**LINUX ADMINISTRATION:**

* Kernel is the chief of programs acting as chief of operations. Kernel is responsible for starting and stopping of other programs, handling requests for memory, managing network connections, etc.
* Two standard process are being followed in linux distributions.
  + **File Hierarchy System (FHS):** Responsible for a standard directory structure, this is needed so an application can run across multiple linux distributions.
* L**inux standard base**: it specifies what a linux distribution must have in terms of tools and libraries.
* Two types of kernels are used in operating systems:
  + **Monolithic Kernel**: Provides all the services that a application needs to run.
  + **Micro Kernel**: Provides a bare minimum of libraries for the operating system to function on an machine.
  + In linux the GUI and Linux kernel are separate, so even if the GUI goes down the Linux Core will still be up and can be accessed via terminal.
* Linux does not has a registry, instead the configuration files are stored as normal text files and can be modified using any text editor available on the LINUX machine, due to this there is no standard of writing the configuration files.
* In linux the configurations are stored under **/etc** directory.
* LINUX users **PAM(Pluggable Authentication Modules)** to looks up the users and group information for applications.
  + Using this any tools e.g. kerberos, LDAP or NIS can be used for authentication.
* **Directory Structure:**
* / : this is the root of the directory structure , similar to C:\
* /boot : all the files required for system boot reside in this directory
* /usr: this where all the program files reside similar to c:\Program Files,
* /home: this is the directory where the user directories for all the users accessing the LINUX OS reside, it is equivalent to C:\Users.
* /var: this directory contains the log files for system events or the applications.
* /tmp: this the temp directory
* SWAP: this is where the virtual memory is stored; this is not user accessible and is only used for managing virtual memory.
* **Shell:** shell is a command language interpreter that executes the commands. Log in shell is the first program that runs when a user logs in to the LINUX OS.
  + **BASH SHELL:** It is a variant of shell and have various built in tools that can be executed.  
    It has the capability of launching other programs and controlling them often termed as JOB control.
* **Environment Variables:** Every instance of the shell or any program that is running has a environment of its own, these settings are controlled by environment variables.  
  **printenv** command prints the list of all the environment variables.
  + **Setting Environment Variables:** var=<var\_value>  
    export <var\_name>. Export command makes the environment variable available to use outside the current shell for other programs too.
  + **Unsetting the environment Variable:** An environment variable can be unset by using unset command.  
    **unset <var\_name>**
* **Pipes:** pipes are the mechanism by which the output of a program can be passed on to the input of another program  
  e.g. grep utility, this utility tries to match the lines with the parameters specified.  
  **ps –ef|grep <term\_to\_be\_matched>**
* **Redirection:** through redirection we can take the output of a command/program and send it directly to a file.Redirection comes in three classes: **output to file,append to a file, send a file as input.**
* **ls > /tmp/file.txt** : inserts the output of the file to file.txt
* **echo “listing” >> /tmp/file.txt:** appends “listing” to file1.txt.
* **grep “list” < /tmp/file.txt :** uses file as input for grep command to list term.
* **Files Ownerships and Permissions:** In linux everything is a file.
  + **Normal Files:** text files, config files and executables files.
  + **Directory Files:** these are special instance of normal files; directories list the location of other files.
* **Hard links:** In linux every file gets a i-node, i-node keeps tracks of attributes and location of a file.  
  If we need to refer a single file name to be reference by two different file names, this can be achieved by creating a hard link. This can only be achieved for the files which are present on the file systems and not on network drives.
* **Symbolic Links:** Unlike Hard link, a symbolic link creates a shortcut for a file by its name rather than I-node. This can be done for the files present on the network drive also.  
  The links created are also referred to as symlinks.
  + **Block Devices:** Files of type block device are used to interface with devices such as disks.  
    these have two identifying traits **major number and minor number**.  
    [devsys@devops-master ~]$ ll /dev/sda

brw-rw----. 1 root disk 8, 0 Jan 22 09:43 /dev/sda  
b -> represents block and number 1(passed as parameter to device driver) and 8(device driver number) are minor and major numbers.

* + **Character Devices:** these are the devices that work one character at a time(example modem), these also have major and minor numbers.  
    [devsys@devops-master ~]$ ls -l /dev/ttyS0

crw--w----. 1 root tty 4, 64 Jan 22 09:43 /dev/ttyS0

* **Change Ownership**: this is achieved by using the command chown, this can only be done by the root user.  
  chown –R <user> <filename/directory\_name>  
  chown –R <user.groupname> <filename/directory\_name>
* **Change Group:** the group permissions of a file or directory can be changed by using command  
  chgrp –R <group\_name> <file\_name>
* **Change Permissions**: any file in linux has three permissions and they are categorized into read, write and execute for the (user, user-group and others)  
    
  401 401 000   
  r - x r-x ---  
  5 5 0  
  user user-group others  
    
  to change permissions , chmod command is used  
  syntax: **chmod 764 <file\_name>** , this will give full permission to user, read write permission to user group and read permission to others.  
  **alternate:** chmod u+rwc,ug+rw,o-r
* **File Management**:
  + **Copy files (cp):** this command can be used to copy files from one location to another  
    syntax: cp <file\_name> destination.  
    cp – options (-R, -f)
  + **Move (mv):** this command can be used for moving files from one location to another (similar to cut paste in windows), the same command is used for renaming a file.  
    syntax: mv <file\_name> <destination>, mv <old\_name> <new\_name>  
    mv –options (mv –f ,-i)
  + **Link files (ln):** ln command is used for making links between two files. This can be used to create hard link and softlink for a file.  
    Syntax: ln <file\_name> <new\_file> this creates a hard link.  
    ln –s <file\_name> <new\_file> this creates a sym link.
  + **Find a file (find):** this command is used to search for a file in a directory and has a variety of options available. **Syntax:** find <directory\_to\_search\_in> -options   
    find . –name “name\_of\_file” –size +100k |xargs rm or find . “name” –size +100k –exec rm {} \;  
    above command finds a file by its name in the current directory of size greater than 100KB and runs the rm command on them.
  + **File Compression:**
    - **Gzip :** is a file ompression command , that compresses the file and deletes the original file.  
      gzip <file\_name>  
      uncompress: gzip –d <gzipped\_file\_name>
    - **Tar:** tape archive , compression tools, takes multiple files and combines them into a single file with compressed size  
      tar <option> <zipped\_file\_name>.tar/.zip file\_name\_to\_be zipped
  + **Mkdir:** create directory **,  
    mkdir <dir\_name>  
    mkdir –p <complete\_path>**
  + **Rmdir :** removes directory
  + **Which:** looks for a file in the $PATH
  + **Whereis:** looks up the file in $PATH and prints the path of the file.
* **Package Management:**
  + **RPM:** Redhat package Manager.  
    to check the packages installed on a machine, following command can be used   
    rpm –query –all   
    to find about a particular package, use the following command  
    rpm –query <package\_name>  
    to find what a package is and what it does , use   
    rpm –qi <package\_name>  
    to find the files contained which in the package , use  
    rpm –ql <package\_name>  
    to find if any configuration files were written for the package, use  
    rpm –qc <package\_name>  
    - Installing an rpm package: get the rpm package to be installed from the repository, download it on the Linux machine.  
      To perform a test install, i.e rpm goes through a pseudo install process without actually installing the package and checks for all the dependencies. We can use  
      rpm –install –verbose –hash –test <rpm\_package>  
      To perform a real install : rpm –ivh <package\_name>
    - Uninstall of rpm package can be done by : rpm –e <package\_name>
    - Package validation: rpm packages can be validated by rpm –Va
  + **YUM:**
* **User Management:**

**LINUX COMMANDS:**

* **Uptime:** tells how long the system has been up since the last boot, how many users are currently logged in and how much load the system is currently experiencing.
* **Jobs:** These commands can be used to list the jobs that are being tracked by the shell.
* **Fg :** this command can be used to bring a job back in foreground.  
  syntax: **fg <job\_number>**
* **Bg:** this command can be used to start a stopped job in the background.  
  Syntax: **bg <jn\_number>  
  note:** any job can be started in the background by appending & at the end of the job execution.  
  e.g. **echo “Adnan” &**
* **Printenv:** command prints the list of all the environment variables.  
  to see a particular environment use: **printenv <VAR\_NAME>**
* **Unset:** An environment variable can be unset by using unset command.  
  **unset <var\_name>**
* **Export:** makes the environment variable available for user to other programs running out of the current shell.  
  usage: export <var\_name>
* **Ls : list files command**common options : ls –l , -r , t ,-a , -R (recursively list contents of the subdirectories)   
  ls A\* (lists all files starting with “A”)
* **Change Ownership**: this is achieved by using the command chown, this can only be done by the root user.  
  chown –R <user> <filename/directory\_name>  
  chown –R <user.groupname> <filename/directory\_name>
* **Change Group:** the group permissions of a file or directory can be changed by using command  
  chgrp –R <group\_name> <file\_name>
* **Change Permissions**: chmod –R 700 <file\_name>