Day 7 Task: Understanding package manager and systemctl

**Package manager:**

**What is Package management?**

Management of packages, software, updates or upgrades where tasks involve like installation, upgrades, deletion, view package info, packages config is called Package management.

Manage software using YUM/DNF and RPM for your RHEL system like CentOS.

APT for Debian based Ubuntu, kali linux etc.

**Package management tool:**

* **Yum :** Yum is the primary package management tool for redhat

Yum performs operation like installing, updating and removing software packages.

How to install and remove a package.

#sudo yum install <package\_name> -y

#sudo yum remove <package\_name> -y

How to upgrade or update a package

#yum upgrade package

#yum update package

How to rollback or undo a package

#sudo yum history

#sudo yum history undo/redo id

* **Apt**

#sudo apt install <package\_name> -y

#sudo remove <package\_name>

#sudo apt autoremove

#sudo apt update

**$ sudo apt upgrade**

All the action of apt command, like installation, upgradation, deletion i.e the log file of apt is at:

**$ cat /var/log/dpkg.log**

**Task A:**

You have to install docker and jenkins in your system from your terminal using package managers

**Docket installation:**

Reference: https://docs.docker.com/engine/install/ubuntu/#uninstall-docker-engine

Prerequisite:

Uninstall all the unofficial dockers or docker releases.

Run the following command to uninstall all conflicting packages:

*$ for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done*

Uninstall the Docker Engine, CLI, containerd, and Docker Compose packages:

*$ sudo apt-get purge docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin docker-ce-rootless-extras*

Images, containers, volumes, or custom configuration files on your host aren't automatically removed. To delete all images, containers, and volumes

*$ sudo rm -rf /var/lib/docker*

*$ sudo rm -rf /var/lib/containerd*

**Install using the apt repository**

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker repository. Afterward, you can install and update Docker from the repository.

1. Set up Docker's apt repository.

*# Add Docker's official GPG key:*

*$ sudo apt-get update*

*$ sudo apt-get install ca-certificates curl gnupg*

*$ sudo install -m 0755 -d /etc/apt/keyrings*

*$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg*

*$ sudo chmod a+r /etc/apt/keyrings/docker.gpg*

*# Add the repository to Apt sources:*

*$ echo \*

*"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \*

*"$(. /etc/os-release && echo "$VERSION\_CODENAME")" stable" | \*

*$ sudo tee /etc/apt/sources.list.d/docker.list > /dev/null*

*$ sudo apt-get update*

1. Install the Docker package latest version

*$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin*

1. **Verify that the Docker Engine installation is successful by running the hello-world image.**

*$sudo docker run hello-world*

**Jenkins installation:**

Jenkins: Jenkins is a CI/CD software platform that supports **continuous integration** (CI) and **continuous delivery** (CD). It’s used to [automate software testing](https://phoenixnap.com/blog/best-automation-testing-tools), building, delivery, and deployment. Using Jenkins on Ubuntu creates a powerful management tool that takes a development workflow to the next level

Installation guide.

Note : At the time of writing this article, Jenkins only supports [Java 8 and Java 11 on Ubuntu](https://phoenixnap.com/kb/how-to-install-java-ubuntu).

***ubuntu@DevOps****:****~****$ java -version*

*Command 'java' not found, but can be installed with:*

*sudo apt install openjdk-11-jre-headless # version 11.0.20.1+1-0ubuntu1~22.04, or*

*sudo apt install default-jre # version 2:1.11-72build2*

*sudo apt install openjdk-17-jre-headless # version 17.0.8.1+1~us1-0ubuntu1~22.04*

*sudo apt install openjdk-18-jre-headless # version 18.0.2+9-2~22.04*

*sudo apt install openjdk-19-jre-headless # version 19.0.2+7-0ubuntu3~22.04*

*sudo apt install openjdk-8-jre-headless # version 8u382-ga-1~22.04.1*

$ sudo apt install openjdk-11-jre-headless

***ubuntu@DevOps****:****~****$ java -version*

*openjdk version "11.0.20.1" 2023-08-24*

*OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04)*

*OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04, mixed mode, sharing)*

*$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \*

[*https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key*](https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key)

*$ echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \*

*https://pkg.jenkins.io/debian-stable binary/ | sudo tee \*

*/etc/apt/sources.list.d/jenkins.list > /dev/null*

*$ sudo apt-get update*

*$ sudo apt-get install jenkins*

Reference: <https://www.jenkins.io/doc/book/installing/linux/#debianubuntu>

**Part 2) systemctl and system**

**systemctl** is a command-line utility in Linux that is used to manage and control the systemd system and service manager, which is a system initialization and service management framework used in many modern Linux distributions. Systemd is responsible for managing system processes and services during the system boot process and while the system is running. systemctl allows administrators to interact with and control systemd units, such as services, timers, sockets, and more. Here are some common uses of systemctl:

**Service Management:**

You can use systemctl to start, stop, restart, enable, disable, and check the status of services. For example:

*Start a service: sudo systemctl start servicename*

*Stop a service: sudo systemctl stop servicename*

*Restart a service: sudo systemctl restart servicename*

*Enable a service to start at boot: sudo systemctl enable servicename*

*Disable a service from starting at boot: sudo systemctl disable servicename*

*Check the status of a service: systemctl status servicename*

**Viewing Service Units:**

You can list all active services and their status using the systemctl list-units or systemctl list-units --type=service command.

**Viewing Service Logs:**

You can view the logs of a specific service using journalctl in combination with systemctl. For example:

**Copy code**

journalctl -u servicename

Managing Timers and Sockets:

systemctl can be used to manage timers (systemd's equivalent of cron jobs) and sockets.

**Changing the System State:**

systemctl also provides commands to change the system state, such as reboot, poweroff, and suspend.

**User Services:**

Systemd also allows users to manage their own user-level services using systemctl --user.

**Target Units:**

Systemd uses target units, which are similar to runlevels in older init systems. You can switch to a specific target unit using systemctl isolate targetname.

**Service Units Configuration:**

Service units are defined as configuration files usually located in /etc/systemd/system/ or /lib/systemd/system/. Administrators can modify these unit files to configure how services behave.

**systemctl** provides a centralized and standardized way to manage system services and resources, making it a powerful tool for system administrators and users. It plays a crucial role in modern Linux distributions, including Ubuntu, CentOS, and Fedora, among others.

Task :

1. Check the status of docker service in your system (make sure you completed above tasks, else docker won't be installed)

**ubuntu@DevOps**:**~**$ sudo systemctl status docker

**×** docker.service - Docker Application Container Engine

Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)

Active: **failed** (Result: exit-code) since Wed 2023-11-08 18:27:39 UTC; 12min ago

TriggeredBy: **×** docker.socket

Docs: https://docs.docker.com

Process: 7343 ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock **(code=exited, status=1/**>

Main PID: 7343 (code=exited, status=1/FAILURE)

CPU: 64ms

Nov 08 18:27:36 DevOps systemd[1]: **docker.service: Main process exited, code=exited, status=1/FAILURE**

Nov 08 18:27:36 DevOps systemd[1]: **docker.service: Failed with result 'exit-code'.**

Nov 08 18:27:36 DevOps systemd[1]: **Failed to start Docker Application Container Engine.**

Nov 08 18:27:39 DevOps systemd[1]: docker.service: Scheduled restart job, restart counter is at 3.

Nov 08 18:27:39 DevOps systemd[1]: Stopped Docker Application Container Engine.

Nov 08 18:27:39 DevOps systemd[1]: **docker.service: Start request repeated too quickly.**

Nov 08 18:27:39 DevOps systemd[1]: **docker.service: Failed with result 'exit-code'.**

Nov 08 18:27:39 DevOps systemd[1]: **Failed to start Docker Application Container Engine.**

***ubuntu@DevOps****:****~****$ sudo systemctl start docker*

***ubuntu@DevOps****:****~****$ sudo systemctl enable docker*

***ubuntu@DevOps****:****~****$ sudo systemctl status docker*

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$ sudo systemctl start docker

**ubuntu@DevOps**:**~**$ sudo systemctl status docker

**●** docker.service - Docker Application Container Engine

Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)

Active: **active (running)** since Wed 2023-11-08 18:40:25 UTC; 3s ago

TriggeredBy: **●** docker.socket

Docs: https://docs.docker.com

Main PID: 7398 (dockerd)

Tasks: 8

Memory: 57.7M

CPU: 247ms

CGroup: /system.slice/docker.service

└─7398 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Nov 08 18:40:24 DevOps systemd[1]: Starting Docker Application Container Engine...

Nov 08 18:40:24 DevOps dockerd[7398]: time="2023-11-08T18:40:24.541716238Z" level=info msg="Starting up"

Nov 08 18:40:24 DevOps dockerd[7398]: time="2023-11-08T18:40:24.557228185Z" level=info msg="detected 127.0.0.53 nameserver>

Nov 08 18:40:24 DevOps dockerd[7398]: time="2023-11-08T18:40:24.822230383Z" level=info msg="Loading containers: start."

Nov 08 18:40:25 DevOps dockerd[7398]: time="2023-11-08T18:40:25.103062432Z" level=info msg="Default bridge (docker0) is as>

Nov 08 18:40:25 DevOps dockerd[7398]: time="2023-11-08T18:40:25.186756404Z" level=info msg="Loading containers: done."

Nov 08 18:40:25 DevOps dockerd[7398]: time="2023-11-08T18:40:25.402198586Z" level=info msg="Docker daemon" commit=311b9ff >

Nov 08 18:40:25 DevOps dockerd[7398]: time="2023-11-08T18:40:25.403891376Z" level=info msg="Daemon has completed initializ>

Nov 08 18:40:25 DevOps dockerd[7398]: time="2023-11-08T18:40:25.483561815Z" level=info msg="API listen on /run/docker.sock"

Nov 08 18:40:25 DevOps systemd[1]: Started Docker Application Container Engine.

**stop the service jenkins and post before and after screenshots**

Reference: https://www.jenkins.io/doc/book/installing/linux/#debianubuntu

START JENKINS

**ubuntu@DevOps**:**~**$ sudo systemctl enable jenkins

Synchronizing state of jenkins.service with SysV service script with /lib/systemd/systemd-sysv-install.

Executing: /lib/systemd/systemd-sysv-install enable jenkins

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$ sudo systemctl start jenkins

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$ sudo systemctl status jenkins

**●** jenkins.service - Jenkins Continuous Integration Server

Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)

Active: **active (running)** since Wed 2023-11-08 18:16:12 UTC; 31min ago

Main PID: 5460 (java)

Tasks: 36 (limit: 1121)

Memory: 280.7M

CPU: 49.541s

CGroup: /system.slice/jenkins.service

└─5460 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/w>

Nov 08 18:15:36 DevOps jenkins[5460]: ------------------------

Nov 08 18:15:36 DevOps jenkins[5460]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword

Nov 08 18:15:36 DevOps jenkins[5460]: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nov 08 18:15:36 DevOps jenkins[5460]: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nov 08 18:15:36 DevOps jenkins[5460]: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Nov 08 18:16:12 DevOps jenkins[5460]: 2023-11-08 18:16:12.699+0000 [id=29] INFO jenkins.InitReactorRunner$1#>

Nov 08 18:16:12 DevOps jenkins[5460]: 2023-11-08 18:16:12.723+0000 [id=22] INFO hudson.lifecycle.Lifecycle#o>

Nov 08 18:16:12 DevOps systemd[1]: Started Jenkins Continuous Integration Server.

Nov 08 18:16:12 DevOps jenkins[5460]: 2023-11-08 18:16:12.887+0000 [id=44] INFO h.m.DownloadService$Download>

Nov 08 18:16:12 DevOps jenkins[5460]: 2023-11-08 18:16:12.888+0000 [id=44] INFO hudson.util.Retrier#start: P>

lines 1-20/20 (END)

**STOP JENKINS:**

**ubuntu@DevOps**:**~**$ sudo systemctl stop jenkins

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$ sudo systemctl status jenkins

○ jenkins.service - Jenkins Continuous Integration Server

Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)

Active: inactive (dead) since Wed 2023-11-08 19:04:27 UTC; 8s ago

Process: 5460 ExecStart=/usr/bin/jenkins **(code=exited, status=143)**

Main PID: 5460 (code=exited, status=143)

Status: "Jenkins stopped"

CPU: 51.382s

Nov 08 19:04:27 DevOps jenkins[5460]: 2023-11-08 19:04:27.374+0000 [id=24] INFO jenkins.model.Jenkins$16#onA>

Nov 08 19:04:27 DevOps jenkins[5460]: 2023-11-08 19:04:27.378+0000 [id=24] INFO jenkins.model.Jenkins#\_clean>

Nov 08 19:04:27 DevOps jenkins[5460]: 2023-11-08 19:04:27.392+0000 [id=24] INFO jenkins.model.Jenkins#\_clean>

Nov 08 19:04:27 DevOps jenkins[5460]: 2023-11-08 19:04:27.392+0000 [id=24] INFO jenkins.model.Jenkins#\_clean>

Nov 08 19:04:27 DevOps jenkins[5460]: 2023-11-08 19:04:27.403+0000 [id=24] INFO jenkins.model.Jenkins#\_clean>

Nov 08 19:04:27 DevOps jenkins[5460]: 2023-11-08 19:04:27.405+0000 [id=24] INFO hudson.lifecycle.Lifecycle#o>

Nov 08 19:04:27 DevOps jenkins[5460]: 2023-11-08 19:04:27.414+0000 [id=24] INFO o.e.j.s.handler.ContextHandl>

Nov 08 19:04:27 DevOps systemd[1]: jenkins.service: Deactivated successfully.

Nov 08 19:04:27 DevOps systemd[1]: Stopped Jenkins Continuous Integration Server.

Nov 08 19:04:27 DevOps systemd[1]: jenkins.service: Consumed 51.382s CPU time.

**Task 3:** read about the commands systemctl vs service

**systemctl and service** are both commands used to manage services in Linux, but they are part of different service management systems and are commonly used in different contexts. Here's an overview of both commands and their differences:

**systemctl:**

**Systemd:** systemctl is a command used to manage services and other units (such as sockets and timers) controlled by the systemd init system.

**Systemd** is a modern initialization system and service manager that is widely used in many Linux distributions, including Ubuntu 16.04 and later, CentOS 7 and later, and Fedora.

**Functionality:** **systemctl** is a more feature-rich and versatile command. It can be used to start, stop, enable, disable, restart, reload, and check the status of services. It also provides better integration with the **systemd** system and is generally the recommended way to manage services on **systemd-based** systems.

**Examples:**

Start a service: sudo systemctl start servicename

Stop a service: sudo systemctl stop servicename

Enable a service to start at boot: sudo systemctl enable servicename

Disable a service from starting at boot: sudo systemctl disable servicename

service:

**SysV Init:** service is a command commonly used on older Linux distributions that use the SysV init system. These distributions include CentOS 6 and earlier, and Debian 7 and earlier.

**Functionality:** service is a simpler command compared to systemctl. It is used to start, stop, and check the status of services that are managed by the SysV init system. It lacks some of the advanced features and flexibility provided by systemctl.

**Examples:**

Start a service: sudo service servicename start

Stop a service: sudo service servicename stop

Check the status of a service: sudo service servicename status

In summary, if you are using a Linux distribution that utilizes the systemd init system (which is the case for most modern distributions), it is recommended to use the systemctl command for managing services. On older distributions that use SysV init, you may use the service command. However, as Linux distributions evolve, many are transitioning to systemd, making systemctl the more widely used and standardized way to manage services across different distributions.