Date 11/01/20

Imp

Date
Page No

Roles of System administrator :-

(i) Different components of a system can be handled by the admin.

(ii) The admin's experience is all about s/w development, the script, and many of the analytical and architectural skills.

(iii) Some of the performance of admin are discussed below :-

- Account provisioning : The system admin adds the account of a new user, remove the account of the users that are no longer active, handles all the account related issues etc.

- Adding & Removing H/W : When a new H/W is purchased or a H/W is moved from one machine to another, it needs to be configured. That is it is to be recognized by the system H/W. This concept of virtualization is a part of H/W configuration. For this, the admin needs to formulate different policies which will allow the H/W to be shared securely and fairely.

- Performing Backups : The performance related to backup is the most important job of the system admin and often it is ignored by the user. Backups are time consuming and boaring but they're absolutely necessary. In some systems backup are automated, still the system admin's job is to check that the backup is executed correctly & on schedule.

- Installing & upgrading s/w: If a new s/w is required, it must be tested and installed for different OS, different types of H/W & different types of installation processes. Once a s/w is working correctly, the users get informed about its availability and its location.

- Monitoring the system: Large installation required vigilant supervision. In such cases, some of the problems may not be informed by the users & some different paths are to be taken by the admin to document and report the issues. Regularly, the admin should ensure that the web services are working correctly, watch the log files so that the signs of trouble can be catched.
Eg:- In some system there is lack of system resources such as disk space

- Troubleshooting: System failures are inevitable so it is the job of the admin to diagnose the problem and call the experts if needed. Finding a problem is much harder than fixing it.

- Maintaining local documentation: According to the need of the organisation, the documentation is customised by the system admin which includes where the cables will run, how to keep the maintainance record, how to handle the H/W, recording the status of H/W and documenting local procedures and policies.

Date
Page No.

- Monitoring the security: The system admin job is to implement a security policy and to periodically check the security of the system is not violated. In a low secure system, it may involve few basic checks for unauthorized access. On a high secure system, it may include another elaborate network of traps and auditing progs.

- Fire Fighting: It is the work of the admin to help the users but some problems it to be handled by the users so the admin's job is to enhance the technical skills in the users so that the trouble can be well handled by them at some point of time.

Date 14/1/20

Unix and Linux

- Linux is the reimplementation of Unix with some more elaboration of Unix kernel.
- Linux has ~~posix~~ POSIX standard. It runs on several hardware ~~com~~ platform and is compatible with existing unix software.
- Both Linux & Unix are free, open-source and cooperatively developed.
- Linux includes some of the technical advancement that didn't exist in Unix. We can say linux is a clone of unix with some better advancement.
- Some popular linux distributions are :- RedHat, Solaris, Solaris 11, Open Solaris 2009.06

Date
Page No

man Page:

- man page or the manual page is used to read about the command.
- It constitutes the traditional online documentation
- man pages are typically installed with the system. Some of the program specific man pages can be installed with the new S/W packages.
- man page is the concised description of the individual commands, drivers, file formats and library routines

Organisation of man page:

- The systems divide the man page into sections with some system specific minor variation.
- In some cases, the sections may be further subdivided.

Function of man:

- 'man title' formats a specific manual page and sends it to the users terminal in a pager environment.
- The title may be a command, device, file or name of a library routine.
- The command 'man ~~stat~~ sync' gets the man page for sync command.

Date
Page No.

Storage of man page:

- The man pages are kept in directories under /usr/share/man
- In the linux system it is always a gif file to save space

Date 14/1/20

Scripting and Shell

Date
Page No

Shell: The standard language for the administrative scripts are defined by shell. When the script become complex, the editor is unable to give the output. So, to encapsulate few static commands, a major software project has been evolved which manages post configurations and administrative data comes under shell. It helps in bug tracking, support functions and do some design review which is needed for larger projects. Most system's default shell bash shell (Bourne-Again shell). The other two shells are C-shell and Korn Shell (KSH)

Shell Basics

Some of the common material which are applied on all the shell lineage are discussed here:

(i) Command editing: There are different types of command editors. For eg:- Vi editor, C-mac etc. for the command editors, we need some basic commands. Eg- for Emac ~~ctrl~~ control-e goes to the end of the line. control-a goes to the beginning of the line. For vi, set -o vi

(ii) Pipes & Redirection: Every process has atleast three communication channels available:
(i) Standard input (STDIN), (ii) standard output (STDO
(iii) Standard error (STDERR)
The kernel sets these three channel on behalf of the process. These channels communicate among themselves and get connected to the terminal or

Date
Page No.

to the file or to the n/w connection.
STDIN, STDOUT and STDERR are given as three integer values 0, 1 and 2 respectively to represent the file descriptor.
In an interactive terminal window, STDIN normally reads from keyboard and mouse. STDOUT will display on the monitor and STDERR will show the errors or warnings.
These shell interpreters can be symbolically given as:

< → STDIN
> → STDOUT
>> → STDERR

>> is also used for appending two files.

Date 18/1/20

Data
Page No

Variables and quoting

(i) Variable name are prefixed with a $ sign while their values are referenced.

(ii) Do not put spaces around the = symbol or the shell will mistake your variable name as a command.

(iii) While referencing a variable, surround the name with curly braces to clarify the parser.

(iv) There is no standard convention for naming the shell variables but all the caps variables are considered as environment variables and they're read from the global configuration file.

(v) All the local variables are lowercase letters with the components separated by underscore (_) Variable names are case sensitive.

* Note: (i) Environment variables are variable that is automatically imported into the bash variable namespace.
(ii) 'Export variable-name' is used to promote a some variable to an environment variable.

(vi) Shell considers strings enclosed in single and double quotes similarly but they have the additional effect of executing the contents of the string as a shell command and giving the output similar to a command.

Date
Page No.

Common filter commands

(i) Most of the filter command accept one or more filename on a command line.
Eg: STDIN and STDOUT can be used as a filter to process data. Other examples:-

- cut: used to separate lines into fields. The cut cmd prints selected portions of the input line. It is mostly used to extract some limited fields.
- sort: It is used to sort the lines. It has diff. options:

Option	meaning
-b	ignore leading whitespace
-f	case sensitive sorting
-k	specify the colum that form the sort key
-n	compare fields as int no.
-r	reverse sort order

- uniq: print unique lines. uniq is similar to sort -u but it also uses certain different options i.e.
  - -c to count the no. of instances of each li
  - -d to show duplicated lines
  - -u to find out only non-duplicated lines

- wc: count the lines, words and characters
  Counting the no. of lines, words and chara in a file is another common op^n and wc is a convinient cmd to do this. It also comes with option such as: wc -w & wc -c

Date
Page No

- tee: it is used to copy input to two places. The tee cmd is used to copy a file and send one copy to the terminal and one to the specified file.

- head & tail: head & tail cmd is used to read the beginning & end of the file. It is one of the common administrative opn for interactive use; head is used mostly in many scripts. The tail cmd comes with option -f where it waits for newline to be added at the end of the file.

- grep: grep cmd is used to search the i/p text and prints the line that makes a given pattern. grep has many options. such as:
  - c to print count of matching lines
  - i to ignore the case
  - v to print non-matching lines
  - l to find out lower case l

bash scripting

* bash is a simple script that automate the things by using the command line. when the script reaches more that 100 lines, the bash can not work, so the user has to move to either perl or python.

* bash comment start with # and continue to the end of the line. The newline can be given with a \n. We can put more that one line in a single statement by separating them with a ;.

Date
Page No.

- A bash script is a series of command line. The first line is known as shebang statement and declares the text file to be a script and is saved in /bin/bash.
- The kernel looks for this syntax (/bin/bash) when deciding to execute the file.
- In the windows world, the file have some extensions indicating the file type. But in Unix and Linux, the file permission which indicate whether file can be executed and by whom.
- The bash scripts is always saved with .sh

Date 21/1/20

Input and Output

- The echo and read command is used to control over the concept of i/p and o/p. In some OS level then printf is also used.
- The read is used for i/p, echo is for o/p (printf)
- The -n in the echo command supresses the new line

Q. Write a script to display "hello world".

Date
Page No.

Command-Line arguments and functions

- Command-line arguments to a script become variables whose names are numbers which is $1 is the first CL argument, $2 is second one and $0 is the name by which the script was invoked. The $# variable contains the number of CL arguments that are supplied and $* contains all the arguments at once.

- Different arguments to bash are treated as Command line arguments which makes the script. In some scripts, the arguments are optional. Only the code ~~###~~ is printed.

Variable scope

- Variables are global within a script but functions can create their own local variables with a local declarations.

Control flow

- Different control flows used in the scripts are if-then, if-then else.

- ~~#~~ if is the starting and fi is the ending. To chain it, other clauses used are: elif, else etc.

```
If [Stmt 1] ; then
elif [Stmt 2] ; then
else [Stmt 3] ; then
    [Stmt 4]
fi
```

Date
Page No

* For comparison in the bash, different comparison operators are used. Some are textual operator and others are symbolic operator used for numbers and strings.

String	Numeric	True if
x = y	x -eq y	x is equal to y
x != y	x -ne y	x is not equal to y
x < y	x -lt y	x is less than y
x <= y	x -le y	x is less than or equal to y
x > y	x -gt y	x is greater than y
x >= y	x -ge y	x is greater than or equal to y
-n x		x is not null
-z x		x is null

Bash file evaluation operator:

Operator	True if
-d file	file exists and is a directory
-e file	file exists
-F file	file exists & is a regular file
-r file	you have the read permission
-s file	file exists & is not empty
-w file	you've write perm. on file
file1 -nt file2	file1 is newer than file2
file1 -nt file2	file1 is older than file2

* Although the elif is useful but the case selection is often a better choice for clarity.

Date
Page No

Case -

St 0
St 1
St 2

esac
read

Arrays & Arithmetic

- Sophisticated calculation can't be done with the bash shell but atleast it offers arrays & arithmetic.
- In bash shell there is no distinguistion b/w a number 1 and a character string 1.
- bash has usual arithmetic, logical and relational operators. Arrays are present but the user has to call them.
- Arrays are ~~de~~ given by parentheses and the elements are separated by white space.

Regular Expressions

- The regular exp" is given with a shortend regex. They're used in UNIX command as grep and vi.
- Regular exp"s are recognised as possible grammars ~~used as~~ it uses arithmetic op" and parentheses for grouping. There is pattern matching for which grep command is used.

Date
Page No.

* Perl and python regular exp" has shown their power. Regular exp" don't come under scripting language but they're useful.

Special characters

* Some basic special characters used in regex:

	Symbols	What it matches
(i)	.	matches any character
(ii)	[chars]	matches any character from a given set
(iii)	[^chars]	matches any character not in given set
(iv)	^	matches the beginning of a line
(v)	$	matches the end of a line
(vi)	\w	matches any word character (A-Z a-z 0-9 _)
(vii)	\s	matches any white space character
(viii)	\d	matches any digit from 0-9.
(ix)	|	matches either the element to the left or to the right
(x)	(expr)	group of elements, allows matches to be captured
(xi)	?	allows 0 or 1 match of the preceding element
(xii)	*	allows 0, 1 or many matches of the preceding element

(xiii) + allows 1 or more users of providing signal

Date 25/1/20

Shell Program

* It is one of the most powerful features of any UNIX system.
* A shell program contains high level programming language features such as variables for storing data, decision making control (if, else), looping abilities, function calls for modularity.
* A shell program can also contain UNIX command and pattern editing utilities
* Steps to create shell program:
  (i) Specify shell to execute program that means the script must begin with # symbol.
  (ii) Make a shell prgm executable using chmod cmd.
  (iii) Formatting the shell program.
  (iv) Comments: Start comment lines include the comment lines to describe section of the program so that later anyone can understand it.
  (v) Guidelines: Use good names for script and variables. The comment line will start with #. Use indentation to reflect logical and nesting

Date
Page No.

The /bin directory

- The /bin is a standard sub-directory of the root directory. It contains the executable programs which are available to attain minimal functionality for the purpose of booting.

- The root directory which is given as '/' is the top level directory in the hierarchy of directories. It is the directory that contains all other directories and their subdirectories as well as the files on the system.

- A directory in a UNIX like OS is merely a special type of file that contains a list of names of objects. A file is a named collection of related information that appears to the user as a single contiguous block of data.

Scripting

bash is not only an excellent command line shell but scripting language in itself.

Difference b/w Programming and Scripting Language

- Programming languages are generally a lot more powerful and faster than scripting language.

- Programming languages generally starts from a source code and are compiled into an executable code.

- This executable code can not be easily ported into different OS
- Scripting language also starts from source code but is not compiled into an executable code. Here, an interpreter reads the instruction in the source file and executes each instruction
- Interpreter programs are generally slower than compiled programs.
- It can be easily ported in any OS

Special shell parameters:

- $#: It is the no. of parameters passed.
- $0: It returns the name of the shell script running and its location in the file.
- $*: gives a single word containing all the parameters passed to the script.
- $@: gives an array of words containing all the parameters passed to the script.

While:

```
while exp"
do
    stmts
done
```

Date 5/11/20

Ch3:

Booting and Shutting Down

Date
Page No.

Imp

Bootstraping

- Bootstraping is the standard term for starting of a computer
- The OS facilitates the necessary files during the startup process.
- During bootstraping, the kernel is loaded into memory and begins to execute. A variety of initialization tasks are performed and the system is then made available to the users.
- Boot time is more vulnerable than other system activities because if errors in configuration, missing or unreliable equipment and damaged file system is there then it can prevent a computer from booting.
- Boot configuration is often one of the first tasks of the administrator on a new system and while adding new hardware.
- When a computer is turned on it first executes the boot code that is stored in Rom. That code attempts to figure out how to load and start the kernel.
- Then the kernel initiates the system's H/w and the system's init process so init process is always process no. 1.

Date
Page No.

- Before the system is fully booted, the file system must be checked and mounted and the system daemons gets started.

Recovery

8 Qs

Imp

1) Steps in Boot Process:-

→ A typical bootstraping process consist of 6 distinct phases

(1) Reading the bootloader from the master boot record

(2) Loading & initialization of kernel.

(3) Device Detection & configuration

(4) Creation of Kernel Processes

(5) Administrator Intervation. (Single User mode only)

(6) Execution of system startup script

① Kernel Initialisation:-

→ A kernel is itself a program and the first boot-strap process is to get this program into memory.

→ The path name of the kernel is vendor dependant but usually it is like UNIX / VMUNIX / VMLINUZ.

→ Most system implement a two stage loading process on the first stage, the system ROM loads a small boot program into memory from disk. This program is called as bootloader.

→ BootLoader Arranges for kernel to be loaded.
→ The memory is preserved for the kernel, the kernel clears / initiates the system about the availability of RAM. Once the kernel is given the portion of memory (RAM) the other user level process are loaded.

(2) Hardware Configuration:

→ One of the kernel first job is to schronize the system environment to check the hardware present.

→ It shows various system buses, and other hardware devices attached along with some configuring information, about each device.

→ The kernel loads the device drivers as independent kernel modules.

→ The hardware configuration is a transparent process for administrations under linux.

→ Generally kernels detect most of the hardware automatically.

(3) Creation of Kernel Processes.

→ Once basic initialization is complete, the kernel creates several continous processes in user space.

→ They are called spontaneous process because they are not created by the normal system called Forks.

→ The exact no of spontaneous processes varies according to the OS although, init — is the first process considered in most unique system sched is considered as pid 0.

2nd

→ But init is really a full fledged user process & others are functions in kernel. So we consider init as the first process.

Operation Intervation :-

→ It is also called single user mode or recovery mode.

→ If the system is brought up with the recovery mode, a command line flag is passed by the kernel which notifies to the root that the root password to be used and how the system is in single user mode.

→ In single user mode it cannot run the programs only some files will run such as /bin/sh, /etc, /bin.

Execution of Startup Scripts :

→ The system is ready to run the startup script, it is recognized by unix but still the system is not fully booted.

→ The startup script are normal script or normal shell script run by init.

Post Process Configuration :

→ After the initialization script is run, now the system is fully operation, system daemons such as cron, sendmail, servers are accepting the service connections.

→ init continues to perform even after booting is complete.

→ init has several user levels that determine the system resources that are enabled.

Booting a PC

→ PC booting is a lengthy process which requires quite a lot of background information.

① When a machine boots it begins by executing the code saved in the ROM. The exact location & nature of code varies depending upon the type of the machine.

② A UNIX system generally depend on a typical firmware that knows how to use the devices connected to the machine.

③ The initial boot code is generally called as a BIOS (Basic Input Output) and it is extremely simpler than a firmware. PC have several level for BIOS one for the machine itself, for video card, network card for SCSI card and other peripherals.

Imp

→ But : init is really a full phased user process & others are portions of kernel. So we consider init as the first process.

Operation Intervation :-

→ It is also called single user mode or recovery mode.

→ If the system is brought up with the recovery mode, a command line flag is passed by the kernel which notifies to the root that the root password to be used and how the system is in single user mode

→ In single user mode it cannot run the programs only some files will run such as /bin, /etc, /bin.

Execution of Startup Scripts :

→ The system is ready to run the startup script, it is recognized by unix but still the system is not fully booted.

→ The startup script are normal script on normal shell script run by init.

Boot Process Compression:

→ After the initialization script is run, now the system is fully operation, system daemons such as DNS, SMTP, Servers are accepting the service connections.

→ init continues to perform even after booting is complete.

→ init has several run levels that determine the system resources that are enabled.

Booting a PC

→ PC booting is a lengthy process which requires quite a lot of background information.

① When a machine boots it begins by executing the code saved in the ROM. The exact location & nature of code varies depending upon the type of the machine.

② A UNIX system generally depend on a typical firmware that knows how to use the devices connected to the machine.

③ The Initial Boot code is generally called as a BIOS. (Basic Input Output) and it is extremely simpler than a firmware. PC have several level for BIOS one for the machine itself, for video card, network card for SCSI card and other peripherals.

→ BIOS knows about some of the devices that are on the motherboard, typically IDE and SATA controls, network interfaces, Power controller and other system hardware.

→ The BIOS allow the user to select from which device you want to boot from.

→ Ex: DVD drive, Hard Drive, USB.

→ Once the BIOS is configured it tries to read from the first block of the device and these 512 bytes segment is known as master boot record or MBR.

→ MBR contains the programs that tells the computer from that which partition, the secondary Boot programs will be loaded & the secondary boot program is known as Boot loader.

1st Feb

GRUB:

→ Grand unified Boot Loader.

→ GRUB is developed by GNU project. It is the default boot loader for most of the unix & linux with intel processes.

→ There are 2 branches of GRUB lineage, the original GRUB is now called as GRUB legacy and the new version is GRUB 2.

→ The default GRUB reads it boot configuration from /boot, /GRUB/menu.lst or /boot/GRUB/GRUB.CONF

→ GRUB reads the configuration file at startup time so it allows dynamic changes at each system boot.

→ GRUB automatically boot if it doesn't receive any keyboard input within 10 sec of configuration.

→ GRUB loads the kernel from /vmlinuz or from the file /boot/GRUB/splash.xpm.gz

→ GRUB supports a powerful cmd line interface & also facilitates file editing configuration to enter into the cmd-mode type c from the GRUB boot screen.

→ It has shell like features.

Some common cmd of GRUB:-

	Cmd	Meaning
1>	Reboot	Reboot the system.
(2)	Find	Find files and mountable partition.
(3)	Root	Specify the root device
(4)	Kernel	Loads a kernel from root device.

⑤	help	Gets interactive help for a cmd.
⑥	Boot	Boots the system from the specified Kernel image.

Working with the Startup Scripts:-

→ After exiting from single user mode: init executes the system startup scripts.

→ These scripts are interpreted by sh or bash.

→ The exact location, content and organization of the script depends on the vendor.

→ Scripts are saved in /etc/init.d and are made in the directory.

→ The startup script both start & stop the service so the startup script also allows to orderly shutdown the system.

→ Some task that are often performed in the startup script are ① setting the name of the comp
② setting the time zone.
③ Checking the disk with fsck
④ Mounting the system disk.
⑤ Removing old file from /temp directory.
⑥ configuring network interfaces.
⑦ starting of demons & network services.

Imp

INIT and its Run-levels:-

→ Init is the first process to run after the system boots.

→ It is the most imp. demons.

→ It has alway PID 1 and is an ancestor of all user processes.

→ Init defines atleast 7 seven run-levels, each of which represents a particular service that should be running in the system.

→ Each run level has different defination, such as at level zero
① The system is completly shutdown.

at level 1 & S:
→ represent single user mode.

at level 2 through 5 (2, 3, 4):
→ include support for networking

at level 6:
↳ It is a reboot level.

Rebooting & Shuting down:-

→ Traditional unix & linux machine are very touchy about the shutdown process,

→ Modern system have become less sensative because they use robust file system for handle

diff" processes:-

→ It always a good idea to shut down the machine with a nice possible way because improper shut down can result in anything subtle / problematic which can create a major effect on the system.

Ex:- Databases that are not haulted properly are notorious for corruption and integrity issues.

→ On a consumer oriented operating system, rebooting of the OS can lead to many problems so rebooting is effective in a smaller percentage of cases.

→ Reboot is allowed when there is a modification in the startup script which makes significant changes.

ShutDown:- The shutdown cmd is the most safest, considerate way to initiate a halt or reboot or to return the system to single user mode.

→ The shutdown cmd wait a while before shutting down a system during this waiting period shutdown sends messages to the locked in user at progressively shorter intervals warning them to stop the processes until the shutdown event has occured.

→ The user cannot login when a shutdown is initiated.

→ Most versions of shutdown allow the machine to halt, to go to the single user mode or to reboot.

Halt & Reboot:-

→ The halt cmd performs the essential duties required to shutdown the system.

→ Halts logs the shutdown, kills the nonessential processes, execute the sink system call, wait for file system to complete its counts and halts the kernel.

→ Reboot is almost identical to halt but is causes the machine to reboot instead of halting.

→ Reboot is also called as shutdown -r

—x—

Ch- -: Access Control & Users:-

Traditional Unix Access Control:-

→ In the simplest version of Unix there was never a single point of access control.

→ During the system design some general rule has been framed for the access control system.

Rules:

① Objects have owners (file & processes). Owners have broad control over their objects.

② The new objects created can only be handled by the owner.

③ The special user account called as root can act as the owner to any object.

④ Only root can perform certain sensitive administrative operations.

Types of Access Control:-

① File System Access Control:- In the traditional model every file has an owner and a group. The group is also known as group owner.

→ The owner can set a restricted permission for its file so that no one can access it.

→ When the owner of a file is more than 1 people it is known as group owner. Groups are traditionally defined in /etc/group file.

→ The ownership of a file can be determined with ls/l <filename>.

→ Both the kernel & the file system can track owners and groups as numbers rather than a text name.

→ The owners have user identification no. on UID's & group is GID's for group.

→ The text name that corresponds to UID's or GID's are only defined for the convinience of system users in a human readable format.

Process Ownership:-

→ The owner of a process can sent the process signal and can also reduce the processes scheduling priority.

→ The process have multiple identity associated with them, it may be a real identity, an effective identity or an saved UID's & identity.

→ It may be a GID.

→ All these identities are generally no. on the real nos.

Root Account:- 8/Feb/2020

→ The root account in unix is to be used by the administration. It is also known as super user account.

→ The characteristics of root account is it has UID=0.

→ The users are not allowed to make any changes in the root account along with its password.

→ Traditional UNIX allows the super user to perform any valid operations on any file or process. Some of the restricted operation performed by the root account are changing the root directory of process with chroot.
- creating device file
- setting the system clock
- Raising usage resource limit & process priority
- setting the system hostname.

- configuring network interfaces.
- opening network ports
- shutting down the system.

Q) What is the function of root account?

setuid & setgid:

→ once the root account has given permission to any account to be the owner it becomes a normal user process then it will be given a user name or a group name having certain UID or GID.

→ In traditional UNIX access control these identity, substitution system has been implemented by the kernel and the file system in collaboration.

→ When the kernel runs an executable file, it has to set a UID which, setUID or setgid permission bit set.

→ In this way the users are privileged for execution of specific cmd & able to change the password

Modern Access Control:

→ The traditional UNIX system was simple, predictable & capable of handling majority of access control requirements.

→ All the UNIX and UNIX system support these model & it remains the default approach that is mostly widely used today.

→ It has certain obvious drawbacks that are from the security perspective the root account represent a potential single point of failure. If it is compromised the integrity of whole system is violated. There is no limit that an attacker can inflict.

→ The only way to subdivide the special previlages of the root account is by writing the setuid programs but on modern day system as the internet steedy stream of security update itself it difficult to write truely secure software.

→ The security model is not strong enough to be used on a network of computer, no user can be trusted to accurately represent the ownership of a process.

→ Many high-security environments enforce convention but it can not be implemented with traditional unix system. The traditional UNIX system security depends on the good will & skill of the individual user in regard of system maintainance.

Q) What is the drawback of traditional unix system & how the modern system can overcome it.

Types:

① Role-based Access Control: Sometime Known as RBAC.

→ The basic ideas is to add a specific layer of

indirection for access control calculative.

→ Instead of permission being assigned directly to the user they are assigned to intermediate constructs known as Roles

→ Roles are assigned to the user.

→ To make the access control decision the access control library assign the roles to the current user & checks the roles have been given the appropriate permission.

→ The RBAC model makes it practical to manage large collections of possible permission.

→ All the permission comes in hierarchy, here the administrator uses a simple system model that supports RBAC. In this way the root account is divided into different fragment that are separately assigned with jobs.

SE-Linux:

→ SE-Linux is a NASA project that has been freely available since 2000. It has been integrated with 2.6 series of Linux Kernel and is also available in most of the current distributions.

→ The primary focus of SE-Linux is to enable mandatory access control (MAC).

→ An access control system in which all permissions are assigned by the administrator.

→ Here the user cannot delegate any access or permission parameters to any object they own.

Imp ★ POSIX's Capability

→ The Linux system do not make use of SE-Linux extensions, theoritically they are capable of dividing the root account according to the POSIX standard for capabilities.

→ The capability specification can be assigned to executable program.

→ The program then aquire the specified capabilities when they are executed.

Imp ★ PAM (Plugable Authentication Module).

→ It is an Authentication technology rather than an access control technology.

→ Most of the system PAM is the important component of the access control chain on most of the system.

→ In the past user password were checked against the /etc/shadow file at login time. So that an appropriate UID could be set for the user's shell.

→ In the modern world it is more flexible because we are using cryptography, biometric identification devices which are open systems hence

PAM is a wrapper for a variety of method specific authentication libraries.

→ The administrator uses a specific authentication method which they want the system to use.

Kerberos

→ It is a 3rd party authentication system.

→ Like PAM Kerberos deals with authentication rather than access control.

→ PAM is an authentication framework and Kerberos is a specific authentication method.

→ They are used together where PAM is the wrapper and Kerberos is the actual implementation.

→ Kerberos uses a 3rd party who is trusted to perform authentication for the entire network.

Access Control list (ACL):-

→ All these are under the modern access control.

→ The file system access control is central to both UNIX & LINUX so most of the common versions use ACL.

→ It is a generalized form found in traditional UNIX system for giving permission either to a user, group & others.

→ ACL are the part of file system & implementation so they are supported by most of the file system in any form.

→ ACL supports generally in two forms
① Posix Draft Standard
② Standarized NFS version.

Difference bet^n PAM & Kerberos.

Real World Access Control:-

→ There are different way applied in modern UNIX system to validate a user account.

→ Also there are add-on tools which can became a bridge bet^n the traditional & access control & modern access control.

Ex: Sudo.

Choosing a root Password:- Generally root does not need a password because. always it is to be remembered.

The most important characteristic of a good password is its length(8) The root password should be 8 character long.

→ If it is seven character it is substentially

→ DES allows 8 characters passcoond always.

→ The passcoond should be a random sequence of letter, punctuation and digits.

→ To make the root passcoond safe and protect the system from any kind of security risk the administraton should change the root passcoond atleast every 3 months

→ Everytime who knooks the passcoonds leaves the site.

→ Whenever garbage the almoctraton thing, the security may be compromise

SU: (Substitute User Identity).

→ SU cs a command used to access the root account.

→ ACU provide the root password and startup the root cell.

→ To exit from the cell the command is to keen.

→ SU doesnot record the cmd executed as root

→ It maintains a log file from which it can be know who became root & when.

→ SU have been ~~hastly~~ largly surpressed by using sudo.

→ SU is mostly used during emergency.

SUDO

→ It is the limited SU, it is mostly used in RBAC and OE 10 AC linux.

→ When a system root account is used by several administraton then it is very difficult to know the status of the root account. to overcome these problem a program called sudo is used.

→ Sudo keeps a log-file for all the cmd-lines that were executed.

→ It also keeps the record of the user, the directory from which they are runn, & who have requested to runn.

→ All these are stored on syslogfile and it is saved in the file system.

→ Sudo takes the argument on a command-line and consults the file /etc/sudoers.

→ The sudoers file is designed so that a single version can be used on many different host at once.

→ The sudoers file contains different line on a no. line from that the first five no comment line define groups of host and commands that are reffered for the perimission specifications

later in the file.

→ Each permission specification line includes information about

(1) The users to whom the line applies.

(2) The host on which the line should be implemented.

(3) The cmds that the specified user can run.

(4) The user who can execute the command.

Difference between SU & SUDO.

Advantages of Sudo

(1) Accountability is much improved because of command logging.

(2) Operators can do chores without unlimited root previlages.

(3) The real root password can be known to only one or two people.

(4) It's faster to use sudo that to use SU during a logon to the root.

(5) Previlages can be revoked without the need of changing the root password.

(6) A canonical list of all user with root previlage is maintained.

(7) There is less chance of a root shell being unattended.

(8) A single file can be used to control the access of the whole network.

** /etc/password file

*** /etc/password file

→ The /etc/password file is a list of user recognised by the system.

→ It can be extended or replaced by a directory service so its authority is only a standalone system.

→ The system consults the /etc/password file at login time to determine a user's UID and the home directory.

→ Each line in the file represents one user & contains seven fields separated by colons.

→ The seven fields are
(1) Login name
(2) Encrypted Password Place holder
(3) UID
(4) Default GID
(5) GECOS (It gives the info about full name, office, any extension, home phone)
(6) Home directory
(7) Login Shell.

Login Name:

→ Also known as the user name, must be unique, it depends on the os, it has length & character set restriction.

→ Login Names can never contains (:), newlines because these characters are used as field separators.

System	Length	character set	FIRST	Special Rule
LINUX	32 bit	a-z, 0-9, _, -	a-z, _	It is man generated
Solaris	8 bit	A-Z, a-z, 0-9, ., _, -	A-Z, a-z	At least one lower case letter
HP/UX	8 bit	A-Z, a-z, 0-9, _	A-Z, a-z	
AIX	8 bit	POSIX, $, no spaces, quotes, #, =, /, :	~, @, -	All the uppercase letters.

→ UNIX systems limited its permissable characters to alpha numeric and imposed an 8 character length limit.

→ Login name are case sensitive but if it ignored in case of email address.

→ login name is easy to remember

→ That must random sequence of character

→ Donot make good login name.

→ Avoid Nickname

→ Choosing a login name eventually result in duplicate name on name that are too long. so the user has to make some exception.

Encrypted Password:

→ Modern system put a placeholder for the encrypted password in /etc/password file & then prompt the user for a real password on first login

→ There are different encryption schem used in UNIX system, different cryptographic algorithm are used for this.

→ Algorithm used are DES, MD5, BLOWFACE etc.

→ The password length is an another important issue which often determine the standard of algorithm & completely defined by the algorithm.

Table: Password Encryption Algorithm:

System	Min	Max	Algorithm
LINUX	5	8	Crypt, MD5, Blowface
Solaris	6	8	Crypt, MD5, Blowfish and SHA256

UID

→ The UID identifies the user to the system.

→ Login Names are provided for the convinience of user but the software & file system uses UIDs internally.

Imp
→ UIDs are generally unsigned 32 bit integers.

→ Always the root has UID zero.

** Note: Some special UID used for pseudo users also know as nobody, they are usally assigned with a higher valued such as -1 or -2 in the UID field which is the highest or next highest possible value for UIDs. The nobody login is used for the root user on one machine tries to access the file that are NFS mounted from another machine & that does not trust the first machine also known as nobody account

→ UID should be kept unique across whole organisation that means a particular UID must be assign to the same login name, to the same person, on every machine.

LDAP: It is a popular management tool for UIDs and user account information.

25/Feb

GID Number:

→ Like UID a group Id number is a 32-bit integer.

→ GID zero is reserved for the root of the group

→ In the traditional system the group was used for accounting purposes that is the consume CPU time, the time elapsed in login, the Kilobytes of disk used, etc. but in modern system, group are used to share to access the file.

→ The Gecos field is sometime used for record the personal information about each user.

→ It has no well defined system.

→ The most commonly used command is finger command.

→ The Gecos Entry are interpreted by the following order
1. Full Name.
2. Office No & Building
3. Office Telephone Extension
4. Home, Phone No.

→ All the field are separated by ,.

→ If the user wants to change their Finger information the command used is
→ chfn

* Home Directory:

→ The user's home directory is default directory at login time

→ If the home directory is loaded & mounted over a network file system, then it may be unavailable if their is a server problem / network problem.

→ If home directory is missing at login time then one error msg will come, no home directory at the time of login & the system will not continue.

Login Shell:

→ The login shell is a cmd interpreter such as Bourne shell or C shell but it can have a program with extension .sh & bash.

→ Some system permits the user to change their shell with chsh command.

/etc/shadow file:

→ A shadow password file is readable only by the super user and keep the encrypted password saves from the cracking program & hackers.

→ It also include some additional account info. that is not provided in /etc/password file

→ Now a days shadow password are the default one nearly in all systems.

→ IBM calls this file /etc/security/password file.

→ The content & the format are same for both the file.

→ The shadow file is not a superscript of password file.

→ So both the file are to be maintained in diff" ways.

→ In this file each line contains 9 fields, separated by (:)

① Login Name
② Encrypted Password
③ Date of last Password change
④ Minimum No. of Days Password change
⑤ Maximum No. of Days " "
⑥ No. of Day in advance to warn the use about password expination

one script will be there in sem.

→ Linux: Days after password expiration the account is disable.

Solaris: Days before account automatically expire

Account Expiration Date:

A reserved field that is generally empty except on solaris.

① login Name is same as /etc/password file.

- The field connects a user password & the shadow entry.

② Encrypted Password is identical in concept & execution to /etc/password file. The algorithm as crypt, md5, blowfish etc.

③ The last Change field records the time, ~~atleast~~ at which the user password was last changed. The field is filled by the password cmd.

④ The no of days that must elaspe bet^n the password changes. This idea force authentic changes by preventing the user from immediately reverting to a familiar password after a required change. This field sometimes may be dangerous, when a security intrusion has occure.

⑤ Sets the max. no. of days allowed bet^n password allowed. This system allows the administrator to ensure password aging.

⑥ Sets the no. of days before password expiration so that the login will begin to warn the user for the impending expiration.

⑦ For different OS the interpretations is different under linux the 7th field specifies how many days after the maximum password age has been reached & the login will be expired but in solaris the account automatically get disable if the user has not logged in within the no. of days specified.

⑧ It specifies the day on which the users account will expire.

ex: Jan 1 1987.

The user is not allowed to login after this date until they field has been reset by the administrator.

⑨ This is reserved for future use incase of linux but solaris uses the last 4 bits to count failed login attempt.