Créer rapidement un Conteneur avec systemd

Table of Contents

Les conteneurs **systemd-nspawn** s'apparentent davantage aux prisons (jail) FreeBSD qu'aux conteneurs Docker. Il s'agit essentiellement d'un chroot sophistiqué avec des intégrations intégrées pratiques avec systemd. On peut démarrer, arrêter, activer et désactiver les conteneurs comme s'il s'agissait de services réguliers.

Cependant **systemd-nspawn** est une fonctionnalité expérimentale qui n'a pas été complètement testée ou auditée. Il n'y a aucune garantie de sécurité ou de stabilité; il est probablement préférable de ne pas les utiliser en production pour le moment.

Création du conteneur

On va travailler sur une nouvelle installation de Rocky8, mais ce processus est possible sur n'importe quel système utilisant systemd. La seule différence sera la première étape d'installation du package. Évidemment, sur Debian, on n'utilise pas yum, et sur Fedora, le noms de dépôt et les versions seront différents (et on utilisera dnf)…

Installation du conteneur

La première étape consiste à “installer” un nouveau système de fichiers racine dans un répertoire.

```
yum -y --releasever=8 --nogpg --installroot=/var/lib/machines/cool-container
\
   --disablerepo='*' --enablerepo=baseos install systemd passwd yum rocky-
release
```

Cela créera un répertoire, /var/lib/machines/cool-container, et le remplira avec un nouveau système de fichiers racine et quelques packages principaux.

Mot de passe root

Deuxièmement, il faut entrer dans le conteneur pour configurer certaines choses de base comme le mot de passe root. On peut utiliser la commande systemd-nspawn directement pour cela :

```
systemd-nspawn -D /var/lib/machines/cool-container
```

Cela ouvre un shell dans le conteneur sans réellement “démarrer” quoi que ce soit à l'intérieur de celui-ci (comme un chroot à partir d'un mode de récupération). À partir de là, on peut définir le mot de passe root afin de pouvoir se connecter plus tard.

Pour une raison quelconque, l'installation ne défini pas la “bonne” étiquette SELinux sur /etc/passwd. Mais il a mis une étiquette sur /bin/passwd. C'est cette incompatibilité qui pose un problème lors de l'utilsation de la commande **passwd**.

Il faut s'assurer que toutes les étiquettes SELinux sont réinitialisées (À la valeur attendue par l'hôte du conteneur, c'est-à-dire sans étiquette), mais tenter d'exécuter la restauration avec `-Rv /` à l'intérieur du conteneur ne fait rien. IIRC libselinux détecte quand il est exécuté dans un conteneur et ne fait rien.

Il faut donc exécuter **restorecon** depuis l'extérieur du conteneur :

```
restorecon -Rv /var/lib/machines/cool-container
```

Ensuite, On peut définir le mot de passe root avec succès.

Désactivation audit du noyau

Si on démarre le conteneur on recevra probablement un avertissement avant le démarrage du conteneur concernant le sous-système d'audit du noyau. Il y a des bogues étranges qui peuvent apparaître si l'audit est activé, il est donc conseillé de simplement le désactiver.

Il faut juste ajouter un indicateur aux paramètres du noyau dans le chargeur de démarrage. Cela variera d'une distribution à l'autre, mais pour CentOS, il suffit de modifier le fichier `/etc/sysconfig/grub` et de modifier la variable **GRUB_CMDLINE_LINUX** en ajoutant `audit=0` à la liste des paramètres.

Après avoir modifié les paramètres, il faut régénérer la configuration GRUB :

```
grub2-mkconfig -o /etc/grub2.cfg
```

Lorsqu'on a terminé, faire simplement ^D pour retourner à le machine hôte.

Utilisation du conteneur

On a maintenant un squelette de conteneur installé, mais il faut encore configurer ce qu'il contient et le préparer pour qu'il démarre automatiquement, ou au moins en tant que service de systemd.

Configuration du système de base

Puisqu'on a maintenant accès au compte root, on peut “démarrer” complètement le conteneur :

```
systemd-nspawn -bD /var/lib/machines/cool-container
```

Le drapeau **-b** est l'abréviation de **–boot** et signifie essentiellement que **systemd-nspawn** recherchera un binaire init et l'exécutera. On verrale journal de démarrage standard passer, puis une invite de connexion PTY standard. Se connecter avec les informations d'identification root qu'on aconfigurées précédemment, et on peut maintenant commencer à installer les choses comme si on était sur une toute nouvelle machine.

Une fois qu'on a configuré le conteneur et que tout est en cours d'exécution et configuré, on peutquitter en « arrêtant » le conteneur comme s'il s'agissait d'une machine physique.

Gestion du conteneur

Alors que le préfixe /var/lib/machines au début pouvait sembler arbitraire, en fait c'était intentionnel - les conteneurs de ce répertoire seront automatiquement découverts par systemd et on pourra les activer et les gérer automatiquement.

Pour que le conteneur démarre avec tout le reste au démarrage de l'hôte :

```
systemctl enable systemd-nspawn@cool-container
```

Et on peut démarrer et arrêter le conteneur comme n'importe quel autre service :

```
systemctl start systemd-nspawn@cool-container
systemctl stop systemd-nspawn@cool-container
```

Accéder au conteneur

Accéder à un conteneur en cours d'exécution peut être un peu délicat ; Une option consiste à installer **openssh** dans le conteneur et à le faire fonctionner sur un port non standard (car les conteneurs partagent les interfaces réseau de l'hôte). Alternativement, on peut accéder à la machine via **machinectl**.

Machinectl sans arguments listera tous les conteneurs en cours d'exécution (et autres machines virtuelles, etc.). On peut utiliser la commande de connexion **machinectl** (l'ancienne version de

machinectl sur CentOS ne permet pas d'utiliser l'argument de connexion, il faut donc installer openssh):

```
machinectl login cool-container
```

Pour Sortir de cette invite sans nécessairement arrêter le conteneur , il existe un bouton de panique pour se déconnecter : appuyer sur le bouton **Echap** trois fois en une seconde (c'est-à-dire rapidement).