cache, is used for mapping hostnames with IP addresses.</p> <p>IP Address Host Name Alias</p> <p>The Alias field is used for mapping short names or labels to host</p>	<pre># cat /etc/hosts 127.0.0.1 localhost.localdomain localhost 192.0.2.101 host01.example.com. host01</pre>
<p>/etc/resolv.conf is the configuration file for DNS resolvers.</p> <p>The “search” keyword specifies a list of domain names to be searched, there must be only one “search” line in the file.</p> <p>The “nameserver” specifies the IP address of a name server that this client should attempt to connect to. There can be many “nameserver” lines in the file.</p> <p>The “domain” specifies the domain name to be used locally.</p> <p>/etc/sysconfig/network-scripts/ifcfg-* will overwrite the contents of resolv.conf, so edit the interface configuration files and allow NetworkManager to populate resolv.conf file.</p>	<pre># /etc/resolv.conf domain sample.com search it.uts.edu.au iwork.uts.eud.au uts.edu.au # central nameserver Nameserver 138.25.9.1 nameserver 191.74.10.12</pre>

To tell the NetworkManager to ignore the DNS server provided by DHCP, use “nmcli con modify ens33 ipv4.ignore-auto-dns true”. Then you will find “PEERDNS=no” in the interface configuration file.	
The order of search domains and nameservers in the resolv.conf is determined by NetworkManager. It has some default priority rules, and if the interfaces have the same priority they are just added in order. The default priority for normal connections is 100. The lower value has higher priority.	nmcli conn modify ens37 ipv4.dns-priority 5

File Permission

File Permission Types

User Owner	Group Owner	Other
r w x	r w x	r w x

Read

The first character of each group represents the *read* permission. There is an `r` character if the group has the read permission, or a `-` character if the group does not.

- On a *file*, this allows processes to read the contents of the file, meaning the contents can be viewed and copied.
- On a *directory*, file names in the directory can be listed, but other details are not available.

Write

The second character of each group represents the *write* permission. There is a `w` character if the group has the write permission, or a `-` character if the group does not.

- A *file* can be written to by the process, so changes to a file can be saved. Note that `w` permission really requires `r` permission on the file to work correctly.
- On a *directory*, files can be added to or removed from the directory. Note that `w` permission requires `x` permission on the directory to work correctly.

Execute

<p>The third character of each group represents the <i>execute</i> permission. There is an <code>x</code> character if the group has the execute permission, or a <code>-</code> character if the group does not.</p> <ul style="list-style-type: none"> • A <i>file</i> can be executed or run as a process. • On a <i>directory</i>, the user can use the <code>cd</code> command to "get into" the directory and use the directory in a pathname to access files and, potentially, subdirectories under this directory. 	
<p>To change the group owner of an existing file the <code>chgrp</code> command can be used.</p> <p>To change the group ownership of all of the files of a directory structure, use the recursive <code>-R</code> option to the <code>chgrp</code> command.</p>	<pre>\$touch sample \$ls -l sample -rw-rw-r-- 1 sysadmin sysadmin 0 Oct 23 22:12 sample \$ chgrp research sample \$ ls -l sample -rw-rw-r--. 1 sysadmin research 0 Oct 23 22:12 sample \$ chgrp -R development test_dir</pre>
<p>The <code>chown</code> command allows the root user to change the user ownership of files and directories. A regular user cannot use this command to change the user owner of a file, even to give the ownership of one of their own files to another user. However, the <code>chown</code> command also permits changing group ownership, which can be accomplished by either root or the owner of the file.</p>	<pre># chown jane /tmp/filetest1 # ls -l /tmp/filetest1 -rw-rw-r-- 1 jane sysadmin 0 Dec 19 18:44 /tmp/filetest1 # chown jane:users /tmp/filetest2 # ls -l /tmp/filetest2 -rw-r--r-- 1 jane users 0 Dec 19 18:53 /tmp/filetest2 jane@localhost:~\$ chown .users /tmp/filetest1 jane@localhost:~\$ ls -l /tmp/filetest1 -rw-rw-r-- 1 jane users 0 Dec 19 18:44 /tmp/filetest1</pre>
<p>The <code>chmod</code> (change mode) command is used to change permissions on files and directories. Only the root user or the user who owns the file is able to change the permissions of a file. There are two techniques that can be used with this command: <i>symbolic</i> and</p>	<pre># touch abc.txt # ls -l abc.txt -rw-r--r-- 1 root root 0 Dec 19 18:58 abc.txt # chmod g+w abc.txt</pre>

<p><i>numeric.</i> Both techniques use the following basic syntax:</p>	<pre># ls -l abc.txt -rw-rw-r-- 1 root root 0 Dec 19 18:58 abc.txt # chmod ug+x,o-r abc.txt # ls -l abc.txt -rwxrwx--- 1 root root 0 Dec 19 18:58 abc.txt # chmod u=rx abc.txt # ls -l abc.txt -r-xrwx--- 1 root root 0 Dec 19 18:58 abc.txt # chmod 754 abc.txt # ls -l abc.txt -rwxr-xr-- 1 root root 0 Dec 19 18:58 abc.txt</pre>																			
<p>Octal method</p> <table><tr><th>Octal Value</th><th>Permission</th></tr><tr><td>4</td><td>Read</td></tr><tr><td>2</td><td>Write</td></tr><tr><td>1</td><td>Execute</td></tr></table>	Octal Value	Permission	4	Read	2	Write	1	Execute												
Octal Value	Permission																			
4	Read																			
2	Write																			
1	Execute																			
<p>Special permission</p> <p>Typically, special permissions are only set by the administrator (the root user) and they perform very specialized functions. They can be set using the <code>chmod</code> command, using either the symbolic or octal method.</p>	<table><tr><th>Permision</th><th>Symbo l</th><th>Symb olic</th><th>Octal Value</th><th>Purpose</th></tr><tr><td>setuid on a file</td><td>s</td><td>u+s</td><td>4000</td><td>Causes an executable file to execute under user owner identity, instead of the user running the command.</td></tr><tr><td>setgid on a file</td><td>s</td><td>g+s</td><td>2000</td><td>Causes an executable file to execute under group owner identity, instead of the user running the command.</td></tr></table>					Permision	Symbo l	Symb olic	Octal Value	Purpose	setuid on a file	s	u+s	4000	Causes an executable file to execute under user owner identity, instead of the user running the command.	setgid on a file	s	g+s	2000	Causes an executable file to execute under group owner identity, instead of the user running the command.
Permision	Symbo l	Symb olic	Octal Value	Purpose																
setuid on a file	s	u+s	4000	Causes an executable file to execute under user owner identity, instead of the user running the command.																
setgid on a file	s	g+s	2000	Causes an executable file to execute under group owner identity, instead of the user running the command.																

	setgid on a directory	s	g+s	2000	Causes new files and directories that are created inside to be owned by the group that owns the directory. In addition, any directories created within a directory with the setgid permission set are not only owned by the group that owns the setgid directory, but the new directory automatically has setgid set on it as well.
	sticky on a directory	t	o+t	1000	Causes files inside a directory to be able to be removed only by the user owner, or the root user.
	\$ ls -l /usr/bin/passwd				
	-rwsr-xr-x 1 root root 59640 Jan 25 2018 /usr/bin/passwd				
	To add the setuid permission symbolically, run:				
	chmod u+s file				
	To add the setuid permission numerically, add 4000 to the file's existing permissions (assume the file originally had 775 for its permission in the following example):				
	chmod 4775 file				
	To remove the setuid permission symbolically, run:				
	chmod u-s file				
	To remove the setuid permission numerically, subtract 4000 from the file's existing permissions:				
	chmod 0775 file				

<p>The <code>umask</code> command is a feature that is used to determine default permissions that are set when a file or directory is created. Default permissions are determined when the <i>umask value</i> is subtracted from the maximum allowable default permissions. The maximum default permissions are different for files and directories.</p> <p>File rw-rw-rw-</p> <p>Directories rw-rw-rw-</p> <p>The new umask is only applied to file and directories created during that session. When a new shell is started, the default umask will again be in effect.</p> <p>Permanently changing a user's umask requires modifying the <code>.bashrc</code> file located in that user's home directory.</p>	<p>\$ umask</p> <p>0002</p> <ul style="list-style-type: none"> • The first 0 indicates that the umask is given as an octal number. • The second 0 indicates which permissions to subtract from the default user owner's permissions. • The third 0 indicates which permissions to subtract from the default group owner's permissions. • The last number 2 indicates which permissions to subtract from the default other's permissions. <p>\$ umask 027</p> <p>\$ touch sample</p> <p>\$ ls -l sample</p> <p>-rw-r-----. 1 sysadmin sysadmin 0 Oct 28 20:14 sample</p> <p>\$ mkdir test-dir</p> <p>\$ ls -ld test-dir</p> <p>drwxr-x---- 1 sysadmin sysadmin 4096 Oct 28 20:25 test-dir</p> <p>\$ vi ~/.bashrc</p> <p># .bashrc</p> <p># Source global definitions</p> <p>if [-f /etc/bashrc]; then</p> <p> . /etc/bashrc</p> <p>fi</p> <p># User specific environment</p> <p>PATH="\$HOME/.local/bin:\$HOME/bin:\$PATH"</p> <p>export PATH</p>
---	--

	<pre># Uncomment the following line if you don't like systemctl's auto-paging feature: # export SYSTEMD_PAGER= # User specific aliases and functions umask 0002</pre>
--	--

Special Permission Scenario

First, consider the following user accounts:

- The user `joe` is a member of the `staff` group.
- The user `maya` is a member of the `payroll` group.
- The user `steve` is a member of the `acct` group.

In this scenario, these three users need to work on a joint project. They approach the administrator to ask for a shared directory in which they can work together, but that no one else can access their files. The administrator does the following:

- Creates a new group called `team`.
- Adds `joe`, `maya`, and `steve` to the `team` group.
- Makes a new directory called `shared` under the `/srv` directory.

```
root@localhost:~# mkdir /srv/shared
```
- Makes the `team` group the group owner of the `shared` directory.

```
root@localhost:~# chgrp team /srv/shared
```
- Make the `shared` directory writable for the group by giving it the `rw-rwx---` permissions:

```
root@localhost:~# chmod 770 /srv/shared
root@localhost:~# ls -ld /srv/shared
drwxrwx--- 2 root team 4096 Mar 28 02:14 /srv/shared
```

This solution is almost perfect: the users `joe`, `maya` and `steve` can now access the `/srv/shared` directory and add new files. However, there is a potential problem, because the user `joe` is a member of the `staff` group, when `joe` makes a new file in the `/srv/shared` directory, the new file is owned by the `staff` group:

```
-rw-rw----. 2 joe staff 8987 Jan 10 09:08 staffdocument.txt
```

The problem with this is that neither `maya` or `steve` are members of the `staff` group. As a result, their permissions are `---`, which means they can't access the contents of this file.

The user `joe` could have used the `newgrp` command to switch to the `team` group before creating the file. Or, after creating the file, the user `joe` could have used the `chgrp` command to change the group

ownership to the `common` group. However, users won't always remember to run these commands; some may not even know that these commands exist.

Instead, the administrator can set up a directory with `setgid` permission. If a directory is `setgid`, then all new files created or copied into the directory will *automatically* be owned by the group that owns the directory. This means users don't have to use the `newgrp` or `chgrp` commands because the group ownership will be managed automatically.

So, a better solution to this scenario would be to set the `setgid` permission on the `/srv/shared` directory using the following command:

```
root@localhost:~# chmod 2775 /srv/shared
```

Listing the details of the directory after running the previous command would result in the following output:

```
drwxrwsr-x. 2 root team 4096 Jan 10 09:08 /srv/shared
```

After completing these steps, the `joe`, `maya`, and `steve` users would now be able to easily create files in the `/srv/shared` directory that would automatically be owned by the `team` group.

File Permission

File Permission Types

User Owner	Group Owner	Other
r w x	r w x	r w x

Read

The first character of each group represents the *read* permission. There is an `r` character if the group has the read permission, or a `-` character if the group does not.

- On a *file*, this allows processes to read the contents of the file, meaning the contents can be viewed and copied.
- On a *directory*, file names in the directory can be listed, but other details are not available.

Write

The second character of each group represents the *write* permission. There is a `w` character if the group has the write permission, or a `-` character if the group does not.

- A *file* can be written to by the process, so changes to a file can be saved. Note that `w` permission really requires `r` permission on the file to work correctly.
- On a *directory*, files can be added to or removed from the directory. Note that `w` permission requires `x` permission on the directory to work correctly.

Execute

<p>The third character of each group represents the <i>execute</i> permission. There is an <code>x</code> character if the group has the execute permission, or a <code>-</code> character if the group does not.</p> <ul style="list-style-type: none"> • A <i>file</i> can be executed or run as a process. • On a <i>directory</i>, the user can use the <code>cd</code> command to "get into" the directory and use the directory in a pathname to access files and, potentially, subdirectories under this directory. 	
<p>To change the group owner of an existing file the <code>chgrp</code> command can be used.</p> <p>To change the group ownership of all of the files of a directory structure, use the recursive <code>-R</code> option to the <code>chgrp</code> command.</p>	<pre>\$touch sample \$ls -l sample -rw-rw-r-- 1 sysadmin sysadmin 0 Oct 23 22:12 sample \$ chgrp research sample \$ ls -l sample -rw-rw-r--. 1 sysadmin research 0 Oct 23 22:12 sample \$ chgrp -R development test_dir</pre>
<p>The <code>chown</code> command allows the root user to change the user ownership of files and directories. A regular user cannot use this command to change the user owner of a file, even to give the ownership of one of their own files to another user. However, the <code>chown</code> command also permits changing group ownership, which can be accomplished by either root or the owner of the file.</p>	<pre># chown jane /tmp/filetest1 # ls -l /tmp/filetest1 -rw-rw-r-- 1 jane sysadmin 0 Dec 19 18:44 /tmp/filetest1 # chown jane:users /tmp/filetest2 # ls -l /tmp/filetest2 -rw-r--r-- 1 jane users 0 Dec 19 18:53 /tmp/filetest2 jane@localhost:~\$ chown .users /tmp/filetest1 jane@localhost:~\$ ls -l /tmp/filetest1 -rw-rw-r-- 1 jane users 0 Dec 19 18:44 /tmp/filetest1</pre>
<p>The <code>chmod</code> (change mode) command is used to change permissions on files and directories. Only the root user or the user who owns the file is able to change the permissions of a file. There are two techniques that can be used with this command: <i>symbolic</i> and</p>	<pre># touch abc.txt # ls -l abc.txt -rw-r--r-- 1 root root 0 Dec 19 18:58 abc.txt # chmod g+w abc.txt</pre>

<p><i>numeric.</i> Both techniques use the following basic syntax:</p>	<pre># ls -l abc.txt -rw-rw-r-- 1 root root 0 Dec 19 18:58 abc.txt # chmod ug+x,o-r abc.txt # ls -l abc.txt -rwxrwx--- 1 root root 0 Dec 19 18:58 abc.txt # chmod u=rx abc.txt # ls -l abc.txt -r-xrwx--- 1 root root 0 Dec 19 18:58 abc.txt # chmod 754 abc.txt # ls -l abc.txt -rwxr-xr-- 1 root root 0 Dec 19 18:58 abc.txt</pre>																			
<p>Octal method</p> <table><tr><th>Octal Value</th><th>Permission</th></tr><tr><td>4</td><td>Read</td></tr><tr><td>2</td><td>Write</td></tr><tr><td>1</td><td>Execute</td></tr></table>	Octal Value	Permission	4	Read	2	Write	1	Execute												
Octal Value	Permission																			
4	Read																			
2	Write																			
1	Execute																			
<p>Special permission</p> <p>Typically, special permissions are only set by the administrator (the root user) and they perform very specialized functions. They can be set using the <code>chmod</code> command, using either the symbolic or octal method.</p>	<table><tr><th>Permision</th><th>Symbo l</th><th>Symb olic</th><th>Octal Value</th><th>Purpose</th></tr><tr><td>setuid on a file</td><td>s</td><td>u+s</td><td>4000</td><td>Causes an executable file to execute under user owner identity, instead of the user running the command.</td></tr><tr><td>setgid on a file</td><td>s</td><td>g+s</td><td>2000</td><td>Causes an executable file to execute under group owner identity, instead of the user running the command.</td></tr></table>					Permision	Symbo l	Symb olic	Octal Value	Purpose	setuid on a file	s	u+s	4000	Causes an executable file to execute under user owner identity, instead of the user running the command.	setgid on a file	s	g+s	2000	Causes an executable file to execute under group owner identity, instead of the user running the command.
Permision	Symbo l	Symb olic	Octal Value	Purpose																
setuid on a file	s	u+s	4000	Causes an executable file to execute under user owner identity, instead of the user running the command.																
setgid on a file	s	g+s	2000	Causes an executable file to execute under group owner identity, instead of the user running the command.																

	setgid on a directory	s	g+s	2000	Causes new files and directories that are created inside to be owned by the group that owns the directory. In addition, any directories created within a directory with the setgid permission set are not only owned by the group that owns the setgid directory, but the new directory automatically has setgid set on it as well.
	sticky on a directory	t	o+t	1000	Causes files inside a directory to be able to be removed only by the user owner, or the root user.
	\$ ls -l /usr/bin/passwd				
	-rwsr-xr-x 1 root root 59640 Jan 25 2018 /usr/bin/passwd				
	To add the setuid permission symbolically, run:				
	chmod u+s file				
	To add the setuid permission numerically, add 4000 to the file's existing permissions (assume the file originally had 775 for its permission in the following example):				
	chmod 4775 file				
	To remove the setuid permission symbolically, run:				
	chmod u-s file				
	To remove the setuid permission numerically, subtract 4000 from the file's existing permissions:				
	chmod 0775 file				

<p>The <code>umask</code> command is a feature that is used to determine default permissions that are set when a file or directory is created. Default permissions are determined when the <i>umask value</i> is subtracted from the maximum allowable default permissions. The maximum default permissions are different for files and directories.</p> <p>File rw-rw-rw-</p> <p>Directories rw-rw-rw-</p> <p>The new umask is only applied to file and directories created during that session. When a new shell is started, the default umask will again be in effect.</p> <p>Permanently changing a user's umask requires modifying the <code>.bashrc</code> file located in that user's home directory.</p>	<p>\$ umask</p> <p>0002</p> <ul style="list-style-type: none"> • The first 0 indicates that the umask is given as an octal number. • The second 0 indicates which permissions to subtract from the default user owner's permissions. • The third 0 indicates which permissions to subtract from the default group owner's permissions. • The last number 2 indicates which permissions to subtract from the default other's permissions. <p>\$ umask 027</p> <p>\$ touch sample</p> <p>\$ ls -l sample</p> <p>-rw-r-----. 1 sysadmin sysadmin 0 Oct 28 20:14 sample</p> <p>\$ mkdir test-dir</p> <p>\$ ls -ld test-dir</p> <p>drwxr-x---. 1 sysadmin sysadmin 4096 Oct 28 20:25 test-dir</p> <p>\$ vi ~/.bashrc</p> <p># .bashrc</p> <p># Source global definitions</p> <p>if [-f /etc/bashrc]; then</p> <p> . /etc/bashrc</p> <p>fi</p> <p># User specific environment</p> <p>PATH="\$HOME/.local/bin:\$HOME/bin:\$PATH"</p> <p>export PATH</p>
---	--

	<pre># Uncomment the following line if you don't like systemctl's auto-paging feature: # export SYSTEMD_PAGER= # User specific aliases and functions umask 0002</pre>
--	--

Special Permission Scenario

First, consider the following user accounts:

- The user `joe` is a member of the `staff` group.
- The user `maya` is a member of the `payroll` group.
- The user `steve` is a member of the `acct` group.

In this scenario, these three users need to work on a joint project. They approach the administrator to ask for a shared directory in which they can work together, but that no one else can access their files. The administrator does the following:

- Creates a new group called `team`.
- Adds `joe`, `maya`, and `steve` to the `team` group.
- Makes a new directory called `shared` under the `/srv` directory.

```
root@localhost:~# mkdir /srv/shared
```
- Makes the `team` group the group owner of the `shared` directory.

```
root@localhost:~# chgrp team /srv/shared
```
- Make the `shared` directory writable for the group by giving it the `rw-rwx---` permissions:

```
root@localhost:~# chmod 770 /srv/shared
root@localhost:~# ls -ld /srv/shared
drwxrwx--- 2 root team 4096 Mar 28 02:14 /srv/shared
```

This solution is almost perfect: the users `joe`, `maya` and `steve` can now access the `/srv/shared` directory and add new files. However, there is a potential problem, because the user `joe` is a member of the `staff` group, when `joe` makes a new file in the `/srv/shared` directory, the new file is owned by the `staff` group:

```
-rw-rw----. 2 joe staff 8987 Jan 10 09:08 staffdocument.txt
```

The problem with this is that neither `maya` or `steve` are members of the `staff` group. As a result, their permissions are `---`, which means they can't access the contents of this file.

The user `joe` could have used the `newgrp` command to switch to the `team` group before creating the file. Or, after creating the file, the user `joe` could have used the `chgrp` command to change the group

ownership to the common group. However, users won't always remember to run these commands; some may not even know that these commands exist.

Instead, the administrator can set up a directory with setgid permission. If a directory is setgid, then all new files created or copied into the directory will *automatically* be owned by the group that owns the directory. This means users don't have to use the `newgrp` or `chgrp` commands because the group ownership will be managed automatically.

So, a better solution to this scenario would be to set the setgid permission on the `/srv/shared` directory using the following command:

```
root@localhost:~# chmod 2775 /srv/shared
```

Listing the details of the directory after running the previous command would result in the following output:

```
drwxrwsr-x. 2 root team 4096 Jan 10 09:08 /srv/shared
```

After completing these steps, the joe, maya, and steve users would now be able to easily create files in the `/srv/shared` directory that would automatically be owned by the team group.

How to enable User and Group Disk Quota on ext4

Quota restricts the users to use only allowed **disk** and **inodes** on the particular file system.

Step:1 Add `usrquota` & `grpquota` option on `/home` in `/etc/fstab` file.

```
[root@linuxtechi ~]# vi /etc/fstab

#
# /etc/fstab
# Created by anaconda on Mon Aug 25 04:24:13 2014
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root / xfs defaults 1 1
UUID=804a7f13-d59b-4148-b9d9-9095a4b00c11 /boot xfs defaults 1 2
/dev/mapper/centos-swap swap swap defaults 0 0
/dev/mapper/centos-home /home ext4 defaults,usrquota,grpquota 1 2
```

Save & exit the file.

In this example user and group quota options are added on `/home`

Step:2 Remount /home file system via mount command

```
[root@linuxtech1 ~]# mount -o remount /home
```

Now recheck the /home file system whether Quota is enable or not.

```
[root@linuxtech1 ~]# mount | grep /home

/dev/mapper/centos-home on /home type ext4
(rw,relatime,seclabel,quota,usrquota,grpquota,data=ordered)
```

Step:3 Create Quota Database Files using quotacheck

```
[root@linuxtech1 home]# quotacheck -cugv /home
```

Whereas :

- c : create quota file and don't use the existing file
- v : verbose output
- u : user disk quota
- g : group disk quota

Above Command will create aquota.user & aquota.group files under /home

Turn on quota on /home using below command :

```
[root@linuxtech1 ~]# quotaon /home/
```

Step:4 Assign user & group disk quota via edquota commands

Syntax # edquota -u <User_Name>

edquota -g <Group_Name>

```
[root@linuxtech1 ~]# edquota -u jack
```

```
Disk quotas for group sys_admin (gid 1002):
Filesystem      blocks      soft      hard      inodes      soft      hard
/dev/mapper/centos-home    0      7000      8000         0         0         0
```

```
[root@linuxtechi ~]# edquota -g sys_admin
```

```
Disk quotas for group sys_admin (gid 1002):
Filesystem      blocks      soft      hard      inodes      soft      hard
/dev/mapper/centos-home    0      7000      8000         0         0         0
```

As shown above we have two kind of **Disk quota limits** :

soft : It will warn the users if the soft limit of disk quota reached (size is in KB), in above example for jack user soft limit is 5500 KB (approx 5.5MB)

hard : It will not allow the users to create new files once the hard limit is reached. (Size in KB), in above example hard limit for jack user is 6000 KB (approx 6 MB)

Note : We can also set the Quota on the basis of the inodes (i.e numbers of files that the user can create on particular file system)

Let's take an example , login as jack user and try to create a file of 8MB.

```
[root@linuxtechi ~]# su - jack

[jack@linuxtechi ~]$ dd if=/dev/zero of=bgfile bs=1M count=8

dm-2: warning, user block quota exceeded.

dm-2: write failed, user block limit reached.

dd: error writing 'bgfile': Disk quota exceeded

6+0 records in
```

```
5+0 records out
```

```
6144000 bytes (6.1 MB) copied, 0.00711317 s, 864 MB/s
```

As we see above soft & hard limit is exceeded for jack user. Now onwards user jack can't create new files.

Step:5 Display Quota report for Users in human readable

```
[root@linuxtech1 ~]# repquota -as
```

```
[root@linuxtech1 ~]# repquota -as
*** Report for user quotas on device /dev/mapper/centos-home
Block grace time: 7days; Inode grace time: 7days
```

		Space limits				File limits			
User		used	soft	hard	grace	used	soft	hard	grace
root	--	20K	0K	0K		2	0	0	
jack	--	0K	5500K	6000K		1	0	0	
mark	--	28K	0K	0K		7	0	0	

Step:6 Configure Grace Period for Soft Limit

Grace period is the amount of time during which soft limit can be exceeded, once the grace period reached then soft limit will become the hard limit.

Use the edquota command to set grace period ,

```
[root@linuxtech1 ~]# edquota -t
```

```
Grace period before enforcing soft limits for users:
Time units may be: days, hours, minutes, or seconds
Filesystem      Block grace period   Inode grace period
/dev/mapper/centos-home  7days                7days
```

NFS Server and Client Set UP

Network File System (NFS) is a file system protocol used to export local file systems over the network. Users can share directories and files with others over a network and interact with them as though they are mounted locally. NFS

protocol is not encrypted by default, and unlike Samba, it does not provide user authentication. Access to the server is restricted by the clients' IP addresses or hostnames.

<p>Step 1 install the required packages:</p> <p>The nfs-utils package provides the NFS utilities and daemons for the NFS server.</p>	<pre>#dnf install nfs-utils #rpm -q nfs-utils nfs-utils-2.3.3-31.el8.x86_64</pre>
<p>Step 2 start the nfs-server service, enable it to automatically start at system boot. By default, on CentOS 8 NFS versions 3 and 4.x are enabled.</p> <p>The other services that are required for running an NFS server or mounting NFS shares such as nfsd, nfs-idmapd, rpcbind, rpc.mountd, lockd, rpc.statd, rpc.rquotad, and rpc.idmapd will be automatically started.</p> <p>The configuration files for the NFS server are:</p> <p>/etc/nfs.conf – main configuration file for the NFS daemons and tools.</p> <p>/etc/nfsmount.conf – an NFS mount configuration file.</p> <p>The default settings are sufficient for us</p>	<pre># systemctl start nfs-server.service # systemctl enable nfs-server.service # systemctl status nfs-server.service</pre>
<p>Step 3 Create the file systems to export or share on the NFS server.</p>	<pre># mkdir -p /share/IT_Projects #chmod 777 /share/IT_Projects</pre>
<p>Step 4 export the above file system in the NFS server /etc/exports configuration file to determine local physical file systems that are accessible to NFS clients. The file contains export point and list of clients allowed to mount the file system at that point and comma-separated list of export options for that client.</p>	<pre># vi /etc/exports /share/IT_Projects 10.0.2.0/24(rw,sync)</pre>

<p>Some exports options(read man exports for more information)</p> <p>rw – allows both read and write access on the file system.</p> <p>sync – tells the NFS server to write changes to disk before reply (applies by default).</p> <p>all_squash – maps all UIDs and GIDs from client requests to the anonymous user.</p> <p>no_all_squash – used to map all UIDs and GIDs from client requests to identical UIDs and GIDs on the NFS server.</p> <p>root_squash – maps requests from root user or UID/GID 0 from the client to the anonymous UID/GID.</p>	
<p>Step 5 To export the above file system, run the exportfs command with the -a flag means export or unexport all directories, -r means reexport all directories, synchronizing <code>/var/lib/nfs/etab</code> with <code>/etc/exports</code> and files under <code>/etc/exports.d</code>, and -v enables verbose output.</p>	<pre># exportfs -arv</pre>
<p>Step 6 To display the current export list.</p>	<pre>#exportfs -s</pre> <p>Or</p> <pre>#exportfs -v</pre>
<p>Step 7 If you have the firewalld service running, you need to allow traffic to the necessary NFS services (mountd, nfs, rpc-bind) via the firewall, then reload the firewall rules to apply the changes</p>	<pre># firewall-cmd --permanent --add-service=nfs</pre> <pre># firewall-cmd --permanent --add-service=rpc-bind</pre> <pre># firewall-cmd --permanent --add-service=mountd</pre> <pre># firewall-cmd --reload</pre>

Step 8 On the client node(s), install the necessary packages to access NFS shares on the client systems.	# dnf install nfs-utils nfs4-acl-tools
Step 9 Run the showmount command to show mount information for the NFS server	# showmount -e 10.0.2.1
Step 10 Create a local file system/directory for mounting the remote NFS file system and mount it as an ntf file system. Confirm that the remote file system has been mounted by running the mount command and filter nfs mounts	# mkdir -p /mnt/projects # mount -t nfs4 10.0.2.1:/share/IT_Projects /mnt/projects # mount grep nfs
Step 11 To enable the mount to persistent even after a system reboot, run the following command to enter the appropriate entry in the /etc/fstab.	# echo "10.0.2.1:/share/IT_Projects /mnt/projects nfs defaults 0 0">>/etc/fstab

Samba Server Configuration

Samba is a free and open source protocol that allows files to be shared across windows and Linux systems in a simple and seamless manner. You can have a Samba server on a Linux server hosting various files and folders which can be accessed by windows clients.

Step 1) Install samba and necessary packages	<p>To verify samba is installed</p> <pre>#rpm -q samba</pre> <p>samba-4.11.2-13.el8.x86_64</p> <p>To install samba and its dependencies</p> <pre>#dnf install samba samba-common samba-client</pre>
<p>Step 2) Configuring Samba</p> <p>Windows and Linux system need be in the same workgroup</p> <p>View smb.conf.example for more configuration options.</p>	<p>a) Backup the samba configuration file</p> <pre>#cp /etc/samba/smb.conf /etc/samba/smb.conf.bak</pre> <p>b) Create a shared folder and give the right permission and ownership</p> <pre>\$ sudo mkdir -p /srv/samba/shared \$ sudo chmod -R 0777 /srv/samba/shared</pre>

c) Now modify samba configuration file.
\$ sudo vim /etc/samba/smb.conf
 Append the configuration below:
#=====Global
Settings=====
 [global]
 workgroup = **WORKGROUP** # This is
 default workgroup for Windows
 machines
 security = user # together with next line,
 enables Linux system users to log in to
 the Samba server.

passdb backend = tdbsam

printing = cups
 printcap name = cups
 load printers = yes
 cups options = raw

netbios name = **MYSAMBASERVER** # used
 to specify a server name that is not tied
 to the hostname, this becomes the
 machines's "Samba hostname"

interfaces = **10.0.2.0/24 127.0.0.0/8**
 #used to configure Samba to listen on
 multiple network interfaces.
 hosts allow = **10.0.2.** # the hosts
 allowed to connect.

#=====Share
Definitions=====
 [homes]
 comment = Home Directories
 valid users = %S, %D%w%S
 browseable = **Yes**
 read only = **Yes**
 inherit acls = Yes

[public] #A public accessible directory
 that is writable.
 comment = Public Stuff
 path = **/srv/samba/shared**
 public = yes

	writable = Yes browsable = yes force user = nobody [opt] #A public accessible directory that is read only. comment = Public Stuff path = /opt public = yes writable = no browsable = yes force user = nobody d) Run testparm command to verify that the configuration file \$ testparm # Or use testparm /etc/samba/smb.conf
Step 3) Allow samba service on the firewall	\$ sudo firewall-cmd --add-service=samba -- zone=public --permanent \$ sudo firewall-cmd --reload
Step 4) Create a Samba user and assign a password use pdbedit. Note: the user must be in /etc/passwd	#pdbedit -a root #pdbedit -a peter #pdbedit -L #pdbedit -L -v
Step 5) Start and enable Samba services. Samba is comprised of three daemons (smbd, nmbd, and winbindd). smbd The smbd server daemon provides file sharing and printing services to Windows clients. In addition, it is responsible for user authentication, resource locking, and data sharing through the SMB protocol. The default ports on which the server listens for SMB traffic are TCP ports 139 and 445.	\$ sudo systemctl start smb \$ sudo systemctl enable smb To confirm if smb service is running : \$ sudo systemctl status smb

<p>The smbd daemon is controlled by the smb service.</p> <p>nmbd</p> <p>The nmbd server daemon understands and replies to NetBIOS name service requests such as those produced by SMB/CIFS in Windows-based systems. These systems include Windows 95/98/ME, Windows NT, Windows 2000, Windows XP, and LanManager clients. It also participates in the browsing protocols that make up the Windows Network Neighborhood view. The default port that the server listens to for NMB traffic is UDP port 137.</p> <p>The nmbd daemon is controlled by the smb service.</p> <p>winbindd</p> <p>The winbind service resolves user and group information on a server running Windows NT 2000 or Windows Server 2003.</p>	
<p>Step 6) Test Samba from Linux system</p>	<pre>\$ smbclient -L 10.0.2.1 \$ smbclient -U peter //10.0.2.1/peter</pre>
<p>Step 7) Accessing Samba share from windows machine</p>	<p>From your Windows PC, press Windows Key + R to launch the Run dialog and type</p> <p><u>\\IP-address-of-samba-server</u></p> <p>Or open Windows File Explorer and in the address bar, type in \\10.0.2.1\peter</p>

Lab 9a Basic Web Server

Step 1 Install Apache.

"httpd" is the name for the Apache service in CentOS.

Apache is controlled by applying directives in configuration files:

/etc/httpd/conf/httpd.conf – Main Apache config file

/etc/httpd/ – Location for all config files

/etc/httpd/conf.d/ – All config files in this directory are included in the main config file

/etc/httpd/conf.modules.d/ – Location for Apache module config files

Verify httpd is installed

```
# rpm -q httpd
```

```
httpd-2.4.37-21.module_el8.2.0+382+15b0afa8.x86_64
```

```
#vi /etc/httpd/conf/httpd.conf
```

Looking for the "ServerName" section, we can change it to our own servername

Looking for the "DocumentRoot" section, this is where the webpage located. Default is **/var/www/html**

Step 2 Start and configure Apache to run on startup.

Note: When making changes to configuration files, remember to always restart the Apache service to apply the new configuration.

```
# systemctl start httpd
```

```
# systemctl enable httpd
```

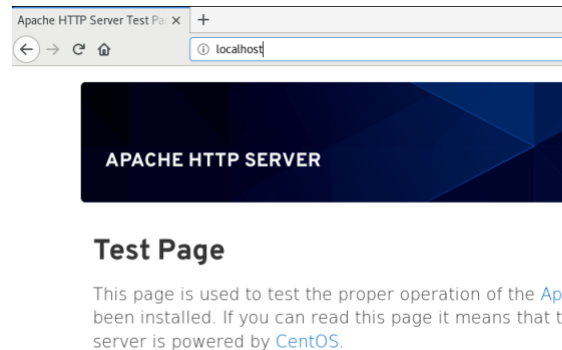
```
# systemctl status httpd
```

Step 3 Check the default Apache Test Page.

The new Apache installation has a default test page, but we can also create a custom test page in **/var/www/html/index.html**

You can designate a directory to store the files for customized website. Use the configuration files to point to the directory you choose. Some typical locations include:

/home/username/my_website



<div>/var/www/my_website</div> <div>/var/www/html/my_website</div> <div>/opt/my_website</div> <div>Apache log files to monitor web server:</div> <div>/var/log/httpd/ – Location of Apache log files</div> <div>/var/log/httpd/access_log – Shows a log of systems that accessed the server</div> <div>/var/log/httpd/error_log – Shows a list of any errors Apache encounters</div>	
Step 4 Adjust Firewall for Apache	<div># firewall-cmd --permanent --zone=public --add-service=http</div> <div># firewall-cmd --reload</div>
Step 5 Test Apache web page from another machine	
Lab 9b Configure Apache with SSL/TLS Certificates	
<div>The mod_ssl module provides SSL v3 and TLS v1.x support for the Apache HTTP Server.</div> <div>Step 1 install mod_ssl module.</div> <div>Note: Start httpd Apache webserver will start mod_ssl as well</div>	<div>#dnf install mod_ssl (mod_ssl is already installed in our VM)</div> <div>#rpm -q mod_ssl</div> <div>mod_ssl-2.4.37-21.module_el8.2.0+382+15b0afa8.x86_64</div>
Step 2 Open TCP port 443 to allow external access to Apache over HTTPS	<div># firewall-cmd --zone=public --permanent --add-service=https</div> <div>success</div> <div># firewall-cmd --reload</div> <div>success</div>

<p>Step 3 Generate self-signed SSL certificate.</p> <p>There are multiple options to choose from when you want to secure Apache with SSL/TLS certificates.</p> <ul style="list-style-type: none"> • Use self-signed certificates for test purposes. • Order for a commercially trusted server certificate from preferred CA • Use the free, automated, and open CA. <p>We will use Openssl command to generate SSL/TLS certificate</p> <ul style="list-style-type: none"> • req: It is used to create CSR as well as the self signed certificates. CSR- certificate signing request. • -newkey rsa:4096: This option creates a new certificate request and a 4096 bits RSA key at the same time. • -nodes: When this option is specified then if a private key is created it will not be encrypted. • -keyout /etc/pki/tls/private/httpd.key: Writes the newly created private key to the specified filename. Replace the filename accordingly. • -x509: This option outputs a self signed certificate instead of a certificate request. • -days 365: Used to specify the validity period for the self signed certificate generated. This therefore is valid for 365 days. • -out /etc/pki/tls/certs/httpd.crt: Specifies the output filename to write the self signed certificate to. 	<pre>#openssl req -newkey rsa:4096 -nodes - keyout /etc/pki/tls/private/httpd.key -x509 - days 365 -out /etc/pki/tls/certs/httpd.crt</pre> <p>Generating a RSA private key</p> <pre>.....++++++++++ writing new private key to '/etc/pki/tls/private/httpd.key' -----</pre> <p>You are about to be asked to enter information that will be incorporated into your certificate request.</p> <p>What you are about to enter is what is called a Distinguished Name or a DN.</p> <p>There are quite a few fields but you can leave some blank</p> <p>For some fields there will be a default value, If you enter '.', the field will be left blank.</p> <pre>----- Country Name (2 letter code) [XX]:AU State or Province Name (full name) []:NSW Locality Name (eg, city) [Default City]:Sydney Organization Name (eg, company) [Default Company Ltd]:UTS Organizational Unit Name (eg, section) []:IT Common Name (eg, your name or your server's hostname) []:www.it.netserv.edu.au Email Address []: -----BEGIN CERTIFICATE----- MDkyNTAwMTQyMloXDTIxMDkyNTAwMTQyMlowVjELMAkGA1UEBhMCQVUxDDAKBgNV</pre>
--	--

<p>After successful execution of the above command the private key has been written to /etc/pki/tls/private/httpd.key while the certificate has been written to /etc/pki/tls/certs/httpd.crt.</p>	<pre>AklUMQ0wCwYDVQQDDARZaW5nMIIBIjAN BgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKC Q4DoXWGEUCz6QoVSH9OifZZf4aGgr5KA/O +0p2iQYzPDq3gUqeAPsshE1WXFrBOx gOtd9li/BSq09Ybg4SDM++uAkLEDqr5Tigmb 0jf+8gfvjokdjrlz5kd3IDhxhIr cQ== -----END CERTIFICATE-----</pre>
<p>Step 4 Configure Apache to use SSL/TLS Certificate</p> <p>Check the Apache configuration for syntax errors:</p> <pre># apachectl configtest</pre>	<pre>#vi /etc/httpd/conf.d/ssl.conf</pre> <p>Change the certificate file and certificate key file to the one we just created.</p> <p>FROM:</p>

	<pre>SSLCertificateFile /etc/pki/tls/certs/localhost.crt SSLCertificateKeyFile /etc/pki/tls/private/localhost.key TO: SSLCertificateFile /etc/pki/tls/certs/httpd.crt SSLCertificateKeyFile /etc/pki/tls/private/httpd.key To check the Apache configuration for syntax errors: # apachectl configtest Or # httpd -t Syntax OK</pre>
<p>Step 5 Reload Apache and test Apache</p> <p>You may see an error. This is normal for a self-signed certificate! The browser is warning you that it can't verify the identity of the server, because our certificate is not signed by any of the browser's known certificate authorities. For testing purposes and personal use this can be fine.</p>	<pre># systemctl reload httpd</pre> <p>Open web browser, use https:// to test</p> <p>For our lab, we are going to use different webpage for testing, we create a new index.html file in /var/www/secure directory, so modify the DocumentRoot "/var/www/secure" to point to the new webpage, then test it with https.</p>
<h2>Lab 9c Virtual Hosting with Apache</h2>	
<p>By default, Apache web server is configured to serve or host only one website. If you plan to host multiple domains on your server, then you need to configure Apache virtual hosts.</p> <p>A virtual host is a separate file that contains configurations that allow you to set up a separate</p>	<pre># mkdir /var/www/a #echo "This is aaaa webpage" > /var/www/a/index.html</pre>

domain from the default one. The default virtual host is located at the `/var/www/html` directory. This works only for a single site. To create a separate virtual host for our domain, we will create another directory structure within the `/var/www` directory then edit the `httpd.conf` file to include information about the virtual host.

We also need a second domain name which can be resolved to an IP address. We can create a second domain name in the DNS server (you will need a separate entry in the `named.conf` file, and you will need to create a separate DNS zone file). An easier alternative, which is sufficient to prove your implementation of virtual hosts, is to define the new web server fully qualified domain names in the `/etc/hosts` file of the VM that you will use to send HTTP requests to the VM running Apache.

```
# vi /etc/httpd/conf/httpd.conf
```

```
<VirtualHost *:80>
```

```
    DocumentRoot "/var/www/a"
```

```
    ServerName www.it.netserv.edu.au
```

```
# Other directives here
```

```
</VirtualHost>
```

```
<VirtualHost *:80>
```

```
    DocumentRoot "/var/www/a"
```

```
    ServerName www2.it.netserv.edu.au
```

```
# Other directives here
```

```
</VirtualHost>
```

```
#vi /etc/hosts
```

```
10.0.2.3  www2.it.netserv.edu.au
```

```
Restart httpd
```

```
#systemctl restart httpd
```