System Administrative Basics Chapter - 4

Presentation from Uplatz

Contact Us: https://training.uplatz.com/
Email: info@uplatz.com
Phone: +44 – 7836 212635

4. System Administrative Basics

- User management (who, whoami, groups, su, adduser, deluser, useradd, userdel, id, usermod, users)
- Time management (time, uptime)
- init levels along with shutdown (init, shutdown, halt, systemctl)
- Accessing administrator (root) privileges
- Package management (apt, apt-get, apt-cache, dpkg)
- Networking (hostname, ifconfig, ping, host, dig, nslookup, route, traceroute, tracepath, netstat, arp, ftp, sftp, scp, wget, telnet, ssh)
- File systems and devices

User Management

- Lists all the users logged on the system.

  $ who

- Tells what user I am logged as.

  $ whoami

- Tell which groups I belong to.

  $ groups

- Tell which groups <user> belongs to.

  $ groups <user>

User Management

- You do not have to log out to log on another user account!
- (Rare) Change to the hyde account but keeping the environment variable settings of the original user.

  $ su hyde
- (More frequent) Log on the jeky11 with exactly the same settings as this new user.

  $ su - jeky11
- When no argument is given it means the root user.

  $ su -

User Management

- Adding user or group to the system
- Using adduser

  $ adduser <user> <group>

  - Perl Script
  - Calls useradd command from the script
  - Creates user directory automatically
- Using useradd

  $ useradd <LOGIN>

  - Native binary (ELF) compiled with the system
  - User need to create login directory

User Management

- Removing user or group from the system
- Using deluser

  $ deluser <user> <group>

  - Perl Script
  - Calls userdel command from the script
  - Need to remove user or login directory manually, if no option is selected (as tested in Ubuntu System).
  - Removes login directory if option --remove-home is selected (as tested in Ubuntu System).
- Using userdel

  $ userdel <LOGIN>

  - Native binary (ELF) compiled with the system
  - Need to remove user or login directory manually (if no option is selected, as in Ubuntu System)
  - Deletes a user account and related files (with –r option, as tested in Ubuntu System)

User Management

- Prints real and effective user and group IDs

  $ id

- Modify user account

  $ usermod

- Print the user name of users currently logged into the current host

  $ users

Time Management

- Waits for 60 seconds (doesn't consume system resources).

  $ sleep 60

- Returns the current date. Useful in scripts to record when commands started or completed.

  $ date

- Count the time taken by a command or measuring elapsed time

  $ time find_expensive_housing --near

  <…command output…>

  real 0m2.304s (actual elapsed time)

  user 0m0.449s (CPU time running program code)

  sys 0m0.106s (CPU time running system calls)

  Real = user + sys + waiting

  Waiting = I/O waiting time + idle time (running other tasks)

Time Management

- Command uptime
  - Tells how long the system has been running

```
$ uptime
 21:58:03 up 42 min,  3 users,  load average: 1.23, 1.20, 1.21
$
$ uptime -p
up 41 minutes
$
$ uptime -s
2020-07-03 21:15:58
$
```

Linux Runlevels

Run Level	Mode	Action
0	Halt	Shuts down system
1	Single-User Mode	Does not configure network interfaces, start daemons, or allow non-root logins
2	Multi-User Mode	Does not configure network interfaces or start daemons.
3	Multi-User Mode with Networking	Starts the system normally.
6	Reboot	Reboots the system

$ init <Run-Level>

https://www.liquidweb.com/kb/linux-runlevels-explained/

Shutting Down

- Immediately halts the system.

  $ halt

  $ shutdown # systemctl halt -i

- Immediately reboots the system.

  $ reboot # systemctl reboot -i

- Also works on GNU/Linux to reboot
  - [Ctrl] [Alt] [Del]
- Embedded systems: You must use an implementation of init and can specify any key combination in /etc/inittab file.

Introduction to Systemctl

- systemctl is a controlling interface and inspection tool for the widely-adopted init system and service manager systemd.
- systemd initializes *user space* components that run after the Linux kernel has booted, as well as continuously maintaining those components throughout a system's lifecycle.
- These tasks are known as *units*, and each unit has a corresponding *unit file*.
- Units might concern mounting storage devices (.mount), configuring hardware (.device), sockets (.socket), or managing services (.service)

https://www.linode.com/docs/quick-answers/linux-essentials/introduction-to-systemctl/

VIEWING systemd INFORMATION

Command	Description
systemctl list-dependencies	Show a unit's dependencies
systemctl list-sockets	List sockets and what activates
systemctl list-jobs	View active systemd jobs
systemctl list-unit-files	See unit files and their states
systemctl list-units	Show if units are loaded/active

https://access.redhat.com/sites/default/files/attachments/12052018_systemd_6.pdf

WORKING WITH SERVICES

Command	Description
systemctl stop service	Stop a running service
systemctl start service	Start a service
systemctl restart service	Restart a running service
systemctl reload service	Reload all config files in service
systemctl daemon-reload	Must run to reload changed unit files
systemctl status service	See if service is running/enabled
systemctl --failed	Shows services that failed to run
systemctl reset-failed	Resets any units from failed state
systemctl enable service	Enable a service to start on boot
systemctl disable service	Disable service--won't start at boot
systemctl show service	Show properties of a service (or other unit)
systemctl edit service	Create snippit to drop in unit file
systemctl edit --full service	Edit entire unit file for service
systemctl -H host status network	Run any systemctl command remotely

CHANGING SYSTEM STATES

Command	Description
systemctl reboot	Reboot the system (reboot.target)
systemctl poweroff	Power off the system (poweroff.target)
systemctl emergency	Put in emergency mode (emergency.target)
systemctl default	Back to default target (multi-user.target)

VIEWING LOG MESSAGES

Command	Description
journalctl	Show all collected log messages
journalctl -u network.service	See network service messages
journalctl –f	Follow messages as they appear
journalctl –k	Show only kernel messages

Using Unit Files

Besides services, most systemd commands can work with these unit types: paths, slices, snapshots, sockets, swaps, targets, and timers

Beware of the dark side of root

- “root” user privileges are only needed for very specific tasks with security risks: mounting, creating device files, loading drivers, starting networking, changing file ownership, package upgrades…
- Even if you have the root password, your regular account should be sufficient for 99.9% of your tasks (unless you are a system administrator).
- In a training session, it is acceptable to use root. In real life, you may not even have access to this account, or put your systems and data at risk if you do.

Using the root account

In case you really want to use root…

- If you have the root password
  - su – # (**S**witch **U**ser)
- In modern distributions, the sudo command gives you access to some root privileges with your own user password.
  - Example: sudo mount /dev/hda4 /home

Software packages

- The distribution mechanism for software in GNU/Linux is different from the one in Windows
- Linux distributions provides a central and coherent way of installing, updating and removing applications and libraries: **packages.**
- Packages contains the application or library files, and associated meta-information, such as version and the dependencies.
  - On debian and Ubuntu → .deb
  - On Mandriva, Fedora, OpenSUSE → .rpm
- Packages are stored in **repositories,** usually on HTTP or FTP servers
- One should only use packages from official repositories of its distribution, unless strictly required.

Uplatz

Package Management

- Most modern Unix-like operating systems offer a centralized mechanism for finding and installing software.
- Software is usually distributed in the form of **packages**, kept in **repositories**.
- Working with packages is known as **package management**.
- Packages provide the basic components of an operating system, along with shared libraries, applications, services, and documentation.

https://www.digitalocean.com/community/tutorials/package-management-basics-apt-yum-dnf-pkg

- Most package systems are built around collections of package files.
- A package file is usually an archive which contains compiled binaries and other resources making up the software, along with installation scripts.
- Packages also contain valuable metadata, including their **dependencies**, a list of other packages required to install and run them.

Operating System	Format	Tool(s)
Debian	.deb	apt, apt-cache, apt-get, dpkg
Ubuntu	.deb	apt, apt-cache, apt-get, dpkg
CentOS	.rpm	yum
Fedora	.rpm	dnf
FreeBSD	Ports, .txz	make, pkg

Software Package Manager	Abbreviation
rpm	Red Hat Package Manager
yum	Yellowdog Updater, Modified
dnf	Dandified YUM (next-generation version of the Yellowdog Updater, Modified, a package manager for .rpm-based distributions)

To find the name of a package to install, the best is to use the search engine on http://packages.debian.org or on http://packages.ubuntu.com.

Packet Management Tools

- Command Line Interface
  - apt - Main Command-line package management tool
  - aptitude - Command-line and text-based interface for apt
- Graphical interfaces
  - Synaptic for GNOME (GNU Network Object Model Environment) a GTK-based GUI for APT package manager
  - Adept (Advanced Packaging Tool) for KDE (K Desktop Environment)
  - KPackage is a package management program for KDE Desktop Enviroment (Linux)
- Further details on package management: http://www.debian.org/doc/manuals/apt-howto/

Packet Management Tools

- The apt package provides command line tools for searching, managing, and querying information about packages, and access all features of the libapt-pkg library:
- apt - Advanced Package Tool - High-level command line interface for the package management system.
- apt-cache - cache manipulator - performs a variety of operations on APT's package cache.
- apt-cdrom - used to add a new CD-ROM to APT's list of available sources.
- apt-config - This program is used by the other apt utilities to provide a standard interface to the apt configuration settings.
- apt-get - command-line tool for handling packages
- apt-key - manage the list of keys used by apt to authenticate packages.

https://wiki.debian.org/PackageManagementTools

Managing software packages (apt-get)

Instructions for Debian based GNU/Linux systems (Debian, Ubuntu …) (apt-get)

- Package repositories are specified in /etc/apt/sources.list
- To update package repository lists:

  $ sudo apt-get update
- To upgrade installed packages:

  $ sudo apt-get upgrade
- To install a given package

  $ sudo apt-get install <package>
- To remove a given package

  $ sudo apt-get remove <package>
- To purge (removes any configuration files as well) a given package

  $ sudo apt-get purge <package> #or

  $ sudo apt-get remove --purge <package>
- To install/remove packages in an attempt to satisfy build dependencies for source package

  $ sudo apt-get build-dep <package>
- To install all available package updates:

  $ sudo apt-get dist-upgrade

Uplatz

Managing software packages (apt-cache)

Instructions for Ubuntu based GNU/Linux systems (apt-cache)

Note: Take example of any package for <package> say vim, bash etc

- Displays information about the packages listed on the command line
  $ apt-cache showpkg <package>
- Displays some statistics about the cache
  $ apt-cache stats
- Shows a short listing of every package in the cache. It is primarily for debugging.
  $ apt-cache dump
- Displays a summary of all unmet dependencies in the package cache.
  $ apt-cache unmet
- Displays the package records for the named packages or get information about a package.
  $ apt-cache show <package>
- Search on all available package lists:
  $ apt-cache search <keyword> # $ apt-cache search vim
- Shows a listing of each dependency a package has and all the possible other packages that can fulfill that dependency.
  $ apt-cache depends <package>
- Prints the name of each package APT knows.
  $ apt-cache pkgnames [prefix]
- To help debug issues relating to the preferences file.
  $ apt-cache policy [package]

Managing software packages (apt)

Instructions for Ubuntu based GNU/Linux systems (apt)

- apt provides a high-level command line interface for the package management system.
- The apt command is a powerful command-line tool, which works with Ubuntu's Advanced Packaging Tool (APT) performing such functions as installation of new software packages, upgrade of existing software packages, updating of the package list index, and even upgrading the entire Ubuntu system.
- It is intended as an end user interface and enables some options better suited for interactive usage by default compared to more specialized APT tools like apt-get and apt-cache.

- Update the Package Index
  - The APT package index is essentially a database of available packages from the repositories defined in the /etc/apt/sources.list file and in the /etc/apt/sources.list.d directory.
  - To update the local package index with the latest changes made in the repositories type the following:

  $ sudo apt update
- Upgrade Packages
  - Over time, updated versions of packages currently installed on your computer may become available from the package repositories (for example security updates).
  - To upgrade your system, first update your package index as outlined above, and then type

  $ sudo apt upgrade
- Install a Package

  $ sudo apt install <package>
- Remove a Package

  $ sudo apt remove <package>
- Search for the given regex term(s) in the list of available packages and display matches.

  $ apt search <package>
- Show information about the given package(s) including its dependencies, installation and download size, sources the package is available from, the description of the packages content and much more.

  $ apt show <package> # or with -a # Example # $ apt -a show vim
- Display a list of packages satisfying certain criteria.

  $ apt list # or # $ apt list <package> # Example # $ apt list vim

  $ apt list --installed # To check installed or available versions

Managing software packages (dpkg)

Instructions for Debian based GNU/Linux systems (Debian, Ubuntu …) (dpkg)

- dpkg is the software at the base of the package management system in the free operating system Debian and its numerous derivatives.
- dpkg is used to install, remove, and provide information about .deb packages.
- To install a .deb package

  $ dpkg -i filename.deb
- To list installed packages

  $ dpkg -l [optional pattern]
- To remove an installed package

  $ dpkg -r packagename

- dpkg-dev contains a series of development tools required to unpack, build and upload Debian source packages. These include:
  - **dpkg-architecture** is to set and determine the architecture for package building.
  - **dpkg-buildflags** it returns build flags to use during package build
  - **dpkg-buildpackage** is a control script that can be used to construct the package automatically.
  - **dpkg-checkbuilddeps** is to check build dependencies and conflicts.
  - **dpkg-deb -** Debian package archive (.deb) manipulation tool
  - **dpkg-distaddfile** adds a file input to debian/files.
  - **dpkg-divert** - override a package's version of a file
  - **dpkg-genchanges** reads the information from an unpacked Debian tree source that once constructed creates a control file (.changes).
  - **dpkg-gencontrol** reads the information from an unpacked Debian tree source and generates a binary package control package, creating an entry for this in Debian/files.
  - **dpkg-gensymbols** - Generates symbols files (shared library dependency information)
  - **dpkg-log-summary**

- **dpkg-maintscript -** Helper works around known dpkg limitations in maintainer scripts
- **dpkg-mergechangelogs -** 3-way merge of debian/changelog files
- **dpkg-name -** rename Debian packages to full package names
- **dpkg-parsechangelog** reads the changes file (changelog) of an unpacked Debian tree source and creates a conveniently prepared output with the information for those changes.
- **dpkg-query -** a tool to query the dpkg database
- **dpkg-scanpackages -** create Packages index files
- **dpkg-scansources** - create Sources index files
- **dpkg-shlibdeps** calculates the dependencies of runs with respect to libraries.
- **dpkg-source** packs and unpacks the source files of a Debian package.
- **dpkg-split** - Debian package archive split/join tool
- **dpkg-statoverride** - override ownership and mode of files
- **dpkg-trigger** - a package trigger utility
- **dpkg-vendor -** queries information about distribution vendors

Networking Basics

- A network connects two or more systems
- Each system on the network is addressed by a hostname or an IP address
  - Hostname looks like myserv (hostname) or myserv.rdom (hostname in another domain)
  - IP address looks like 192.168.0.2 (IPv4 Address)
- Every system has MAC (Media Access Control) address, which is unique address that is assigned by the manufacturer of network hardware (like ethernet card or wireless card)
- Networking utilities are used to communicate over the network

Networking Utilities

- Obtaining network information using utilities such as ping, finger, traceroute, tracepath, netstat, host, dig, route, nslookup, hostname
- Transferring files between a local host and a remote host or between two remote hosts using networking utilities such as ftp, tftp, scp
- Remote login to other Linux Systems using telnet, rlogin, ssh
- Remote login to access Linux Desktop using tools such as TeamViewer, RealVNC (Virtual Network Computing) etc

Basic Networking utilities

- Show or set the system's host name

  $ hostname

  ubuntu-ThinkPad-T420

- System administration utility for network interface configuration. It has features for configuring, controlling and querying network interface parameters.

  $ ifconfig [-v] [-a] [-s] [interface]

- Networking utility ping
  - Ping is a computer network administration software utility used to test the reachability of a host on an Internet Protocol network.

  $ ping google.com

  $ ping ubuntu-ThinkPad-T420

- Simple utility for performing Domain Name System lookups (host)

  $ host google.com

Network Setup

- Print details about all the network interfaces available on your system
  $ ifconfig –a
- Lists details about the eth0 interface
  $ ifconfig eth0
- Assigns the 192.168.0.100 IP address to eth0 (1 IP address per interface).
  $ ifconfig eth0 192.168.0.100
- Shuts down the eth0 interface (frees its IP address)
  $ ifconfig eth0 down
- Sets the default route for packets outside the local network. The gateway (here 192.168.0.1) is responsible for sending them to the next gateway until the final destination.
  $ route add default gw 192.168.0.1
- Lists the existing routes
  $ route –n
- Deletes the given route. Useful to redefine a new route.
  $ route del default or route del <IP>
- Your programs need to know what IP address corresponds to a given host name (such as kernel.org). Domain Name Servers (DNS) take care of this.
- You just have to specify the IP address of 1 or more DNS servers in your /etc/resolv.conf file:
  $ nameserver 217.19.192.132
  $ nameserver 212.27.32.177
  - The changes take effect immediately.

Option –n: Immediately displays ip addresses instead of trying to find their domain names.

Basic Networking utilities

- Domain Information Groper (dig)
  - A network administration **command**-line tool for querying Domain Name System (DNS) name servers. It is useful for verifying and troubleshooting DNS problems and also to perform DNS lookups and displays the answers that are returned from the name server that were queried

  $ dig
- Name server lookup (nslookup)
  - Tool used for querying the Domain Name System (DNS) to obtain domain name or IP address mapping, or other DNS records. The name "nslookup" means "name server lookup".

  $ nslookup google.com
- View and manipulate the IP routing table (route)

  $ route

Basic Networking utilities

- Computer network diagnostic command (traceroute)
  - Traceroute command is for displaying the route (path) and measuring transit delays of packets across an Internet Protocol (IP) network.

  $ traceroute google.com # For IPv4 addresses

  $ traceroute --mtu google.com # With MTU option

  $ traceroute6 # For IPv6 Addresses
- To trace path to destination discovering MTU (Maximum Transmission Unit) along the path.

  $ tracepath google.com

Basic Networking utilities

- Network Statistics (netstat)
  - Command line tool for monitoring network connections both incoming and outgoing as well as viewing routing table, interface statistics, multicast memberships etc

  $ netstat
- ARP (Address Resolution protocol) cache management (arp)
  - Displays and modifies entries in the ARP cache, which contains one or more tables that are used to store IP addresses and their resolved Ethernet or Token Ring physical addresses.

  $ arp –a

  $ arp hostname

Basic Networking utilities

- File Transfer Protocol (ftp)

  $ ftp [<hostname>]

  - You will be asked for username (a valid username on the target hostname) and password
  - Once you login, you can execute several commands (cd, get, put, mget, mput, bin, asc, help etc)
- Secure File Transfer Protocol (sftp)
- Secure Copy – Remote file copy (scp)

  $ scp [[user@]host1:]file1 ... [[user@]host2:]file2

Basic Networking utilities

- Retrieves content from web servers (wget).
  - It supports HTTP, HTTPS and FTP

  $ wget https://en.wikipedia.org/wiki/Wget

  $ wget google.com # No page is specified, index page is downloaded

Basic Networking utilities

- TELNET protocol based communication (telnet)
  - $ telnet [-l <username>] [<hostname>]
    - Telnet to hostname using username
- Remote login (rlogin)
  - $ rlogin [-l <username>] <hostname>
    - Allows you to login to a remote host
- Secure Shell Protocol (ssh) – Remote login
  - $ ssh [user@]hostname [command]
- User information lookup (finger)
  - $ finger [username | username@hostname]
    - Lookup users details such as when was the last login, what shell is used, if there is mail etc

Network Testing

- First, try to ping the IP address of your gateway. This will confirm that your network adapter works fine.
- Then, make sure you can ping the name server IP address, which will confirm that your gateway is configured properly.
- Finally, make sure you can ping any host using its name, which will confirm that the name server configuration is correct.

Device Names

- In Linux/Unix everything is a file.
- A device represents as a file. All device type file located at /dev location.
- So **sda** is a block device type special file.
- hd(x) - IDE
- sd(x) - SATA, SSD, SCSI, SAS

Note: where x is a variable. x represents the position of hard disk.

- if x = a for IDE disk means hda represents primary master disk
  x = b for IDE disk means hdb represents primary slave disk

Now in sd*

- if x = a for SATA/SSD/ISCI/SAS sda represents first disk
  x = b fo SATA/SSD/ISCI/SAS sdb represents second disk

Devices

Symbol	Abbreviation
IDE	Integrated Drive Electronics.
SATA	Serial AT Attachment
SSD	Solid-State Drive
SCSI	Small Computer System Interface
ISCSI	Internet Small Computer Systems Interface
SAS	Serial-attached SCSI

Creating file systems

- Formats your USB key (/dev/sda1: 1st partition raw data) in ext2 format

  $ mkfs.ext2 /dev/sda1

- Formats a disk image file in ext2 format. Option "-F": force. Execute even if not a real device file.

  $ mkfs.ext2 –F disk.img

- Formats your USB key back to FAT32 format.

  $ mkfs.vfat –v –F 32 /dev/sda1 (-v: verbose)

- Formats a disk image file in FAT32 format.

  $ mkfs.vfat –v –F 32 disk.img

- Blank disk images can be created as in the below example (64MB file):

  $ dd if=/dev/zero of=disk.img bs=1M count=64

Mounting devices

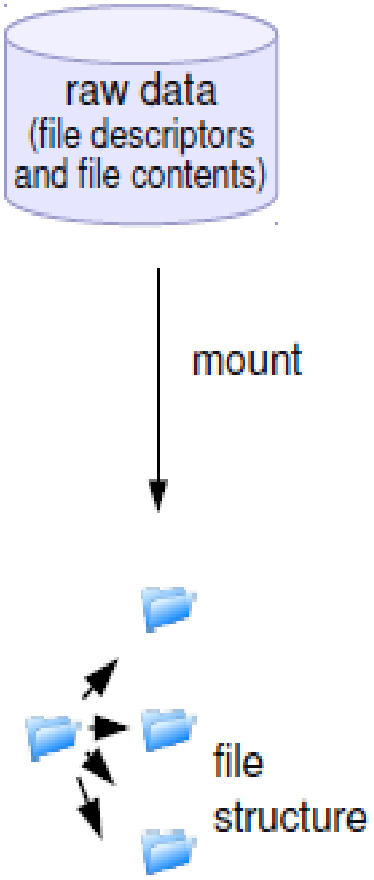
- To make filesystems on any device (internal or external storage) visible on your system, you have to mount them.
- The first time, create a mount point in your system: Example:

  $ mkdir /mnt/usbdisk
- Now, mount it:

  $ mount –t vfat /dev/sda1 /mnt/usbdisk
  - /dev/sda1: physical device
  - "-t": specifies the filesystem (format) type
  - (ext2, ext3, vfat, reiserfs, iso9660…)
- You can also mount a filesystem image stored in a regular file (loop devices)
  - Useful to develop filesystems for another machine
  - Useful to access the contents of an ISO cdrom image without having to burn it.
  - Useful to have a Linux filesystem inside a file in a Windows partition.

    $ cp /dev/sda1 usbkey.img

    $ mount –o loop –t vfat usbkey.img .mnt/usbdisk



Listing mounted file systems

- Just use the mount command with no argument:

```
$ mount # Sample output
 /dev/hda6 on / type ext3 (rw, noatime)
 none on /proc type proc (rw, noatime)
 none on /sys type sysfs (rw)
 none on /dev/pts type devpts (rw, gid=5, mode=620)
 usbfs on /proc/bus/usb type usbfs (rw)
 /dev/hda4 on /data type ext3 (rw, noatime)
 none on /dev/shm type tmpfs (rw)
 /dev/hda1 on /win type vfat (rw, uid=501, gid=501)
 none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
```

- View the file /proc/filesystems, which filesystems your kernel currently supports (ext3, ext2, ext4, squashfs, vfat, fuseblk)

Unmount the devices

- Commits all pending writes and unmounts the given device, which can then be removed in a safe way.

  $ umount /mnt/usbdisk

- To be able to unmount a device, you have to close all the open files in it:
  - Close applications opening data in the mounted partition.
  - Make sure that none of your shells have a working directory in this mount point.
  - You can run the lsof <mount point> command (list open files) to view which processes still have open files in the mounted partition.