

Red Hat Podman

- [Introduction](#)
 - [Podman, Buildah et Skopeo](#)
- [Installation](#)
 - [Utiliser Podman sur une base ArchLinux/Manjaro](#)
- [Les commandes de bases](#)
 - [Podman](#)
- [Les astuces](#)
 - [Créer un pod et l'exporter dans un fichier de définition Kubernetes](#)

Introduction

Podman, Buildah et Skopeo

Liens officiels :

- Podman : <https://podman.io/>
- Buildah : <https://buildah.io/>
- Skopeo : <https://github.com/containers/skopeo>

Ressources :

- **Podman and Buildah for Docker users**
- **Containers without daemons: Podman and Buildah available in RHEL 7.6 and RHEL 8**
- **Running Linux containers as a non-root with Podman**
- **Buildah, Podman, and Skopeo**

Documentations :

- **Lien vers la documentation Red Hat Enterprise 8 (Podman, Buildah, Skopeo)**

Installation

Utiliser Podman sur une base ArchLinux/Manjaro

Afin d'utiliser Podman et/ou Buildah pour construire des containers "unprivileged" sur un système ArchLinux ou Manjaro, il est nécessaire de suivre plusieurs étapes d'installation supplémentaires.

Tout d'abord, il faut un noyau qui prend en charge les "user namespaces". Tous les noyaux Arch Linux prennent en charge cette fonctionnalité de base, cependant, le noyau Arch par défaut est livré avec les "user namespaces" activés uniquement pour l'utilisateur root...

- Tout d'abord, il faut créer deux fichiers qui vont être utilisés par Podman pour créer des containers "unprivileged" :

```
$ sudo /etc/subgid
votreuser:10000:65536

$ sudo /etc/subuid
votreuser:10000:65536
```

Explication : l'utilisateur root (UID 0) du container va être mappé sur le système avec un UID compris entre 10000 et 75536 (+ 65536), et donc ses privilèges seront minimales.

- Enfin, il convient de créer un fichier de configuration sysctl afin d'activer le mécanisme d'user namespaces pour les utilisateurs non-root. Pour cela, créer un fichier `/etc/sysctl.d/00-local-usersns.conf` et ajouter le contenu suivant :

```
# vi /etc/sysctl.d/00-local-usersns.conf
kernel.unprivileged_usersns_clone=1
```

- Reboot le système !

Les commandes de bases

Podman

Commande	Description	Commande	Description
attach	S'attacher à un container en cours d'exécution	commit	Créer une image depuis un container modifié
build	Construire une image depuis un Dockerfile	create	Créer un container (mais ne le démarre pas)
diff	Inspecter les changements dans un filesystem d'un container	exec	Exécuter un processus au sein d'un container
export	Exporter le contenu du filesystem du container	help, h	Afficher la liste des commandes disponibles
history	Afficher l'historique d'une image	images	Lister les images disponible sur le stockage local
import	Importer une archive pour créer une image de filesystem	info	Affiche les informations systèmes
inspect	Afficher la configuration d'une image ou d'un container	kill	Envoie un signal "kill" vers un ou plusieurs containers
load	Charger une image depuis une archive	login	Se connecter à une registry
logout	Se déconnecter d'une registry	logs	Récupérer les logs d'un container
mount	Monter un root filesystem d'un container	pause	Mettre en pause tous les processus dans un ou plusieurs containers
ps	Lister les containers	port	Lister les ports mappés pour un container
pull	Télécharger une image depuis un registry	push	Pousser une image vers une registry

restart	Redémarrer un ou plusieurs containers	rm	Supprimer un ou plusieurs containers
rmi	Supprimer une ou plusieurs images du stockage local	run	Exécuter une commande dans un container
save	Sauvegarder une image vers une archive	search	Recherche une image dans les registries disponible
start	Démarrer un ou plusieurs containers	stats	Afficher différentes stats pour un container comme le CPU ou les I/O
stop	Stopper un ou plusieurs containers	tag	Taguer une image
top	Afficher le processus en cours dans un container	umount, unmount	Démonter un root filesystem d'un container
unpause	"Unpause" les processus au sein d'un container	version	Afficher la version de Podman
wait	Interrompre l'exécution d'un ou plusieurs containers		

Les astuces

Créer un pod et l'exporter dans un fichier de définition Kubernetes

Attention : Contrairement à l'exécution de containers de manière standard sans droits root sur Podman, la création et le management de pods nécessitent des droits root (sudo) à l'heure actuelle.

- **Commençons par créer un pod vide :**

```
$ sudo podman pod create --name demo
```

Vous pouvez aussi créer un pod lors de la première exécution d'un container avec le mot clé "new" dans le flag "--pod" :

```
$ sudo podman run -dt --pod new:demo -p 8080:80 --name nginx-demo docker.io/library/nginx
```

- **Listons les pods disponibles :**

```
$ sudo podman pod list
```

POD ID	NAME	STATUS	CREATED	# OF CONTAINERS	INFRA ID
b57d1832894f	demo	Running	1 minutes ago	1	2ce0af493a0c

La présence d'un container au sein de ce pod "vide" est normal, pour chaque pod Podman, un container appelé "infra" est créé. Ce conteneur ne fait rien (sleep). Son but est de contenir les namespaces associés au pod et de permettre à Podman de connecter d'autres conteneurs à ce pod. Cela vous permet de démarrer et d'arrêter les conteneurs à l'intérieur du pod tout permettant à ce pod de rester en fonctionnement, alors que si le conteneur primaire contrôlait le pod, cela ne serait pas possible, le pod serait à l'arrêt.

- **Attacher un nouveau container à ce pod :**

```
$ sudo podman run -dt --pod demo --name nginx-demo docker.io/library/nginx
```

```
$ sudo podman pod ps
```

POD ID	NAME	STATUS	CREATED	# OF CONTAINERS	INFRA ID
b57d1832894f	demo	Running	16 minutes ago	2	2ce0af493a0c

```
$ sudo podman ps -a --pod
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
302dc5040c15	docker.io/library/nginx:latest	nginx -g daemon o...	2 seconds ago	Up	2 seconds ago
nginx-demo	b57d1832894f				
2ce0af493a0c	k8s.gcr.io/pause:3.1		15 minutes ago	Up	5 minutes ago
infra	b57d1832894f				b57d1832894f-

- **A partir de ce pod en état de fonctionnement, vous pouvez exporter un fichier de définition Kubernetes. Vous pouvez ainsi déployer votre pod créé avec Podman sur votre cluster Kubernetes.**

```
$ sudo podman generate kube demo > demo.yml
# Generation of Kubernetes YAML is still under development!
#
# Save the output of this file and use kubectl create -f to import
# it into Kubernetes.
#
# Created with podman-1.3.1
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: "2019-05-27T14:15:37Z"
  labels:
    app: demo
    name: demo
spec:
  containers:
  - command:
    - nginx
    - -g
    - daemon off;
  env:
```

```
- name: PATH
  value: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
- name: TERM
  value: xterm
- name: HOSTNAME
- name: container
  value: podman
- name: NGINX_VERSION
  value: 1.15.12-1~stretch
- name: NJS_VERSION
  value: 1.15.12.0.3.1-1~stretch
image: docker.io/library/nginx:latest
name: nginx-demo
resources: {}
securityContext:
  allowPrivilegeEscalation: true
  capabilities: {}
  privileged: false
  readOnlyRootFilesystem: false
tty: true
workingDir: /
status: {}
```