# Linux Administration

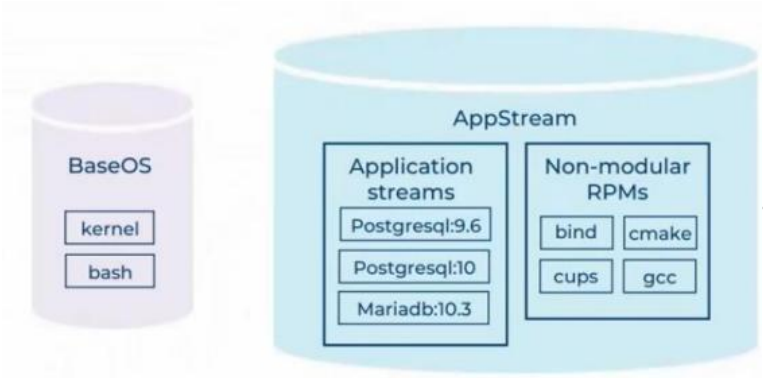

## Azza Khalel

khalelazza@gmail.com
https://www.linkedin.com/in/azzakhalel/

# Linux package management

- RPM (RedHat Package Manager)
  - RPM package files names consists of 4 elements
    - Name-version-release.architecture.rpm
    - To get your OS release
      - #cat /etc/redhat-release
    - To get your CPU arch
      - #uname –a


  - RPM used to install a package which already downloaded without any dependencies
  - To download a package
    - #wget <package_URL>
- YUM (Yellowdog Updater, Modified)
  - Its designed to be a better system for managing rpm-based packages
  - Its used to download and install any package with all its dependencies
  - It uses repository concept.
    - Starting from RHEL8, there are 2 main repos
      - BaseOS (all OS related packages)
      - AppStream (any other packages)
- Dnf (Dandified YUM)
  - is the next-generation version of the yum



# RPM

- rpm -i <pck_name> OR rpm --install <pck_name>
- rpm -ivh <pck_name> (install verbose installation progress hash ###)
- rpm --install --nodeps <pck_name> ➔ ignore dependency checks
- rpm --install --force <pck_name> ➔ force the installation, even if there are conflicts or problems.
- rpm -U <pck_name> ➔ upgrade the pck (if it's installed,update it and if not installed,install it then update)
- rpm -F <pck_name>➔ fresh the pck (if it's installed, update it and if not installed, no actions)
- rpm -e <pck_name> OR rpm --erase <pck_name> ➔ uninstall pck

# YUM

- Yum list ➔ print all installed packages
- Yum search <keyword> ➔ list all package which contain this keyword in the name or summary fields only
- Yum info <pck_name> ➔ print information about pck
- Yum install <pck_name> ➔ to install package and its dependencies
- Yum provides <dir_path> ➔ to know this dir is related to which package
- yum update <pck_name>
  - Checks for updates to package metadata (e.g., information about available packages, their versions, and dependencies) from configured repositories.
  - Does not install any new packages.
  - Updates the local package database to reflect the latest available versions.
- yum upgrade <pck_name>
  - Installs the latest versions of packages that are already installed on your system, based on the updated metadata obtained from #dnf update
- #yum remove <pck_name>
- yum localinstall <pck_name>➔ install a downloaded pck with it’s dependencies

# dnf

| Task: | Command: |
| --- | --- |
| List installed and available packages by name. | `dnf list [NAME-PATTERN]` |
| List installed and available groups. | `dnf group list` |
| Search for a package by keyword. | `dnf search KEYWORD` |
| Show details of a package. | `dnf info PACKAGENAME` |
| Install a package. | `dnf install PACKAGENAME` |
| Install a package group. | `dnf group install GROUPNAME` |
| Update all packages. | `dnf update` |
| Remove a package. | `dnf remove PACKAGENAME` |
| Display transaction history. | `dnf history` |

# Repositories

- #dnf  repolist ➔ to list all enabled repos
- #dnf repolist all ➔ to list all enabled and disabled packages
- /etc/yum.repo.d ➔
  - the dir which contain all .repo files which contain all repos
  - Each file contain one or more repos
  - Each section in this file represent a repo
- To enable a specific repo
  - #dnf config-manager --enable <repo_id>
  - #dnf config-manager --disable <repo_id>

# Own repo

- #mkdir /myrepo
- #cp /media/DVD_name/Packages /myrepo
- #chmod -R 755 /myrepo
- #createrepo /myrepo

- vi /etc/yum.repos.d/ownrepos.repo
  [ownrepos]
  name=that's my own repos
  baseurl=file:///myrepo
  gpgcheck=0 (not check from redhat or from any)
  enabled=1

- #yum clean all ➔ clean up the yum cache
- #yum repolist ➔ to list all system repos

# Controlling services and demons

- Systemd is the first process to start (PID1).
- Listing service units
  - #systemctl list-units --type=service ➔ to list all services
  - #systemctl --failed --type=service ➔ to list all failed services
  - #systemctl status sshd.service ➔ to check status of specific service
- Controlling system services
  - # systemctl start sshd.service #systemctl stop sshd.service
  - # systemctl enable sshd.service #systemctl disable sshd.service
  - #systemctl enable --now sshd.service
  - # systemctl restart sshd.service ➔ stop the service then start
  - systemctl reload sshd.service ➔ just reload the config files without stopping the service
- Masking and unmasking services
  - #systemctl mask httpd ➔ to **stop** the service and avoid to be started by mistake, no one can start it

    Created symlink /etc/systemd/system/httpd.service → /dev/null.
  - #systemctl start httpd

    Failed to start httpd.service: Unit httpd.service is masked.
  - #systemctl unmask httpd

    Removed "/etc/systemd/system/httpd.service".

# Scheduling(periodic)

- cron
- /var/spool/cron/user_name
- crontab -e
- crontab -u <user_name> -e
- Syntax ⇒ min hour day month no.of.day full_path_of_cmd
- 0-59 0-23 1-31 1-12 0-7(sunday) /usr/bin/date
- Output is sent to mail until you redirect it.
- crontab -l
- crontab -u <user_name> -l
- crontab –r ➔ will remove all tasks
- /etc/cron.allow ⇒ isn't exist by default
- /etc/cron.deny ⇒ it's exist by default
- Crontab mechanism ⇒ see the cron.allow then decide.

# Scheduling(one time)

- at
- /var/spool/at/user_name
- at 17:30 [at 17:30, at now+2 min,at 17:30+4 days, at noon, at midnight, at teatime ] teatime⇒ 5PM
  - at> then writ wanted command and Enter then the second command
  - Ctrl +d (at the end)
- Output is sent to mail until you redirect it.
- atq (list all at jobs)
- at -d <at_job-name> OR atrm <at_job_name>
- /etc/at.allow
- /etc/at.deny

# Storage management

| Type of device | Device naming pattern |
| --- | --- |
| SATA/SAS/USB-attached storage (SCSI driver) | `/dev/sda`, `/dev/sdb`, `/dev/sdc`, ... |
| `virtio-blk` paravirtualized storage (VMs) | `/dev/vda`, `/dev/vdb`, `/dev/vdc`,... |
| `virtio-scsi` paravirtualized storage (VMs) | `/dev/sda`, `/dev/sdb`, `/dev/sdc`, ... |
| NVMe-attached storage (SSDs) | `/dev/nvme0`, `/dev/nvme1`, ... |
| SD/MMC/eMMC storage (SD cards) | `/dev/mmcblk0`, `/dev/mmcblk1`, ... |

# Storage management

- 3 steps should be done to use a disk partition and access it’s data
  - Create the partition
    - #fdisk –l ➔ print full info about disks and their partitions
    - #df –h ➔ print all mounted partitions which ready for use
    - #fdisk  /dev/sda [to create a new partition or delete a partition]
    - #cat /proc/partitions
    - #partporbe
    - #reboot
    - #lsblk ➔ to check block devices we have attached to our system
    - #lsblk –fp ➔ to list full path of the device, the UUIDs and mount points, and the partition's file-system type
  - Format this partition with a certain file system
    - #mkfs -t ext3  /dev/sda1  OR  mkfs.ext3 /dev/sda5
    - #fsck ➔ to check the file system issues and try to repair it
  - Mount this partition to a free dir on our system
    - Temporary mount
      - #mount  /dev/sda5  /dir1
      - #df  -h
    - Permeant mount
      - #vi /etc/fstab

| HD_partition | mount_point | fs_type | options | dump | pass |
|---|---|---|---|---|---|
| /dev/sda1 | <mount_point> | fs_type | defaults | 0 | 0 |

      - After writing in fstab, you should reboot system or run mount command

# SWAP

- #swapon ➔ print info about swap partitions
- Create a SWAP
  - From disk partition
    - #fdisk /dev/sdb ➔ create /dev/sdb1 partition
    - During the partition creation change its type to be swap (click t)
    - #mkswap /dev/sdb1 ➔ make it a swap partition
    - #swapon /dev/sdb1 ➔ add it to swap area
    - #vi /etc/fstab ➔ add this partition on fstab file to be permanently on the system]

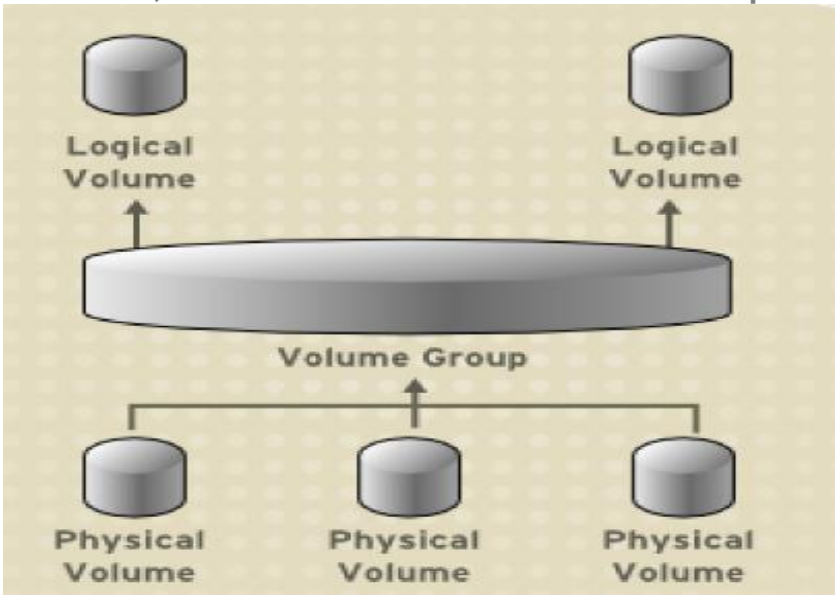
      /dev/sdb1 swap swap defaults 0 0
  - From file
    - #fallocate -l 4G /swapfile
    - #ls –lh /myswap
    - #chmod 600 /myswap
    - #mkswap /myswap
    - #swapon /myswap
    - #vi /etc/fstab ▫ add this partition on fstab file to be permanently on the system]

      /dev/sdb1 swap swap defaults 0 0

# LVM (Logical Volume Manager)

- If the HD was partitioned and then want to extend its size, These partitions should be LVM to apply that.
  - Partitions' type should be LVM
  - Then convert them to physical volumes.
  - Put all physical volumes in a volume group.
  - Then we can partition this volume group to logical volumes.
  - Mount these logical volumes on your system; it will be your system partitions.

- Here, Data is stored in an extent, before that in the normal partition data is stored in blocks.



# LVM

- Create a partition
  - #fdisk /dev/sda n p t 8e :wq
- Make physical volume
  - #pvcreate /dev/sda1 #pvs #pvdisplay #pvdisplay /dev/sda1 #pvremove /dev/sda1
- Create volume group
  - #vgcreate vg0 /dev/sda1 /dev/sda2 #vgs vgdisplay #vgremove vg0
- Create logical volume
  - #lvcreate -L +50G -n lv0 vg0 #lvs #lvdisplay #lvremove lv0
- Make file system
  - #mkfs.ext3 /dev/vg0/lv0
- mount and fstab
  - #mount /dev/vg0/lv0 /part1
  - #vi /etc/fstab
    /dev/vg0/lv0 /part1 ext3 defaults 0 0
- #vgextend vg0 /dev/sdb ➔ will create a PV from this disk and expand the VG
- #vgreduce vg0 /dev/sda3
- #lvextend -r -L +10G lv0 ➔ will add 10G to this lvm and –r to extend the XFS File system
- #lvextend –r –L 10G lv0 ➔ this will set this lvm to be 10G
- #resize2fs /dev/vg1/lv0
- #lvreduce -L -10G lv0

# Partition reducing

- Backup your data
- Identify the LVM and its mount point
- Umount the mountpoint
- Resize the filesystem
  - #resize2fs /dev/vg0/lv0 2G ➔ desired size
- Reduce the lvm size
  - #lvreduce -L 2G lv0