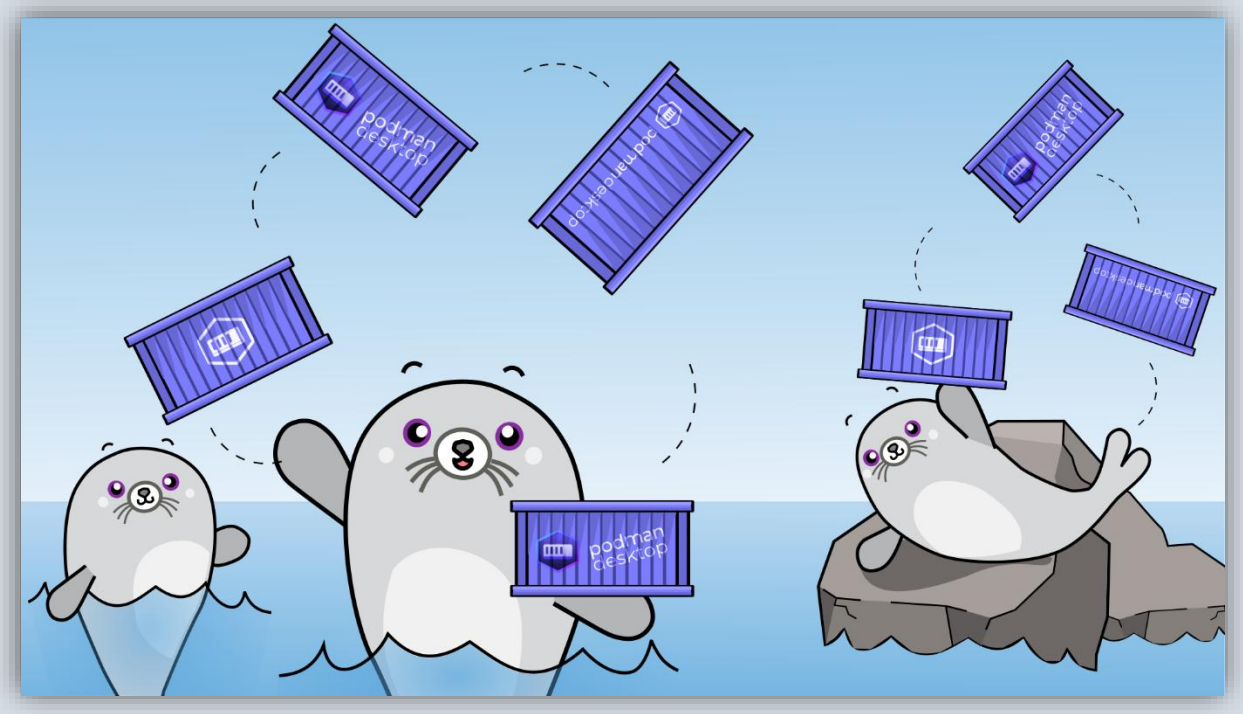
Podman: A Comprehensive Guide with Detailed Hands-on Examples



1.1 What is Podman?

Podman (**Pod Manager**) is an **open-source, daemonless container engine** that allows users to **run, manage, and build OCI (Open Container Initiative) containers**. It is a **lightweight and secure alternative to Docker**, designed to work without a background service or root privileges.

Podman provides a **Docker-compatible CLI**, making it easy for users to transition from Docker. It also **supports rootless containers**, improving security by allowing users to run containers without administrative access.

1.2 Key Features of Podman

✅ **Daemonless** – No long-running background process like Docker.
✅ **Rootless Mode** – Containers run with user permissions, increasing security.
✅ **Docker-Compatible CLI** – Works with Docker commands using alias docker=podman.
✅ **Kubernetes-Native** – Supports pods and integration with Kubernetes.

✅ **Security-Enhanced** – Uses systemd for better process management.
✅ **Buildah Integration** – Can build images using Buildah directly.

1.3 Why Use Podman Over Docker?

- **More Secure** – No privileged daemon process.
- **Better Resource Management** – Lower system overhead.
- **Rootless Containers** – Ideal for multi-user environments.
- **Works with Systemd** – Better service control for production workloads.

---

2. Podman vs. Docker

Feature	Podman	Docker
Daemon Requirement	No	Yes
Security	More secure (rootless mode)	Requires root daemon
Running Containers	Uses systemd for service management	Uses Docker daemon
Networking	Uses CNI (Container Networking Interface)	Uses built-in networking
Compatibility	Can use Docker commands	Docker-specific
Pod Support	Supports Kubernetes pods	Does not support pods natively

---

3. Installing Podman

Podman is available on various Linux distributions.

3.1 Install Podman on RHEL 9 / CentOS 9

- **sudo dnf install -y podman**

3.2 Install Podman on Fedora

- **sudo dnf install -y podman**

3.3 Install Podman on Ubuntu/Debian

- sudo apt update
- sudo apt install -y podman

3.4 Install Podman on Arch Linux

- sudo pacman -S podman

3.5 Verify Installation

- podman --version

```
controlplane:~$ podman --version
podman version 4.9.3
```

---

4. Podman Basics: Managing Containers

Command	Description
podman images	List available container images.
podman pull <image>	Pull an image from a registry.
podman run <image>	Run a container from an image.
podman ps	List running containers.
podman stop <container>	Stop a running container.
podman rm <container>	Remove a stopped container.
podman rmi <image>	Remove an image.

---

5. Hands-on: Running Containers with Podman

Let's go step by step to run and manage an **Nginx container** using Podman.

5.1 Step 1: Pull an Image

- podman pull nginx

```
controlplane:~$ podman pull nginx
Resolving "nginx" using unqualified-search registries (/etc/containers/registries.conf)
Trying to pull docker.io/library/nginx:latest...
Getting image source signatures
Copying blob 943ea0f0c2e4 done   |
Copying blob 7cf63256a31a done   |
Copying blob bf9acace214a done   |
Copying blob 513c3649bb14 done   |
Copying blob d014f92d532d done   |
Copying blob 9dd21ad5a4a6 done   |
Copying blob 103f50cb3e9f done   |
```

This downloads the latest **Nginx** image from Docker Hub.

5.2 Step 2: Run a Container

- **podman run -d --name mynginx -p 8080:80 nginx**

```
controlplane:~$ podman run -d --name mynginx -p 8080:80 nginx:latest
17a48d4ed0451b6af03410718e817f8d0dd099e7ab1b573a4eedf8815b860947
```

- -d → Runs the container in **detached mode** (in the background).
- --name mynginx → Assigns a name to the container.
- -p 8080:80 → Maps **port 8080** on the host to **port 80** in the container.

5.3 Step 3: Verify the Running Container

- **podman ps**

```
controlplane:~$ podman ps
CONTAINER ID  IMAGE                           COMMAND               CREATED         STATUS         PORTS                   NAMES
17a48d4ed045  docker.io/library/nginx:latest  nginx -g daemon o...  49 seconds ago  Up 49 seconds  0.0.0.0:8080->80/tcp  mynginx
```

This shows a list of running containers.

5.4 Step 4: Access the Container in a Browser

Open a web browser and go to:

- http://localhost:8080

You should see the **Nginx welcome page**.

5.5 Step 5: Stop and Remove the Container

- **podman stop mynginx**
- **podman rm mynginx**

```
controlplane:~$ podman stop mynginx
mynginx
controlplane:~$ podman rm mynginx
mynginx
```

6. Podman Rootless Mode

Podman allows **non-root users** to run containers without requiring elevated privileges.

6.1 Enabling Rootless Podman

Switch to a normal user and run:

- **podman info**

If Podman is running in **rootless mode**, it will display:

- **rootless: true**

```
sidhu@controlplane:~$ podman info | grep -i rootless
    rootless: true
```

6.2 Running a Rootless Container

- **podman run -d --name mynginx -p 8080:80 nginx**

```
sidhu@controlplane:~$ podman run -d --name mynginx -p 8080:80 nginx
Resolving "nginx" using unqualified-search registries (/etc/containers/registries.conf)
Trying to pull docker.io/library/nginx:latest...
Getting image source signatures
Copying blob 943ea0f0c2e4 done   |
Copying blob 7cf63256a31a done   |
Copying blob bf9acace214a done   |
Copying blob 513c3649bb14 done   |
Copying blob d014f92d532d done   |
Copying blob 9dd21ad5a4a6 done   |
Copying blob 103f50cb3e9f done   |
Copying config b52e0b094b done   |
Writing manifest to image destination
ce631500fa4966eb3df0f2e4012aded6ab13c2d902f91b7bb8b2dcd0af9fb9aa
```

This runs **Nginx** in rootless mode, reducing security risks.

---

7. Creating and Managing Images with Podman

Podman can **build and modify container images** similar to Docker.

7.1 Step 1: Create a Containerfile (Dockerfile Alternative)

```
FROM ubuntu:latest
RUN apt update && apt install -y nginx
CMD ["nginx", "-g", "daemon off;"]
```

```
controlplane:~$ cat Containerfile
FROM ubuntu:latest
RUN apt update && apt install -y nginx
CMD ["nginx", "-g", "daemon off;"]
```

7.2 Step 2: Build the Image

- podman build -t mynginx-container .

```
controlplane:~$ podman build -t mynginx-container .
STEP 1/3: FROM ubuntu:latest
Resolved "ubuntu" as an alias (/etc/containers/registries.conf.d/shortnames.conf)
Trying to pull docker.io/library/ubuntu:latest...
Getting image source signatures
Copying blob 5a7813e071bf done   |
Copying config a04dc4851c done   |
Writing manifest to image destination
STEP 2/3: RUN apt update && apt install -y nginx

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:3 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1024 kB]
```

7.3 Step 3: Run the Custom Image

- podman run -d -p 8080:80 mynginx-container

```
controlplane:~podman run -d -p 8080:80 mynginx-container
8a3e79a1a0dc1cc6e6195a0a2ca2e635cab7a214af9bed16893eb57746f382fb
```

---

8. Working with Pods in Podman

Podman supports **Kubernetes-style pods**, grouping multiple containers.

8.1 Creating a Pod

- podman pod create --name mypod -p 8080:80

```
controlplane:~$ podman pod create --name mypod -p 8080:80
57bb0b4f563b90633ff87d29c16bcf35178bc7dcce0f2fb1682c305dda92d4f0
```

8.2 Adding Containers to a Pod

- podman run -d --pod mypod nginx

```
controlplane:~$ podman run -d --pod mypod nginx
Resolving "nginx" using unqualified-search registries (/etc/containers/registries.conf)
Trying to pull docker.io/library/nginx:latest...
Getting image source signatures
Copying blob 97f5c0f51d43 done   |
Copying blob 6e909acdb790 done   |
Copying blob 5eaa34f5b9c2 done   |
Copying blob 417c4bccf534 done   |
Copying blob e7e0ca015e55 done   |
Copying blob 373fe654e984 done   |
Copying blob c22eb46e871a done   |
Copying config 53a18edff8 done   |
Writing manifest to image destination
f9feddcffd7f1ec9da24eae86f4cc7ddfce65174e7b2f6a72d0ddd052c2ef969
```

This starts an **Nginx container** inside the pod.

8.3 Viewing Pod Details

- podman pod ps

```
controlplane:~$ podman pod ps
POD ID        NAME        STATUS      CREATED         INFRA ID      # OF CONTAINERS
57bb0b4f563b  mypod       Running     9 minutes ago   47c5856ad56c  2
```

---

9. Podman and Systemd Integration

Podman can generate **systemd service files** for running containers as services.

9.1 Generating a Systemd Service for a Container

```
controlplane:~$ podman run -itd --name mynginx -p 18080:80 nginx
1a1be19a915d6cc661c1fd11a46aa21e0d7cc3ba8cb5128c4c2c6fb6510e5407
```

```
[root@localhost ~]# cd .config/systemd/user/
```

- podman generate systemd --name mynginx –new –files

```
[root@localhost user]# podman generate systemd --name mynginx --new --files

DEPRECATED command:
It is recommended to use Quadlets for running containers and pods under systemd.

Please refer to podman-systemd.unit(5) for details.
/root/.config/systemd/user/container-mynginx.service
```

9.2 Starting the Container as a Service

- systemctl --user enable podman-mynginx.service
- systemctl --user start podman-mynginx.service

```
[root@localhost user]# ls
container-mynginx.service  default.target.wants
[root@localhost user]# systemctl daemon-reload
[root@localhost user]# systemctl start container-mynginx.service --user
[root@localhost user]# systemctl enable container-mynginx.service --user
[root@localhost user]# systemctl status container-mynginx.service --user
● container-mynginx.service - Podman container-mynginx.service
     Loaded: loaded (/root/.config/systemd/user/container-mynginx.service; enabled; preset: disabled)
     Active: active (running) since Sat 2025-03-29 11:47:35 IST; 12s ago
       Docs: man:podman-generate-systemd(1)
   Main PID: 3579 (conmon)
      Tasks: 1 (limit: 20080)
     Memory: 1.2M
        CPU: 2.267s
     CGroup: /user.slice/user-0.slice/user@0.service/app.slice/container-mynginx.service
             └─3579 /usr/bin/conmon --api-version 1 -c 5095f0158089a87c3ca0de56c2a8d00986dea13dd5c8cd9eccb37883251c8429 -u 5>

Mar 29 11:47:35 localhost.localdomain mynginx[3579]: /docker-entrypoint.sh: Configuration complete; ready for start up
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: using the "epoll" event method
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: nginx/1.27.4
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14>
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: OS: Linux 5.14.0-503.26.1.el9_5.x86_64
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:104>
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: start worker processes
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: start worker process 24
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: start worker process 25
Mar 29 11:47:35 localhost.localdomain mynginx[3579]: 2025/03/29 06:17:35 [notice] 1#1: start worker process 26
```

10. Pushing and Pulling Images in Podman

Podman can push and pull images from **Docker Hub, Quay.io, and private registries**.

10.1 Pushing an Image to Docker Hub

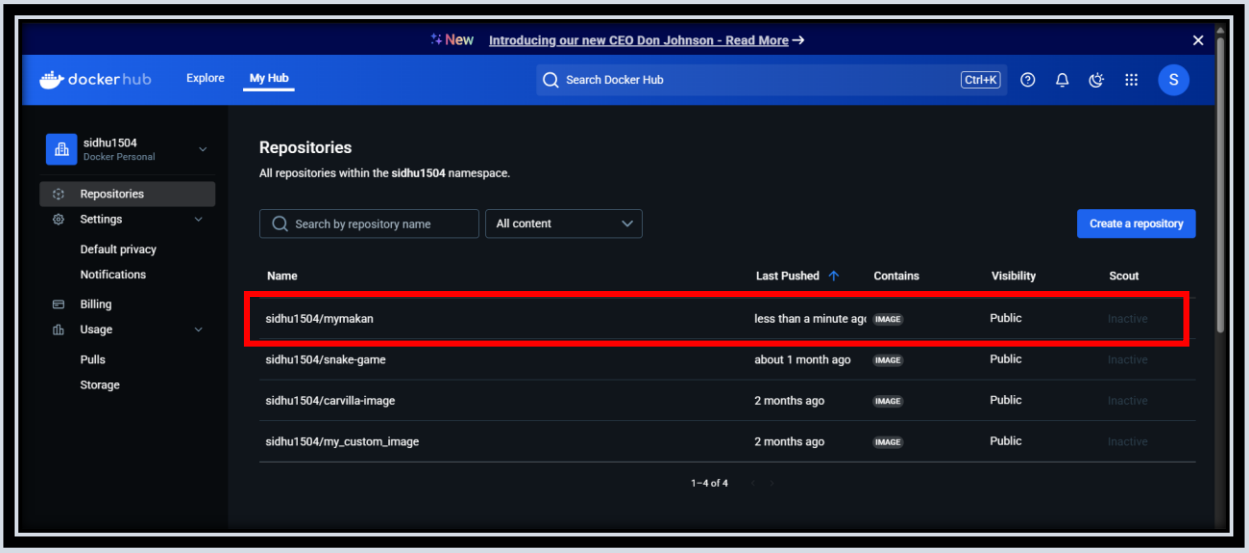
```
controlplane:~$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to
create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using
 SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: sidhu1504
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

- podman tag mynginx-container docker.io/yourusername/mynginx-container

```
controlplane:~$ podman tag mymakan:latest docker.io/sidhu1504/mymakan
controlplane:~$ podman images
REPOSITORY                  TAG         IMAGE ID      CREATED         SIZE
docker.io/sidhu1504/mymakan latest      1ab7dfea2854  4 minutes ago   209 MB
localhost/mymakan           latest      1ab7dfea2854  4 minutes ago   209 MB
docker.io/library/ubuntu    latest      a04dc4851cbc  2 months ago    80.7 MB
```

- podman push docker.io/yourusername/mynginx-container

```
controlplane:~$ podman push docker.io/sidhu1504/mymakan:latest
Getting image source signatures
Copying blob eafcc29a2f24 done   |
Copying blob 251c71a67d99 done   |
Copying blob b32fe66fb482 done   |
Copying blob 4b7c01ed0534 done   |
Copying config 1ab7dfea28 done   |
Writing manifest to image destination
```



10.2 Pushing an Image to Quay.io

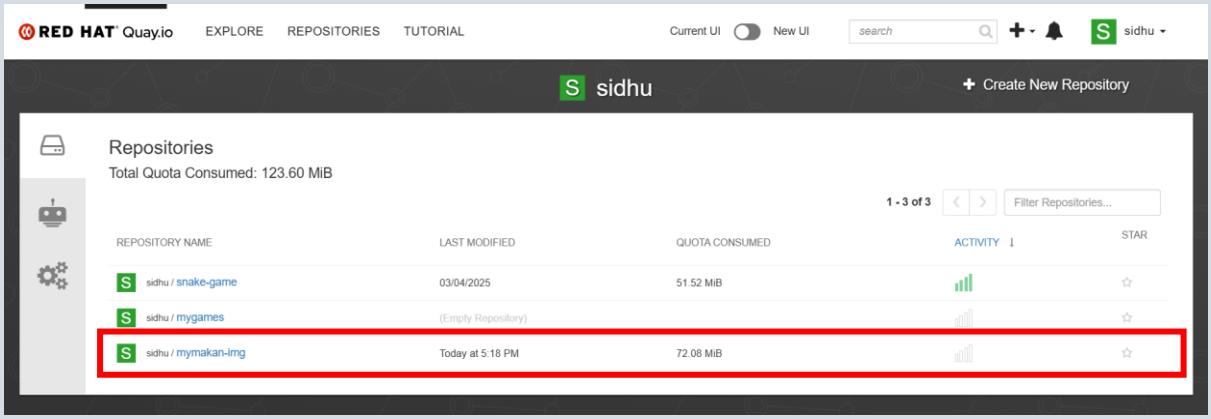
```
controlplane:~$ docker login quay.io
Username: sidhu
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

- podman tag mynginx-container quay.io/yourusername/mynginx-container

```
controlplane:~$ podman tag mymakan:latest quay.io/sidhu/mymakan
controlplane:~$ podman images
REPOSITORY                  TAG         IMAGE ID      CREATED         SIZE
quay.io/sidhu/mymakan       latest      1ab7dfea2854  17 minutes ago  209 MB
docker.io/sidhu1504/mymakan latest      1ab7dfea2854  17 minutes ago  209 MB
localhost/mymakan           latest      1ab7dfea2854  17 minutes ago  209 MB
docker.io/library/ubuntu    latest      a04dc4851cbc  2 months ago    80.7 MB
```

- podman push quay.io/yourusername/mynginx-container

```
controlplane:~$ podman push quay.io/sidhu/mymakan-img
Getting image source signatures
Copying blob f6ef60526dc3 done   |
Copying blob 9d3f6e7414b7 done   |
Copying blob d14690e911d5 done   |
Copying blob 4b7c01ed0534 done   |
Copying config cfe70beb1b done   |
Writing manifest to image destination
```



11. Running Containers in Kubernetes with Podman

Step 1: Generate Kubernetes YAML

```
controlplane:~$ podman pod ls
POD ID        NAME        STATUS      CREATED         INFRA ID      # OF CONTAINERS
b0d89faf3d98  mypod       Running     2 minutes ago   d5d879455c59  2
```

- podman generate kube mypod > mypod.yaml

```
controlplane:~$ podman generate kube mypod > mypod.yaml
```

Step 2: Apply the Pod in Kubernetes

- kubectl apply -f mypod.yaml

```
controlplane:~$ kubectl apply -f mypod.yaml
pod/mypod created
```

---

12. Troubleshooting Podman Issues

Issue	Solution
"Command not found"	Ensure Podman is installed using which podman.
"Image pull failed"	Try podman pull --tls-verify=false <image>
"Container fails to start"	Check logs using podman logs <container>

---

13. Conclusion

Podman is a **secure, lightweight, and daemonless** container engine. It offers a **Docker-compatible experience** while providing **rootless execution, systemd integration, and pod support**.