LXD: Mappage des IDs dans l'espace de noms d'utilisateur

Table of Contents

- LXD: Mappage des IDs dans l'espace de noms d'utilisateur
- Mappage des espaces de noms
  - Gammes autorisées
  - Migrations des hôtes
  - Différents idmaps par conteneur
  - cartes id personnalisées
- Exemples d'utilisation lors du montage de /home dans LXD
  - Montage /home en lecture seule
  - Montage /home en lecture-écriture
    - Autoriser LXD à remapper l'ID utilisateur
    - Remappage de l'ID utilisateur à l'intérieur du conteneur
    - Montage de /home

LXD gère des conteneurs sécurisés. Ceci est réalisé principalement par l'utilisation des espaces de noms utilisateur permettant d'exécuter des conteneurs sans privilège,limitant considérablement la surface d'attaque.

Les espaces de noms utilisateur nécessitent un noyau>= 3.12, LXD démarrera même sur les anciens les noyaux, mais refusera de démarrer les conteneurs.

Mappage des espaces de noms

Le mappage des espaces de noms d'utilisateur fonctionnent en liant un ensemble de UID et de GID sur l'hôte à un ensemble d'uid et de gids dans le conteneur

Par exemple, on peut définir que les UIDs et GIDs hôtes de 100000 à 165535 seront utilisés par LXD et doivent être mappé sur uid/gid 0 à 65535 dans le conteneur.

En conséquence, un processus exécuté en tant que uid 0 dans le conteneur sera en réalité en cours d'exécution en tant que 100000 UID.

Les allocations des UID et des GID doivent toujours être d'au moins de 65536 pour couvrir la gamme POSIX comprenant root (0) et nobody (65534).

Gammes autorisées

Sur la plupart des hôtes, LXD vérifiera dans **/etc/subuid** et **/etc/subgid** les ID pour allocation à un utilisateur “lxd” au premier démarrage, définir la valeur par défaut profile pour utiliser les 65536 premiers uids et gids de cette gamme :

- Si la plage est inférieure à 65 536 (ce qui n'inclut aucune plage), alors LXD ne parviendra pas à créer ou à démarrer un conteneur jusqu'à ce que cela soit corrigé.
- Si l'un des fichiers **/etc /subuid**, **/etc/subgid**, **newuidmap** (recherche de chemin) et **newgidmap** (chemin d'accès) n'existe pas sur le système, le démarrage de tout conteneur par LXD échouera jusqu'à ce que cela soit corrigé car cela montre un configuration de **shadow** brisée.
- Si aucun de ces 4 fichiers ne peut être trouvé, LXD supposera qu'il est en cours d'exécution. sur un hôte utilisant une ancienne version de **shadow**. Dans ce mode, LXD assumera qu'il peut utiliser tous les uids et gids supérieurs à 65535 et prendra la première 65536 comme sa carte par défaut.

shadow est un fichier qui contient les informations de mot de passe pour les comptes du système.

Migrations des hôtes

La carte source doit être embarquée lors du déplacement de conteneurs entre des hôtes afin qu’ils puisse être remappé sur l'hôte récepteur.

Différents idmaps par conteneur

LXD prend en charge l’utilisation de différentes cartes idmaps par conteneur pour isoler davantage conteneurs les uns des autres. Ceci est contrôlé avec des clés de configuration par conteneur:

- Les conteneurs avec **security.idmap.isolated** auront une plage d'identifiant unique calculée pour eux parmi les autres conteneurs avec **security.idmap.isolated** défini (si aucune est disponible, la définition de cette clé échouera tout simplement).
- Pour les conteneurs avec **security.idmap.size**, leur plage d'id sera définie sur la taille. Les conteneurs isolés sans cette propriété définissent par défaut une plage d'id de taille 65536; cela permet la conformité POSIX et un utilisateur “nobody” à l'intérieur du récipient.
- Pour sélectionner une carte spécifique, la clé **security.idmap.base** permet d'écraser le mécanisme de détection automatique et d'indiquer à LXD quel hôte utiliser comme base pour le conteneur.

Ces propriétés nécessitent un redémarrage du conteneur pour prendre effet.

cartes id personnalisées

LXD prend également en charge la personnalisation de bits de idmap, par exemple. pour permettre aux utilisateurs de se lier des parties du système de fichiers de l'hôte dans un conteneur sans avoir besoin de Système de fichiers. La clé de configuration par conteneur pour cela est `raw.idmap`, et ressemble à:

```
both 1000 1000
uid 50-60 500-510
gid 100000-110000 10000-20000
```

La première ligne configure les uid et gid 1000 sur l'hôte pour mapper vers les uid 1000 à l'intérieur du conteneur (cela peut être utilisé par exemple pour lier le montage d'un utilisateur de base dans un conteneur).

Les deuxième et troisième lignes mappent uniquement les plages uid ou gid dans le conteneur, respectivement. La deuxième entrée par ligne est l'identifiant source, c'est-à-dire l'identifiant de l'hôte, et la troisième entrée est la plage à l'intérieur du conteneur. Ces gammes doivent être de même taille.

Cette propriété nécessite un redémarrage du conteneur pour prendre effet.

Exemples d'utilisation lors du montage de /home dans LXD

Montage /home en lecture seule

L'idée de base est d'utiliser la gestion des périphériques de LXD pour monter le répertoire /home de l'hôte sur /home de l'invité:

```
$ lxc init ubuntu-daily:z unmapped
$ lxc config device add unmapped homedir disk source=$HOME path=/home/ubuntu
$ lxc start unmapped
```

C'est la seule chose à faire pour y avoir accès en lecture seule, car l'utilisateur **ubuntu** de l'invité (UID 1000 dans l'invité) verra très bien less fichiers /home, bien qu'ils appartiennent à un UID. et GID "65534" =- nobody. L'utilisateur ubuntu de l'invité ne peut donc pas écrire dans ce répertoire, mais peut voir les fichiers en fonction des autorisations "other" normales.

Montage /home en lecture-écriture

Autoriser LXD à remapper l'ID utilisateur

LXD utilise des espaces de noms Linux pour isoler les processus. Par défaut, même root n'est pas autorisé à réutiliser les UID de l'hôte à l'intérieur de conteneurs.

La première étape consiste à autoriser LXD à remapper l'ID utilisateur de l'hôte dans un conteneur:

Il est probable que $ UID soit égal à 1000 sur un système Ubuntu mono utilisateur.

```
echo "root:$UID:1" | sudo tee -a /etc/subuid /etc/subgid
```

C'est une étape unique, il ne sera plus jamais nécessaire de répéter cela sur l'hôte.

Remappage de l'ID utilisateur à l'intérieur du conteneur

Une fois que LXD est autorisé à remapper l'UID, il faut lui indiquer de le faire dans le conteneur:

```
$ lxd init ubuntu-daily:z remapped
$ lxc config set remapped raw.idmap "both $UID 1000"
```

Il existe de nombreuses syntaxe, mais **“both $UID 1000”** permet simplement de “mapper à la fois l'UID et le GID, de $UID de l'hôte à 1000 l'UID et le GID de l'invité”.

Pour un mappage plus explicites définir plutôt **“uid $UID 1000” et “gid $(id -g) 1000”** , mais la syntaxe “both” est pratique.

Montage de /home

Comme auparavant, on peut maintenant monter /home à l'intérieur du conteneur, pour y accéder en lecture-écriture à l'intérieur du conteneur:

```
$ lxc config device add remapped homedir disk source=$HOME path=/home/ubuntu
```