2015

LPI-Linux Essentials

Study Sheet

Table of Contents

Introduction

This study sheet is intended to help you do a targeted review of key topics that are very likely to be covered during your Linux Essentials LPI exam. This is not intended to replace any quizzes, labs or lectures here at Linux Academy, but can be used as a final review just prior to your exam.

Study Sheet Conventions

This follows the LPI syllabus for the Linux Essentials Exam. Each section contains the full LPIC designation, section title and description and weighting you can expect for these topics during the exam.

There are more topics that could potentially be covered than are listed in this review material. However, the items on this study sheet are the most common topics tested in each section based on community feedback and experience. Please do take the time to review all the topics from the formal syllabus available at http://www.lpi.org.

Topic 1: The Linux Community and a Career in Open Source (weight: 7)

1.1 Linux Evolution and Popular Operating Systems

Weight: 2

Description: Knowledge of Linux development and major distributions

- A Linux Introductory
  - Linux was created by Linus Torvalds in 1991
  - It is similar but is not the same as the first open source minimal release of Minix
  - Linus created the first Linux Kernel, BASH, GCC, and update utility – the first Linux release.
- Open Source Philosophy
  - The idea that Linux is Open Source means that its source code of the kernel and programs is freely available to anyone to download, modify, and redistribute.
  - The Free Software Foundation (FSF) started with this concept in what made Linux popular in its early days.
  - Example of this: Unix, Windows and other large Operating Systems are developed in house and the source code and documentation is very closed to the outside world. You cannot modify it or redistribute. Open Source, however, is open development and the source code can be modified and redistributed.
- Distributions
  - Linux has several releases of Linux "flavors"- different tools, applications – known as distributions
    - Linux Kernel

        - A Linux kernel is at the core of every Linux OS
    - Core Unix Tools
        - GNU tool set, X-Windows GUI, other tools for critical disk and system resource management
    - Supplemental Software
        - Typically distributions have add on software for desktop environments, productivity tools
    - Startup Scripts
        - Traditionally Linux distributions use start-up scripts to start major operating system functions
        - This is primarily replaced by system services
    - Installers
        - Package managers and utilities used to install software

- A few common Linux Distributions:
    - Centos
    - Debian
    - Fedora
    - Gentoo
    - OpenSuse
    - Red Hat
    - Slackware
    - SUSE
    - Ubuntu
    - Arch
    - Mint
- Embedded Systems
    - Android
    - Raspberry Pie
    - Cable boxes
    - smart TV's
    - smart phones

1.2 Major Open Source Applications

Weight: 2

Description: Awareness of major applications as well as their uses and development

- Desktop Applications
    - OpenOffice.org
    - Libre Office
    - Audacity
    - ImageMagick

    - Blender
    - Thunderbird
    - Firefox
    - GIMP
    - X Windows GUI desktop environments
        - KDE
        - GNOME
        - LXDE
        - Unity
        - XFCE
- Server Applications
    - Apache HTTPD – The most popular and widely used web software used today
    - NGINX – Widely used web server with built in load balancing
    - MySQL – Widely used open source database server
    - NFS
    - Samba
- Common TCP ports:

P#	Service	Server Applications
22	SSH	OpenSSH
23	Telnet	TelnetD
25	SMTP	Postfix, Sendmail
53	DNS	Bind
67	BOOTP, DHCP	dnsmasq, dhcpd
80	HTTP	Apache, NGINX
443	HTTPS	Apache, NGINX

- Mobile Applications
    - Great examples are applications that run on android (embedded Linux) which runs on mobile games etc.
        - These could be “weather apps, fitness apps, mobile games” etc.
        - These mobile apps are usually installed and managed through a store front application
- Development Languages
    - Cathedral Model
        - Organized method of development
        - Each programmer is assigned a portion of the project
        - Source Code is closed DURING development then released openly AFTER release
    - Bazaar Model
        - Less organized form of development

        - The idea of using more developers with openness where the hope is to give better results at the end project
        - Chaotic way of development
        - This is the model Linus Torvalds used when developing the first Linux Kernel
    - Common Development Languages
        - C
        - Java
        - Perl
        - Python
        - PHP
- Types of languages
    - Compiled Languages
        - To compile or (convert) a program written from its source code to the machine code known as binary
    - Interpreted Languages
        - Writing in human form language type of code
        - Line-By-Line code that is one line at a time and not from the complete compiled application
    - Assembly Language
        - Very low-level code like binary (1's and 0's)
- Package Management Tools and Repositories
    - Visual tools and command line tools
    - dpkg, apt-get, rpm, yum

1.3 Understanding Open Source Software and Licensing

Weight: 1

Description: Open communities and licensing Open Source Software for business

- Licensing
    - Software is a type of intellectual property.
    - A copyright is a legalized right to copy something.
    - Berne Convention is an international treaty that requires countries to recognize other countries copyrights.
    - Patents are the very idea of a copyrighted work.
    - Trademarks - The essential function of a trademark is to exclusively identify the commercial source of origin of products or services, so a trademark, properly called, indicates source or servers as a badge of origin. In other words, trademarks serve to identify a particular business as the source of goods or services. The use of a trademark in this way is known as trademark use.

- Commercial Software is software that would be developed with the intent to then sell that software as a profit.
- Share Software is similar to commercial software except in the copyright and legal perspective in that you can typically download this software and, based on an honor system, you can give payment to the author of that software later on.
- Freeware is like shareware but is always available for free.

- Free Software Foundation (FSF), Open Source Initiative (OSI)
  - The General Public License (GPL), which is the license used by the Linux Kernel, grants you the right to redistribute the software, including both the source code and its binaries.
  - Open Source licenses typically grant you additional rights to software
  - In 1985 Richard Stallman founded the Free Software Foundation (FSF)
  - The phrase and idea is that it is "free as in speech, not free as in beer"
  - 3 Freedoms that the FSF states:
    - Freedom to use the Software for any purpose
    - Freedom to examine the source code and modify it as you see fit
    - Freedom to redistribute the software
    - Freedom to redistribute your modified software
  - FSF is ok with selling software for profit, but given the other freedoms typically this software type is free
  - FSF and GPL often known as GNU GPL
  - 2 versions of this GPL:
    - Version 2 (GPLv2)
      - Release 1991
      - The Linux Kernel itself is released under GPLv2
        - This is important because it means that the Linux Kernel can still be used on what is otherwise a closed source hardware device such as TiVo and Android devices for example
        - These devices actually use a restrictive boot process to prevent unauthorized kernels to boot to them – this is something that the GPLv3 would not allow
      - Version 3 (GPLv3)
        - Intention was to close what is though of loopholes in GPLv2
        - Most new software under FSF are being released under GPLv3
    - another GPL known as the lesser GPL or simply the LGPL
      - This is applied to the libraries or Code libraries
    - (FDL) Free Documentation License

- The Open Source Initiative (OSI)
  - Founded in 1998 by both Bruce Perens and Eric Raymond
  - Typically most software falls under the OSI rather than the FSF
  - Open Source is a development method for software that harnesses the power of distributed peer review and a very transparent process.
    - The promise in this method is that the quality is higher, reliability is higher, more flexible, lower cost, and an end to vendor lock-in
  - These are the 10 principles that define the OSI; the first three of these are the most important:
    - Permission to derive works
    - Respect for source code integrity
    - No Discrimination against persons or groups
    - No Discrimination against fields of endeavor
    - Automatic license distribution
    - Lack of restrictions on other software
    - Technology Neutrality
- Creative Commons
  - Creative Commons was founded by Lawrence Lessig
  - The creative commons license is one of several public licenses that enable the free distribution of works that are otherwise copyrighted
- Further License types:
  - BSD
  - MIT
  - Apache
  - Artistic
  - NPL and MPL
- Open Source Business Models
  - How does business make profit or exist using free software platforms?
  - Service and Support
  - Dual Licensing – two versions of a product; one that is open source and another with features that are not available on the open source license
  - Multiple Products – While the company might have one open source product, other products in the line could have paid licensing or even other product types like manuals
  - Open Source Drivers – A special case of the preceding one is hardware vendors, who make money by selling hardware. They might opt to release drivers, or perhaps even hardware-specific applications, as open source as way to promote their hardware. Some hardware vendors are reluctant to release open source drivers for their hardware because in doing so it actually reveals information about their hardware which some vendors and reluctant to share.

        - Bounties – users can drive open source creation by offering to pay for new software or new features in existing software
        - Donations – A lot of open source projects accept donations that help fund their development.
    - Outside if these business methods a lot of open source projects come out of the academics, non-for-profit organizations, governments, hobbyists and more.

1.4 ICT Skills and Working in Linux

Weight: 2

Description: Basic Information and Communication Technology (ICT) skills and working in Linux

- Desktop Skills
    - Opening up Programs – You can open programs using the following placements/methods
        - Desktop Menus
        - Desktop Icons
        - Panels
        - Context Menus
        - Searching for Programs
        - Terminals
    - Getting to the Command Line
        - A Linux command line, or shell as it's more properly called, is a program like any other and must be launched in some way. Three methods are commonly used for this: starting a shell in a GUI window called a terminal, logging in to a text-mode console, and logging into the computer remotely using a text-mode login protocol such as telnet or SSH. The default shell in most Linux distributions is the Bourne-Again Shell (Bash), which is based on an older shell called the Bourne shell.
    - Industry uses of Linux, Cloud Computing and Virtualization

Topic 2: Finding Your Way on a Linux System (weight: 9)

2.1 Command Line Basics

Weight: **3**

Description: Basics of using the Linux command line.

- Shells:
    - sh – bourne shell – the sh shell was the earliest shell, developed for UNIX in the 1970's.

- bash – bourne again shell – the bash shell is an improved version of the sh shell and is one of the most used today. Typically most Linux Distro's default.
- csh – C shell – the csh shell was originally developed for BSD Unix. Similar Syntax to C programming.
- tsch – tsch – the tsch shell is an improved version of the C shell. Default for most FreeBSD systems.
- zsh – Z shell – The Z shell is an improved version of the bash shell.
- ksh – Korn Shell – The korn shell is an early shell that is similar to the C shell.

- Toggle between shells:
  - Ctrl-Alt F1 Ctrl-Alt F6 _ return to the GUI environment with Ctrl-Alt-F7
- Command Line Syntax
  - ls –a will show all files including hidden files
  - ls –d Directory Flag – will only list directories
  - ls –l long list of files and folders showing permissions string, owner, group, size, and even the creation date
  - ls –R Recursive options; it displays a directory's contents recursively

  - View environment Variables:
    - echo $PATH
    - env – also shows our environment variables
  - Basic Commands:
    - halt – this shuts our OS down, but important to know only root can do this
    - reboot – this shuts down our OS however also re-starts it back up, only root
    - init 0 – This will shut down the OS – only root can do this
    - init 6 – This will shut down the OS and then start it back up – only root
    - shutdown – can shut down OR reboot OS depending on flag used; example shutdown –r
    - exit – This will terminate our current running process, including the current SHELL session so if you open a terminal session within the Linux GUI and enter ext at the prompt the terminal session actually closes – same as if you are on a shell session only
    - su – substitute user OR Super User – This can switch our current user to a different user account. Change to the root user with su –
    - echo – echoes a line of text to our screen. example: echo $PATH
    - top – top is used to show a list of all the applications and processes currently running on our system
    - which – shows the full path to a shell command or a utility
    - whoami – will show the username of the current logged in user
    - netstat – shows the status of our network and its current counteractions, routing tables and more
    - route – Used to both view and manipulate our systems routing table

- ifconfig – This can manage our network card installed on our system and can be used to display and modify our NIC config settings – only root can use
- ip addr – newer versions of Linux such as Red Hat CentOS to see NIC settings
- uname – This can return information about our Linux system and has several options we can use known flags:
  - -s Displays the Linux kernel's name
  - -n Displays the system's hostname
  - -r Displays the Linux Kernel's release number
  - -v Displays Linux Kernel's version number
  - -m shows the system's hardware architecture
  - -p processor type
  - -i hardware platform
  - -o operating system
  - -a will display all the information above at once

- Command Line History
  - view the history of our command type: history
  - can change the config with HISTSIZE and HISTFILESIZE
    - example: export HISTFILESZIE=99999
  - Linux can do command line completion by starting command and then using the tab key

- SHELL configuration files:

/etc/bashrc	Non-Login Shell	system-wide functions and aliases
~/.bashrc	Non-Login Shell	user-specific functions and alias
/etc/profile	Login Shell	system-wide shell env config param
~/.bash_profile	Login Shell	user-specific shell preferences
~/.bash_login	Login Shell	user-specific shell preferences
~/.profile	Login Shell	user-specific shell preferences
~/.bash_logout	Login Shell	user-specific shell preferences

- Common Environment Variables
  - CPU
  - DISPLAY
  - ENV
  - EUID
  - HISTFILE
  - HISTSIZE
  - HOME
  - HOST and HOSTNAME
  - LOGNAME
  - MAIL

        - MANPATH
        - OLDPWD
        - OSTYPE
        - PATH
        - PSI
        - PWD
        - USER and USERNAME
    - You can view what a current Variable is with:
        - # echo $PATH
        - #echo $HOME
    - You can see current Environment Variables:
        - env –list variables
        - env | more will allow you to page through the output of these variables
        - set – will show all the variables HOWEVER in Alphabetical order
    - Editing Environment Variables
        - View path:
            - echo $PATH (will return)
                - /usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/user/.local/bin:/home/user/bin
        - We can add a folder to this path by adding to our $PATH
            - PATH=$PATH:/var/opt/
        - Change the path globally in the /etc/bashrc file

- Globbing
    - the process of using wildcard characters for expanding non-specific file names.
        - an asterisk (*) matches any characters
        - $ ls *.jpg #list all JPEG files
        - $ ls ?..jpg #list JPEG files with 1 char name (eg a.jpg, 1.jpg)
        - [] brackets bracketed value can match any character within the set of []
        - $ rm [A-Z]*.jph #This Removes JPEG files that start with a capital letter A through Z.
    - Quoting
        - There are three acting mechanisms or types: escape character, single quotes, and double quotes.
            - (\) a non-quoted backslash is the escape character
            - <newline>
            - “ ” Enclosing your characters in a double quotes preserves the literal value of ALL characters within those double quotes EXCEPT the ($,’,\. and if the history expansion is enabled. (!)
            - ‘ ‘ Enclosing your characters within single quotes preserves the literal value of each character within the quotes

        - A single quote may not occur between single quotes, even when preceded by a backslash (\)
    - The characters: $ and ‘ retain their special meaning within double quotes.
    - The backslash retains its special meaning only when followed by one of the following characters: $,’,”,\, or <newline>
    - A double quote may be quoted within double quotes by preceding it with a backslash.
    - If enabled, the history expansion will be performed UNLESS a ! appearing in double quotes is escaped using the backslash. The backslash pre-ceding the ! is not removed.
    - Special parameters * and @ have special meaning when in double quotes.

2.2 Using the Command Line to Get Help

Weight: **2**

Description: Running help commands and navigation of the various help systems.

- MAN PAGES – allows us to read manual documentation for Linux commands and applications
    - example:
    - man cp
    - man rm
    - man vim
    - Manual Sections:
        - Section 1: Executable programs and shell commands
        - Section 2: System calls provided by the kernel
        - Section 3: Library calls provided by program’s libraries
        - Section 4: Device files (usually stored in /dev)
        - Section 5: File Formats
        - Section 6: Games
        - Section 7: Miscellaneous (macro packages, conventions, and so on)
        - Section 8: System administration commands (programs run mostly or exclusively by root)
        - Section 9: Kernel routines
    - man –k #allows us to search through all the man pages for keywords
    - apropos #allows us to search through all the man pages for keywords
    - info
    - whatis
    - whereis
    - We can use less to read the /usr/share/Doc documentations files
- INFO PAGES

- newer than man but offers help and information about utilities and programs like man

2.3 Using Directories and Listing Files

Weight: 2

Description: Navigation of home and system directories and listing files in various locations.

- The Linux File System
  - Role and purpose of the Linux File System:
    - Data is organized and easily located
    - Data can be saved in a persistent manner
    - Data integrity is preserved
    - Data can be quickly retrieved for a use at a later point in time
  - (FHS) Filesystem Hierarchy Standard
    - very top is the /root directory and the FHS makes sure that on all distros that things like /usr, the /home and other file system directories have a consistency between the distros.
    - **/etc** contains text-based configuration files used by the system as well as services running on the system. We can edit these files with a text editor and then customize how linux behaves in different manners. Here are some important ones that reside in the /etc folder:
      - /etc/aliases Contains a table used to redirect ail to local users
      - /etc/exports Configured file systems to be exported to remove NFS clients
      - /etc/fstab lists the partitions and file systems that will be automatically mounted when we boot our linux system
      - /etc/ftpusers controls user's access to FTP service running on a linux system
      - /etc/group contains local group definitions
      - /etc/grub.conf contains configuration parameters for the init process
      - /etc/hosts this contains a list of hostname to IP address mappings that we can use to resolve certain hostnames
      - /etc/inittab contains configuration parameters for the init process
      - /etc/init.d this is a subdirectory that contains startup scripts for services installed on the system. On a RedHat or Centos system, these are located at /etc/rc.d/init.d/
      - /etc/passwd this is our linux system's user accounts file
      - /etc/shadow this contains encrypted password for our user accounts
      - /etc/resolve.conf This is where we specify what DNS server and domain suffix that our system is going to use
      - /etc/X11/ has the X windows configuration files

- **/home** directory that contains the subdirectories that serve as home directories for our user accounts that live on our linux system
- **/lib** directory contains code libraries used by programs that live in /bin and /sbin
- **/media** directory is used by some linux distros like OpenSUSE and Fedora to mount internal devices, including optical drives and USB drives
- **/mnt** directory is used by some linux distros to mount external drives like cd drives, dvd drives, usb drives and more
- **/opt** this directory contains files for some programs you install on the system
- **/proc** this directory is different than the others in this list. The /proc doesn't actually exist in the file system… instead, it's a pseudo-file system that is dynamically created whenever it is accessed. It's used to access process and other system information from the linux kernel. Within the /proc, there are several subdirectories and each of these subdirectories is identified with a number and not a name. These numbers actually correspond to Process ID numbers or (PID) number of the associated software process
- **/root** the /root directory is the root user's home directory - the root's home dir always lives outside the rest of the OS's user account's home dirs
- **/sbin** this directory contains important system management and administration files, such as fdisk, fsch, ifconfig, ifconfig, init, mkfs, shutdown, and halt.
- **/srv** this directory contains subdirectories where services running on the system (like httpd for apache or even ftpd) actually save their files
- **/sys** contains information about the hardware in your system
- **/tmp** directory contains temporary files created by you or by the system
- **/usr** contains application files; most of the application files used on your system are stored in a subdirectory of /usr. These subdirectories include the following:
    - /usr/bin Most of your executable programs
    - /usr/lib Library files
    - /usr/lib64 64-bit versions of your library files
    - /usr/local locally installed software that you created yourself (used to prevent it from being overwritten during a system update)
    - /usr/sbin system administration programs
    - /usr/share This is where Documentation and man page files reside
- **/var** this directory contains our linux variable data, and linux log files

- Linux DISK File Systems
    - ext2
    - ext3
    - Reiser
    - XFS
    - ext4

2.4 Creating, Moving, and Deleting Files

Weight: 2

Description: Create, move and delete files and directories under the home directory.

- touch – allows us to create a file
  - touch -c #will tell touch to NOT create a new file if one already exists
  - touch -d string or –date=string #set the date of a file that is represented by the string (example: touch -d "February 1 2015" file.txt)
- Copy Files
  - cp
    - cp file.txt /tmp/file.txt
    - cp file.txt newfilename.txt
    - The cp command has many different options that we can pass along with it that can do some handy stuff.
      For example:
    - the -f flag will force the system to overwrite any existing files without prompting us
    - the -i flag is the interactive option in which cp will always ask you before overwriting any existing files that might already exist
    - the -p flag is for preserving original file's owners and users so that whatever user you are using the cp command to copy files it will keep the original files ownership in tact
    - the -R is for recursive options. SO, for example, if we were copying a folder that also had subdirectories then cp would copy ALL of those directories recursively.
    - We can also use the -a flag to archive - it works the same way recursively as the -R does but it will preserve the ownership of the original files being copied as well as links and symbolic links. Again this is good for backing up or archival purposes.
    - the -u flag is the update option which will tell cp to copy the file only if the original is newer than the target OR if the target does not yet exist
- Moving and Renaming Files
  - mv
    - mv file.txt ~/fodler1/ #This moves the file file.txt to the folder folder1
  - rm
    - rm file1.txt #Deletes the file file1.txt
- Create a Directory
  - mkdir #Create a Directory
    - mkdir newdir #Creates a new directory called newdir
  - rmdir #Remove a Directory
    - rmdir newdir # Removes directory newdir
- Linux is a CASE SENSITIVE File System
  - The file File.txt is a different file than file.txt
  - Commands are also case sensitive
    - mv and not MV or Mv
- Hidden files and directories
  - You can make a file hidden by placing a period (.) in front of the file name

        - example: .file.txt # is hidden
    - You can also make directories hidden by placing a period (.) in front of the folder's filename
        - example: .folder1 # this folder would be hidden
    - We can see hidden files/folders with the ls command and flag –a
        - example:
            - # ls -a
              .file
              .fodler1
    - Absolute and relative paths
        - Absolute:
            - Absolute paths: when we specify the complete and full path from the /; for example:
                - cd /var/log NOT cd /log
            - Relative paths:
                - If you are already in /home/Stephen and there are subfolders such as /doc and /folder2
                    - Then you can just cd /doc and cd /folder2
                    - Again the relative in the above would be: cd /home/Stephen/doc and cd /home/Stephen/folder2
            - cd . # means to cd in to same directory
            - cd .. # means to change directory up a level

Topic 3: The Power of the Command Line (weight: 9)

3.1 Archiving files in the user home directory.

Weight: 2

Description: Archiving files in the user home directory

- Backup Utilities
    - tar
        - Send backup jobs to media such as tape drives, other disk drives and so forth
        - If we wanted to create a backup of /home called backup.tar to an external USB drive we could do:
            - tar -cvf /media/usb/backup.tar /home
        - -c create a new archive
        - -v verbose mode
        - -f specify tar file
        - To unpack this tar file:

        - tar-xvf backup.tar
    - -x extract from an archive
    - -f specify the file
- Compress these tar files using:
    - gzip, gunzip, bzip, and bzip2
    - gzip and gunzip file extensions:
        - .gz
    - bzip2 extension
        - .bz2
    - Compress a file called file2.txt with GZIP:
        - gzip file2.txt
          file2.txt.gz
    - Decompress the .gz files:
        - gunzip file2.txt.gz
          or
          gzip -d file2.txt.gz
    - Compress a file called file2.txt with BZIP2:
        - bzip2 file2.txt
          file2.txt.bz2
    - Decompress the .bz2 file:
        - bunzip2 file2.txt.bz2

- dd
    - dd can be used to create an image file of an entire hard disk.
        - Syntax: dd if=device_file of=output_file
        - Example
            - dd if=/dev/sdb of=/mnt/backdrive/drivebackup

3.2 Searching and Extracting Data from Files

Weight: 3

Description: Search and extract data from files in the home directory

- less – reads a file
- head and tail
    - head – view the beginning of a file's contents (10 lines)
        - head –n 3 file.txt #this would show the first 3 lines of our txt file
    - tail – view the end of a file's contents (10 lines)
        - tail -n 3 file.txt #this would show the last 3 lines of the txt file
- find – locates files on our file system
    - find . # where we want to look, in this example the . is the current directory or / recursively through everything in the root dir
    - find . -type f #find only files, not directories, with the -f flag

        - find . type d #find only directories within the current directory
        - find . –name "file*" # look for naming pattern 'globbing' for anything with the name file*
    - grep
        - -v Selected lines are those not matching any of the specified patterns
        - -i Ignores case and matches any single character
        - [^] Matches any character not contained in the brackets; [^abc] matches any character other than a, b, or c.
        - ^ Matches the starting position of a line
        - $ Matches the ending position of a string or the position just before a string ending new line; it matches the ending position of any line
        - * Matches the preceding element 0 or more time; ab*c matches b zero or more times: IE ac, abc, abbbbc, abbbbbbbbbbc.
        - + Matches the preceding element 1 or more times.
    - sort
        - sort forward/reverse contents by alpha/numeric characters
    - cut
        - used to remove sections from each line of files
            - cut -c2-4 file2.txt (shows each results from range 2-4 each line)
            - cut -c2,4 file2.txt (then we get column 2 and column 4 using the comma)
    - wc - word count
        - wc file3.txt (gives us statistics on our file)
        - wc -w file3.txt (give us just the word count)
        - wc -c file3.txt (I just want the character count)

3.3 Turning Commands into a Script

Weight: 4

Description: Turning repetitive commands into simple scripts

- Linux Command line text editors
    - nano
        - nano filename # nano followed by the filename
        - ctrl + k deletes a whole line
        - ctrl + x to save
    - vi
        - vi filename # vi followed by the filename to edit file
        - (i) – insert mode
        - (esc) takes us to command mode
        - :wq! – this will save our file and close the vi text editor
        - :q! – quotes no changes or anything is saved
        - u – undo last change

- w! – saves and closes
- q! – closes vi

Topic 4: The Linux Operating System (weight: 8)

4.1 Choosing an Operating System

Weight: 1

Description: Knowledge of major operating systems and Linux distributions

- Difference between Windows, MAC, and Linux
- Windows
  - Uses proprietary apps
  - Needs for Active Directory
  - Needs for MS SQL specific app back-ends
- MAC
  - Has their own Hardware and Software
  - Tight integration within it's own ecosystem
  - It is extremely difficult to manage and lock-down security on Macs within the enterprise.
- Linux
  - Desktop is free and personal
  - Extreme flexibility within the server room
  - License cost is usually free
  - Open Source applications which you can customize for your organization's needs
- GUI and CLI?
  - All of the OS's have both GUI and CLI's
  - Linux Server can be run on CLI only and gui not installed with the perhaps un needed resource overhead.
  - OSX/MAC has UNIX under the hood so we could also manage via CLI.
  - Windows can have both GUI and CLI. Have to use PowerShell to manage certain task via cli.

4.2 Understanding Computer Hardware

Weight: 2

Description: Familiarity with the components that go into building desktop and server computers

- Power Supplies – Convert AC current to DC current that our PC components need
- Motherboard – This interconnects all of our PC's devices
- CPU – Performs arithmetic and logical functions
- DRAM – fast but not persistent RAM

- SDRAM – Used for memory on the PC:
  - DDR
  - DDR2
  - DDR3
- PCI bus – typically 32 bits wide and runs at a clock speed of 33mhz
- PCI Express – provides serial point-to-point connection to the bus
- Interrupts are alerts that the CPU uses that a device needs attention.
- PnP – Plug and play devices allow devices to be automatically configured with system resources when the PC is booted
- Hard disks – aluminum platters are used to store data on traditional hard drives
  - Heads, Cylinders, and Sectors per track
- PATA (IDE) hard drives have a disk controller that integrates into the drive itself
  - Uses Master and slave drives
  - Can only have one drive act as master
- Optical drives use pits and lands to represent binary 0's and 1's
- USB allows devices to connect up to 127 external devices
  - USB devices are self-configuring and hot-swappable
- Video interfaces used to give our PC a display
  - Video Graphics Array (VGA)
  - DVI and HDMI (digital video interface) HDMI is also digital but also carries the audio signal not just video.
  - Linux uses two ways to load a driver
    - loaded as a kernel module after the operating system has started
    - compiled within the actual kernel itself

4.3 Where Data is Stored

Weight: 3

Description: Where various types of information are stored on a Linux system

- kernel – is core to the Linux operating system.
- The kernel interfaces with the hardware on the computer to the software layer.
  - Uses a hierarchy set of processes with the start of the process tree know as init.
- Processes
  - Process ID or PID
    - every single process has an associated PID number
  - top and ps
  - Using ps: Example
    - ps will return priority and CPU information and will sort it by the PID number.
    - ps -u Stephen –forest
      PID TTY TIME CMD

19570 ? 00:00:00 sshd
The above would show all of the processes owned by the user Stephen with the -u flag and the --forest option displaying the parent and child relationships
        - Use:
            - ps ax | grep vim
              Above example to search for processes with the name vim
    - top
        - The top command will show us system summary info about CPU utilization as well as memory.
    - free
        - You can use free to display amount of free and used memory in the system.

- Linux System Logs
    - Linux stores most of its logs in the /var directory
    - Common Logs:
        - boot.log: logs services that are started late within the boot process such as startup scripts and SysV
        - cron: processes that are run at regular intervals via the cron daemon – cron is a Linux scheduling service that you can use to schedule and kick off scripts.
        - messages or syslog file: these are the general purpose log files that contain messages from many of the linux daemons that don't have there own dedicated log files
    - Kernel Ring Buffer:
        - This is like a log file however this is stored in memory not the disk. We can view the kernel ring buffer with the dmesg tool and grep:
            - dmesg | grep console
            - dmesg | grep secure
            - dmesg | grep user

- /lib
    - dynamic libraries
- /var/lib

4.4 Your Computer on the Network

Weight: 2

Description: Querying vital networking configuration and determining the basic requirements for a computer on a Local Area Network (LAN)

- Protocol – Is a common networking language that must be configured for network hosts to communicate
- The Internet Protocol (IP) – Works in conjunction with the TCP or the UDP to fragment, transmit, defragment, and resequence network data
- TCP – Core protocol to the internet protocol suite; provides reliable, ordered, and error-checked delivery
- UDP – User datagram protocol is a simple connectionless transmission protocol that does not guarantee delivery
- ICMP – Internet control message protocol is used to test and verify network communications between hosts
- The OSI Reference model is layers that break down the overall compunction process into specific terms:
- Physical
- Data Link
- Network
- Transport
- Session
- Presentation
- Application
- PORT – is a logical connection provided by TCP and UDP for the upper layers – see common ports also in this study guide
- Ports are lumped in to:
    - well-known ports
    - registered ports
    - dynamic ports
- Each host on your network must have a unique IP address that is assigned to it as well as the correct subnet mask
- Subnet mask – defines how much of a given host's IP address is the network address and how much is the IP address.
- IP addresses are categorized into the following classes and default subnet mask:
    - Class A – 255.0.0.0
    - Class B – 255.255.0.0
    - Class C – 255.255.255.0
- DNS servers – need resolve IP addresses to domain names
- Public subnets – Must be globally unique and are registered IP addresses
- Private subnets – are networks that are non-routable globally; they are not registered IP addresses

- NAT – can be used to hide out private subnet and IP addresses and are behind one or more publically unique globally accessed IP address
- You can assigne an IP address to a Linux interface:
  - ifconfig *interface ip_address* netmask *subnet_mask* broadcast *broadcast_address*
- We specify our DNS servers in our Linux file at /etc/resolv.conf file
- ifdown is used to bring a Linux interface down
- ifup is used to bring a Linux interface up
- dhclient *interface* is used to dynamically assign an address to an interface
- ping – is used to test connectivity between host
- netstat – is used to view a network interface information using –a, -I, -r, -l
- traceroute – used to trace the route that your packets follow to reach a remote system
- example: traceroute 4.4.4.4 (traceroutes your location to googles DNS)

Topic 5: Security and File Permissions (weight: 7)

5.1 Basic Security and Identifying User Types

Weight: 2

Description: Various types of users on a Linux system

- Set owner of a file or folder:
  - chown <user.group> <file/folder name>
  - chown –R <user.group><file/folder name> Sets recursively to subfolders
- Set permissions on a file/folder
  - chmod 777 <file/folder name> full rights
  - chmod 775 <file/folder name> full rights for user/group but no write/execute for “other”
- Add –R for recursive:
  - chmod –R 775 <file/folder name> applies to subfolders.
  - UserGroupOther
  - -RWX RWX RWX
  - R = Read
  - W = Write
  - X = execute
  - - is for file and d is for directory
- File permissions using bits:
  - Read 4
  - Write 2
  - Execute 1
  - (added up you get 7 therefore 777 is read, write, execute)

        - chmod –R 755 /var/www/ and subfolders would have -rwxrwxrw-x or full rights except for the "Others" group which in this case does not have write permissions.
    - Sticky Bits:
        - Example: 1777- the first bit is for the sticky bit. If I say 1777 on a folder then the 1 on the first of the 4 digits or bits is the sticky bit. If at the end of a permissions string we see the letter 't", this indicated that the folder has sticky bit permission in which only the creators of the files/folders can delete there own files/folders, even if the folder has 777 permissions.

5.2 Creating Users and Groups

Weight: 2

Description: Creating users and groups on a Linux system

- within the /etc directory you have passwd and the shadow file
    - /etc/passwd: holds our user account
    - /etc/shadow: holds our passwords
- When we add a user to our system, it gets placed in the /etc/passwd:
    - user:x:1001:1001::/home/user:/bin/bash
    - The above says that user is the username, the :x: indicates that the password is encrypted and stored in the /etc/shadow file, and the first :1001: is the UID. The second :1001: is the primary Group ID (GID). The Home directory is in the /home/user directory and the default shell for this user shows that it is bash with /bin/bash.
- The /etc/group file holds all of the group information as well as the user's belongings to each of those groups:
    - user:x:1001
    - The above entry indicates that the name of the group is user, :x: indicates that the password for the group is encrypted and not really used with groups. The 1001 is the GUI or Group ID. Other users that would be in this group would follow the last ":"
- Adding Users:
    - useradd user
        - -d sets home directory for the user (if other than default which is: /home/"username") useradd –d /home/user user
        - -m creates the home directory useradd –d/home/user –m user
- Removing Users:
    - userdel user
- Set a user's password: As root:
    - passwd user
- Adding users to groups:

- useradd –accounting user
- The above command adds the user (user) to the group accounting

- Add a Primary group to a user:
  - -G
- Modifying users:
  - usermod –accounting user
  - usermod –d/home/differntfoldername user
- Options:
  - -d change user's home directory
  - -m creates directory
  - -s used to change the default shell
  - -r remove home directory when deleting a user
  - "Passwd" will change the user's password

5.3 Managing File's Permissions and Ownership

Weight: 2

Description: Understanding and manipulating file permissions and ownership settings

- Permissions:

-=file
d=directory
r=read (4 bits)
w=write (2 bits)
x=execute (1 bit)
First 3 permissions bits are for user.
Second 3 permission bits are for groups.
Third permission bits are for others.
Directory
drwxrwxr-x. → Users/Groups have read, write, execute. Other has read and execute.
File
-rwxrwxrwx. → Users/Groups AND others have read, write, execute.

- Note – when a directory has x (execute) it means we can change directories in to it.

- Octal Notation

rwx	rwx	rwx
421	421	421
7	7	7

R=4
W=w
X=1
Total = 7

- Octal Notation adding:

-rw-r--r--
420 4 4
6 4 4 = 644

- Setting Permission with octal

chmod 644 filename
or for example
chmod ugo+w (this adds write permissions to the user, group and other)

- You must either be root or the owner of a file/directory to change its permissions or change its owner
- chown username.groupname filename
- chown stephen.accounting filename.txt
- Removing permissions:
- chmod o-r filename.txt (we removed the read permission for the others)
- chmod g-w filename.txt (we removed the write permission from the groups)
- chmod u-r (we removed the read permission from the user)
- Adding permissions:
- chmod o+r (we added the read permission)

5.4 Special Directories and Files

Weight: 1

Description: Special directories and files on a Linux system including special permissions

- Symbolic Links
  - ln -s (creates a shortcut or LINK to a filename)
  - Symbolic links have their own inodes
    - syntax:
      - ln –s filewearelinkingto.txt nameofthesymblink.txt
      - ls –l
        nameofthesymlink.txt → filewearelinkto.txt
- Hard Links
  - Using the ln command without any options creates a hard link
  - Hard links point to another inode
- Special Directories
  - /var – contains files that change often such as mail, logs, etc
  - /var/tmp – contains temporary files that do NOT get deleted on reboot
  - /tmp – contains temporary files that DO get deleted on reboot
- Sticky Bits
  - Sticky Bit: 1
  - We can use the sticky bit permission on folders in which we want to keep users from deleting other users files/folders even if the folder has open

permissions with 777. We add a sticky bit permission to this file and only the owners that created files can delete their own files and not others.
    - Add the sticky bit to a folder:
        - chmof o+t nameoffolder
- SUID
    - Set User ID
        - SUID: 4
        - SGID: 2
- Example using Special Permission with the extra digit at the beginning:
    - chmod 6554 file1
        - This means that the SUID(4) and the SGID(2) so the total of the first of the four digits would be 6. Also the owner and group have (4) and execute permissions of (1) for a total of (5) in the second and third digits. It furthermore says that the others are allowed to read (4) but they can not modify or run (so the last digit total is 4); so the total bits are 6554 for file1 shown above.