Creating a full LXD server On Rocky Linux (Italian version)

A book from the Documentation Team

Version : 2025/05/21

Rocky Documentation Team

Table of contents

1. Licence

RockyLinux offers Linux courseware for trainers or people wishing to learn how to administer a Linux system on their own.

The documents and their sources are freely downloadable from:

- https://docs.rockylinux.org
- https://github.com/rocky-linux/documentation

Our media sources are hosted at github.com. You'll find the source code repository where the version of this document was created.

From these sources, you can generate your own personalized training material using mkdocs. You will find instructions for generating your document here.

How can I contribute to the documentation project?

You'll find all the information you need to join us on our git project home page.

We wish you all a pleasant reading and hope you enjoy the content.

2. Creare un server LXD completo

Informazione

Questa procedura dovrebbe funzionare per Rocky Linux 8.x o 9.x. Se cercate un'implementazione moderna di questo progetto degli ex sviluppatori di LXD, ma disponibile solo per Rocky Linux 9.x, date un'occhiata a [il libro di Incus Server] (../incus_server/00-toc.md).

Cosa è successo al progetto LXD

Più di un anno fa, nella mailing list lxc-users è stato pubblicato il seguente annuncio:

Canonical, the creator and main contributor of the LXD project has decided that after over eight years as part of the Linux Containers community, the project would now be better served directly under Canonical's own set of projects.

Uno dei fattori decisivi sono state le dimissioni di alcuni sviluppatori principali di LXD, i quali hanno poi effettuato il fork di LXD in Incus, annunciando il fork nell'agosto 2023. Una versione di rilascio (0.1) è stata rilasciata nell'ottobre 2023, e da allora gli sviluppatori hanno rapidamente sviluppato questa versione con rilasci successivi fino alla 0.7 (marzo 2024). Dopo la 0.7 è arrivata la versione di supporto a lungo termine, la 6.0 LTS, il 4 aprile 2024, e ora la 6.4 LTS (da settembre 2024).

Durante tutto il processo, si pensava che Cannonical avrebbe continuato a mantenere i collegamenti alle immagini dei container fornite da Linux Containers, ma a causa di un cambio di licenza è diventato impossibile per Linux Containers continuare a offrire le immagini dei container all'interno di LXD. Tuttavia, a causa di un cambio di licenza, è diventato impossibile per Linux Containers continuare a offrire le immagini dei container all'interno di LXD. Mentre Linux Containers non può più fornire immagini di container a LXD, il progetto LXD è riuscito a costruire alcuni container, compresi quelli per Rocky Linux.

Questo documento utilizza LXD anziché Incus.

2.1 Introduzione

LXD è descritto meglio sul sito web ufficiale, ma consideratelo come un sistema di container che offre i vantaggi dei server virtuali in un container.

È molto potente e, con l'hardware e la configurazione giustia, è possibile creare molte istanze di server su un singolo pezzo di hardware. Se lo si abbina a un server snapshot, si ha anche una serie di container che possono essere avviati quasi immediatamente se il server primario si guasta.

(Non si deve pensare a questo come a un backup tradizionale. È comunque necessario un sistema di backup regolare di qualche tipo, ad esempio rsnapshot)

La curva di apprendimento di LXD può essere un po' ripida, ma questo libro cercherà di fornirvi un bagaglio di conoscenze a portata di mano, per aiutarvi a distribuire e utilizzare LXD su Rocky Linux.

Per coloro che desiderano utilizzare LXD come ambiente di laboratorio sui propri notebook o workstation, vedere l'Appendice A: Configurazione della workstation.

2.2 Prerequisiti e presupposti

- Un server Linux Rocky, ben configurato. Considerare un disco rigido separato per lo spazio disco ZFS (è necessario se si usa ZFS) in un ambiente di produzione. E sì, il presupposto è un server bare metal, non un VPS (Virtual Private Server).
- Si tratta di un argomento avanzato, ma non troppo difficile da comprendere. Se si seguono queste istruzioni fin dall'inizio, si dovrebbe avere successo. Detto questo, la conoscenza di alcune nozioni di base sulla gestione dei container è molto utile.
- Comfort dalla riga di comando sulla/e macchina/e e dimestichezza con l'editor della riga di comando. (In questi esempi si utilizza *vi*, ma è possibile sostituirlo con il proprio editor preferito.)
- Per la maggior parte di questi processi è necessario essere un utente non privilegiato. Per le prime fasi di configurazione, è necessario essere l'utente root o essere in grado di usare `sudo` per diventarlo. In tutti questi capitoli si assume che l'utente non privilegiato sia "lxdadmin". L'account utente dovrà essere creato più avanti nel processo.
- Per ZFS, assicurarsi che il UEFI secure boot NON sia abilitato. Altrimenti, si finirà per dover firmare il modulo ZFS per farlo caricare.
- Utilizzo di container basati su Rocky Linux per la maggior parte del tempo

2.3 Sinossi

- Il **Capitolo 1: Installazione e configurazione** tratta l'installazione del server primario. In generale, il modo corretto di utilizzare LXD in produzione è quello di avere un server primario e un server snapshot.
- Il **Capitolo 2: Configurazione di ZFS** tratta l'impostazione e la configurazione di ZFS. ZFS è un gestore di volumi logici e un file system open source creato da Sun Microsystems, originariamente per il suo sistema operativo Solaris.
- Il **Capitolo 3: Inizializzazione di LXD e configurazione dell'utente** si occupa dell'inizializzazione di base e delle opzioni, nonché della configurazione

dell'utente non privilegiato che verrà utilizzato per la maggior parte del resto del processo

- **Capitolo 4: Impostazione del firewall** Ha le opzioni di configurazione di `firewalld`
- Il **capitolo 5: Impostazione e gestione delle immagini** descrive il processo di installazione delle immagini del sistema operativo in un contenitore e la loro configurazione
- Il **Capitolo 6: Profili** si occupa dell'aggiunta di profili e della loro applicazione ai container, in particolare di macvlan e della sua importanza per l'indirizzamento IP sulla LAN o sulla WAN
- Il **capitolo 7: Opzioni di configurazione dei contenitori** copre brevemente alcune delle opzioni di configurazione di base per i contenitori e offre alcuni vantaggi ed effetti collaterali per la modifica delle opzioni di configurazione
- Il **capitolo 8: Istantanee dei contenitori** illustra in dettaglio il processo di istantanea dei contenitori sul server primario
- Il **capitolo 9: Il server snapshot** tratta l'impostazione e la configurazione del server snapshot e come creare una relazione simbiotica tra il server primario e il server snapshot
- Il **capitolo 10: Automatizzazione delle istantanee** tratta dell'automazione della creazione di istantanee e del popolamento del server di istantanee con le istantanee
- L'**Appendice A: Configurazione della Workstation** non fa tecnicamente parte dei documenti del server di produzione, ma offre una soluzione per chi vuole costruire un laboratorio di contenitori LXD sul proprio notebook o stazione di lavoro personale.

2.4 Conclusioni

Questi capitoli consentono di configurare efficacemente un server LXD primario e uno snapshot di livello aziendale. Nel corso di questo processo, imparerete molto su LXD. Siate consapevoli che c'è ancora molto da imparare e considerate questi documenti come un punto di partenza.

Il vantaggio principale di LXD è che è economico da usare su un server, consente di avviare rapidamente le installazioni del sistema operativo e permette di eseguire molti server applicativi stand-alone su un singolo componente hardware, sfruttando l'hardware per il massimo utilizzo.

3. Capitolo 1: Installazione e configurazione

In tutto questo capitolo dovrete essere l'utente root o dovrete essere in grado di passare a root con *sudo*.

3.1 Installare i repository EPEL e OpenZFS

LXD richiede i repository EPEL (Extra Packages for Enterprise Linux), che sono facili da installare:

```
dnf install epel-release
```

Una volta installato, verificare che non vi siano aggiornamenti:

```
dnf upgrade
```

Se durante il processo di aggiornamento sono stati effettuati aggiornamenti del kernel, riavviare il server.

3.1.1 Repository OpenZFS per 8 e 9

Installare il repository OpenZFS con:

```
dnf install https://zfsonlinux.org/epel/zfs-release-2-2$(rpm --eval "%
{dist}").noarch.rpm
```

3.2 Installare `snapd`, `dkms`, `vim` e `kernel-devel`

L'installazione di LXD richiede un pacchetto snap su Rocky Linux. Per questo motivo, è necessario installare `snapd` (e alcuni altri programmi utili) con:

```
dnf install snapd dkms vim kernel-devel
```

Ora abilitate e avviate snapd:

```
systemctl enable snapd
```

Quindi eseguire:

```
systemctl start snapd
```

Riavviare il server prima di continuare.

3.3 Installare LXD

L'installazione di LXD richiede l'uso del comando snap. A questo punto, si sta solo installando, non si sta eseguendo la configurazione:

```
snap install lxd
```

3.4 Installare OpenZFS

```
dnf install zfs
```

3.5 Impostazione dell'ambiente

La maggior parte delle impostazioni del kernel del server non sono sufficienti per eseguire un gran numero di container. Se si presume fin dall'inizio che si utilizzerà il server in produzione, è necessario apportare queste modifiche in anticipo per evitare errori come "Too many open files".

Fortunatamente, modificare le impostazioni di LXD non è difficile, basta modificare alcuni file e riavviare il sistema.

3.5.1 Modifica di `limits.conf`

Il primo file da modificare è il file `limits.conf.` Questo file è autodocumentato. Esaminate le spiegazioni nei commenti del file per capire cosa fa questo file. Per apportare le modifiche, inserire:

```
vi /etc/security/limits.conf
```

L'intero file è costituito da commenti e, in fondo, mostra le impostazioni predefinite correnti. Nello spazio vuoto sopra il marcatore di fine file (#End of file) è necessario aggiungere le nostre impostazioni personalizzate. Una volta completato, il file avrà il seguente aspetto:

```
# Modifications made for LXD

*               soft    nofile          1048576
*               hard    nofile          1048576
root            soft    nofile          1048576
root            hard    nofile          1048576
*               soft    memlock         unlimited
*               hard    memlock         unlimited
```

Salva le tue modifiche ed esci (⇧ Shift + : + w + q + ! per *vi*).

3.5.2 Modifica di sysctl.conf con `90-lxd.override.conf`

Con *systemd*, è possibile apportare modifiche alla configurazione generale del sistema e alle opzioni del kernel *senza* modificare il file di configurazione principale. Le impostazioni vanno invece inserite in un file separato che sovrascrive le impostazioni specifiche necessarie.

Per apportare queste modifiche al kernel, si deve creare un file chiamato `90-lxd-override.conf` in `/etc/sysctl.d`. Per farlo, digitare:

```
vi /etc/sysctl.d/90-lxd-override.conf
```

> **RL 9 e valore MAX di `net.core.bpf_jit_limit`**
>
> A causa dei recenti aggiornamenti della sicurezza del kernel, il valore massimo di `net.core.bpf_jit_limit` sembra essere 1000000000. Se si utilizza Rocky Linux 9.x, regolare questo valore nel file di autodocumentazione sottostante. Se lo si imposta al di sopra di questo limite **O** se non lo si imposta affatto, verrà impostato il valore predefinito di sistema di 264241152, che potrebbe non essere sufficiente se si esegue un numero elevato di contenitori.

Inserite il seguente contenuto nel file. Se vi state chiedendo cosa state facendo, il contenuto del file è autodocumentante:

```
## The following changes have been made for LXD ##

# fs.inotify.max_queued_events specifies an upper limit on the number of events
```

```
that can be queued to the corresponding inotify instance
 - (default is 16384)

fs.inotify.max_queued_events = 1048576

# fs.inotify.max_user_instances This specifies an upper limit on the number of
inotify instances that can be created per real user ID -
(default value is 128)

fs.inotify.max_user_instances = 1048576

# fs.inotify.max_user_watches specifies an upper limit on the number of watches
that can be created per real user ID - (default is 8192)

fs.inotify.max_user_watches = 1048576

# vm.max_map_count contains the maximum number of memory map areas a process
may have. Memory map areas are used as a side-effect of cal
ling malloc, directly by mmap and mprotect, and also when loading shared
libraries - (default is 65530)

vm.max_map_count = 262144

# kernel.dmesg_restrict denies container access to the messages in the kernel
ring buffer. Please note that this also will deny access t
o non-root users on the host system - (default is 0)

kernel.dmesg_restrict = 1

# This is the maximum number of entries in ARP table (IPv4). You should
increase this if you create over 1024 containers.

net.ipv4.neigh.default.gc_thresh3 = 8192

# This is the maximum number of entries in ARP table (IPv6). You should
increase this if you plan to create over 1024 containers.Not nee
ded if not using IPv6, but...

net.ipv6.neigh.default.gc_thresh3 = 8192

# This is a limit on the size of eBPF JIT allocations which is usually set to
PAGE_SIZE * 40000. Set this to 1000000000 if you are running Rocky Linux 9.x

net.core.bpf_jit_limit = 3000000000

# This is the maximum number of keys a non-root user can use, should be higher
than the number of containers

kernel.keys.maxkeys = 2000
```

```
# This is the maximum size of the keyring non-root users can use

kernel.keys.maxbytes = 2000000

# This is the maximum number of concurrent async I/O operations. You might need
to increase it further if you have a lot of workloads th
at use the AIO subsystem (e.g. MySQL)

fs.aio-max-nr = 524288
```

Salvare le modifiche e uscire.

A questo punto riavviare il server.

3.5.3 Controllo dei valori di *sysctl.conf*

Dopo il riavvio, accedere nuovamente al server come utente root. È necessario verificare che il nostro file di override abbia effettivamente completato il lavoro.

Non è difficile da fare. Non è necessario verificare tutte le impostazioni, a meno che non lo si voglia fare, ma controllarne alcune consente di verificare che le impostazioni siano state modificate. Per farlo, utilizzare il comando `sysctl`:

```
sysctl net.core.bpf_jit_limit
```

Che vi mostrerà:

```
net.core.bpf_jit_limit = 3000000000
```

Eseguire la stessa operazione con alcune altre impostazioni del file di sovrascrittura per verificare le modifiche.

4. Capitolo 2: Configurazione di ZFS

In tutto questo capitolo è necessario essere l'utente root o poter fare `sudo` per diventare root.

Se avete già installato ZFS, questa sezione vi guiderà nella configurazione di ZFS.

4.1 Abilitazione di ZFS e impostazione del pool

Per prima cosa, inserite questo comando:

```
/sbin/modprobe zfs
```

Se non ci sono errori, si ritorna al prompt e non viene emesso alcun eco. Se si verifica un errore, interrompere subito e iniziare la risoluzione dei problemi. Anche in questo caso, assicuratevi che il secure boot sia disattivato. Questo sarà il colpevole più probabile.

Successivamente è necessario esaminare i dischi del sistema, scoprire dove si trova il sistema operativo e quali sono disponibili per il pool ZFS. Lo si farà con `lsblk`:

```
lsblk
```

Che restituirà qualcosa di simile (il vostro sistema sarà diverso!):

```
AME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop0    7:0    0  32.3M  1 loop /var/lib/snapd/snap/snapd/11588
loop1    7:1    0  55.5M  1 loop /var/lib/snapd/snap/core18/1997
loop2    7:2    0  68.8M  1 loop /var/lib/snapd/snap/lxd/20037
sda      8:0    0 119.2G  0 disk
├─sda1   8:1    0   600M  0 part /boot/efi
├─sda2   8:2    0     1G  0 part /boot
├─sda3   8:3    0  11.9G  0 part [SWAP]
├─sda4   8:4    0     2G  0 part /home
└─sda5   8:5    0 103.7G  0 part /
sdb      8:16   0 119.2G  0 disk
├─sdb1   8:17   0 119.2G  0 part
└─sdb9   8:25   0     8M  0 part
sdc      8:32   0 149.1G  0 disk
└─sdc1   8:33   0 149.1G  0 part
```

In questo elenco, si può notare che */dev/sda* è utilizzato dal sistema operativo. Si utilizzerà */dev/sdb* per il nostro zpool. Si noti che se si dispone di molti dischi rigidi, si può prendere in considerazione l'uso di raidz (un software raid specifico per ZFS).

Questo aspetto esula dall'ambito di questo documento, ma è sicuramente da prendere in considerazione per la produzione. Offre migliori prestazioni e ridondanza. Per ora, create il vostro pool sul singolo dispositivo che avete identificato:

```
zpool create storage /dev/sdb
```

Questo dice di creare un pool chiamato "storage" che è ZFS sul dispositivo */dev/sdb*.

Dopo aver creato il pool, riavviare il server.

5. Capitolo 3: Inizializzazione di LXD e configurazione dell'utente

In questo capitolo è necessario essere root o poter fare `sudo` per diventare root. Inoltre, si presuppone che sia stato configurato un pool di archiviazione ZFS descritto nel Capitolo 2. È possibile utilizzare un pool di archiviazione diverso se si è scelto di non utilizzare ZFS, ma sarà necessario apportare modifiche alle domande e alle risposte di inizializzazione.

5.1 Inizializzazione di LXD

L'ambiente del server è pronto. Si è pronti a inizializzare LXD. Si tratta di uno script automatico che pone una serie di domande per rendere operativa l'istanza LXD:

```
lxd init
```

Ecco le domande e le nostre risposte per lo script, con una piccola spiegazione dove necessario:

```
Would you like to use LXD clustering? (yes/no) [default=no]:
```

Se siete interessati al clustering, fate ulteriori ricerche al riguardo qui

```
Do you want to configure a new storage pool? (yes/no) [default=yes]:
```

Questo sembra controintuitivo. Il pool ZFS è già stato creato, ma sarà chiaro in una domanda successiva. Accettare l'impostazione predefinita.

```
Name of the new storage pool [default=default]: storage
```

Lasciare questo nome "default" è un'opzione, ma per chiarezza è meglio usare lo stesso nome dato al pool ZFS.

```
Name of the storage backend to use (btrfs, dir, lvm, zfs, ceph) [default=zfs]:
```

You want to accept the default.

```
Create a new ZFS pool? (yes/no) [default=yes]: no
```

È qui che entra in gioco la risoluzione della domanda precedente sulla creazione di un pool di storage.

```
Name of the existing ZFS pool or dataset: storage
Would you like to connect to a MAAS server? (yes/no) [default=no]:
```

Il Metal As A Service (MAAS) non rientra nell'ambito di questo documento.

```
Would you like to create a new local network bridge? (yes/no) [default=yes]:
What should the new bridge be called? [default=lxdbr0]:
What IPv4 address should be used? (CIDR subnet notation, “auto” or “none”)
[default=auto]:
What IPv6 address should be used? (CIDR subnet notation, “auto” or “none”)
[default=auto]: none
```

Se si desidera utilizzare IPv6 sui propri contenitori LXD, è possibile attivare questa opzione. Questo dipende da voi.

```
Would you like the LXD server to be available over the network? (yes/no)
[default=no]: yes
```

È necessario per eseguire lo snapshot del server.

```
Address to bind LXD to (not including port) [default=all]:
Port to bind LXD to [default=8443]:
Trust password for new clients:
Again:
```

Questa password di fiducia è il modo in cui ci si connette al server snapshot o si torna indietro dal server snapshot. Impostate qualcosa che abbia senso nel vostro ambiente. Salvare questa voce in un luogo sicuro, ad esempio in un gestore di password.

```
Would you like stale cached images to be updated automatically? (yes/no)
[default=yes]
Would you like a YAML "lxd init" preseed to be printed? (yes/no) [default=no]:
```

5.2 Impostazione dei privilegi degli utenti

Prima di continuare, è necessario creare l'utente "lxdadmin" e assicurarsi che abbia i privilegi necessari. È necessario che l'utente "lxdadmin" sia in grado di eseguire il `sudo` a root e che sia membro del gruppo lxd. Per aggiungere l'utente e assicurarsi che sia membro di entrambi i gruppi, procedere come segue:

```
useradd -G wheel,lxd lxdadmin
```

Impostare la password:

```
passwd lxdadmin
```

Come per le altre password, salvatela in un luogo sicuro.

6. Capitolo 4: configurazione del firewall

In questo capitolo è necessario essere root o poter fare `sudo` per diventare root.

Come per qualsiasi server, è necessario assicurarsi che sia sicuro sia dal mondo esterno che sulla propria LAN. Il server di esempio ha solo un'interfaccia LAN, ma è assolutamente possibile avere due interfacce, una per ciascuna delle reti LAN e WAN.

> **Una nota riguardante Rocky Linux 9.x e `iptables`**
>
> A partire da Rocky Linux 9.0, `iptables` e tutte le utility associate sono ufficialmente deprecate. Ciò significa che nelle future versioni del sistema operativo scompariranno del tutto. Una versione precedente di questo documento conteneva istruzioni per la configurazione di `iptables`, ma ora è stata rimossa.
>
> Per tutte le versioni attuali di Rocky Linux, si raccomanda l'uso di `firewalld`.

6.1 Configurazione del firewall - `firewalld`

Per le regole di *firewalld*, è necessario utilizzare questa procedura di base o avere familiarità con questi concetti. Le nostre ipotesi sono: Rete LAN 192.168.1.0/24 e un bridge chiamato lxdbr0. Per essere chiari, si potrebbero avere molte interfacce sul server LXD, con una magari rivolta verso la WAN. Si creerà anche una zona per le reti bridged e locali. Questo solo per chiarezza di zona. Gli altri nomi delle zone non sono applicabili. Questa procedura presuppone che si conoscano già le basi di *firewalld*.

```
firewall-cmd --new-zone=bridge --permanent
```

È necessario ricaricare il firewall dopo aver aggiunto una zona:

```
firewall-cmd --reload
```

Si vuole consentire tutto il traffico dal bridge. È sufficiente aggiungere l'interfaccia e modificare il target da "default" a "ACCEPT":

> **Attenzione**
>
> La modifica del target di una zona `firewalld` deve essere fatta con l'opzione `--permanent`, quindi tanto vale inserire questo flag anche negli altri comandi e rinunciare all'opzione `--runtime-to-permanent`.

Nota

Se si desidera creare una zona che consenta tutti gli accessi all'interfaccia o alla sorgente, ma non si vuole specificare alcun protocollo o servizio, è necessario cambiare l'obiettivo da "default" ad "ACCEPT". Lo stesso vale per "DROP" e "REJECT" per un particolare blocco IP per il quale sono state create zone personalizzate. Per essere chiari, la zona "drop" si occuperà di questo aspetto, a patto che non si utilizzi una zona personalizzata.

```
firewall-cmd --zone=bridge --add-interface=lxdbr0 --permanent
firewall-cmd --zone=bridge --set-target=ACCEPT --permanent
```

Supponendo che non ci siano errori e che tutto funzioni ancora, è sufficiente ricaricare:

```
firewall-cmd --reload
```

Se ora si elencano le regole con `firewall-cmd --zone=bridge --list-all` si vedrà:

```
bridge (active)
  target: ACCEPT
  icmp-block-inversion: no
  interfaces: lxdbr0
  sources:
  services:
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Si noti che si desidera consentire anche l'interfaccia locale. Anche in questo caso, le zone incluse non hanno un nome appropriato. Creare una zona e utilizzare l'intervallo IP di origine per l'interfaccia locale per garantire l'accesso:

```
firewall-cmd --new-zone=local --permanent
firewall-cmd --reload
```

Aggiungere gli IP di origine per l'interfaccia locale e modificare la destinazione in "ACCEPT":

```
firewall-cmd --zone=local --add-source=127.0.0.1/8 --permanent
firewall-cmd --zone=local --set-target=ACCEPT --permanent
firewall-cmd --reload
```

Procedere con l'elenco della zona "local" per assicurarsi che le regole siano presenti con `firewall-cmd --zone=local --list all`, che visualizzerà:

```
local (active)
  target: ACCEPT
  icmp-block-inversion: no
  interfaces:
  sources: 127.0.0.1/8
  services:
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Si vuole consentire SSH dalla nostra rete fidata. In questo caso utilizzeremo gli IP di origine e la zona "trusted" integrata. L'obiettivo di questa zona è già "ACCEPT" per impostazione predefinita.

```
firewall-cmd --zone=trusted --add-source=192.168.1.0/24
```

Aggiungere il servizio alla zona:

```
firewall-cmd --zone=trusted --add-service=ssh
```

Se tutto funziona, spostare le regole su permanente e ricaricarle:

```
firewall-cmd --runtime-to-permanent
firewall-cmd --reload
```

L'elenco della zona "fidata" visualizzerà:

```
trusted (active)
  target: ACCEPT
```

```
  icmp-block-inversion: no
  interfaces:
  sources: 192.168.1.0/24
  services: ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Per impostazione predefinita, la zona "pubblica" è nello stato abilitato e ha SSH consentito. Per motivi di sicurezza, non si vuole che SSH sia consentito nella zona "pubblica". Assicurarsi che le zone siano corrette e che l'accesso al server avvenga tramite uno degli IP della LAN (nel nostro esempio). Se non lo verificate prima di continuare, potreste rimanere bloccati fuori dal server. Quando si è sicuri di avere accesso dall'interfaccia corretta, rimuovere SSH dalla zona "pubblica":

```
firewall-cmd --zone=public --remove-service=ssh
```

Verificate l'accesso e assicuratevi di non essere bloccati. In caso contrario, spostare le regole su permanenti, ricaricare ed eliminare la zona "pubblica" per garantire la rimozione di SSH:

```
firewall-cmd --runtime-to-permanent
firewall-cmd --reload
firewall-cmd --zone=public --list-all
```

Potrebbero esserci altre interfacce da considerare sul vostro server. È possibile utilizzare le zone integrate, ove opportuno, ma se i nomi non appaiono logici, è possibile aggiungere zone. Ricordate che se non ci sono servizi o protocolli che dovete consentire o rifiutare in modo specifico, dovrete cambiare il target di zona. Se è possibile utilizzare le interfacce, come nel caso del bridge, è possibile farlo. Se avete bisogno di un accesso più granulare ai servizi, utilizzate invece gli IP di origine.

7. Capitolo 5: Configurazione e gestione delle immagini

In questo capitolo dovrete eseguire i comandi come utente non privilegiato ("lxdadmin" se avete seguito questo libro dall'inizio).

7.1 Elenco delle immagini disponibili

Probabilmente non vedete l'ora di iniziare con un contenitore. Esistono molte possibilità di sistemi operativi per container. Per avere un'idea del numero di possibilità, inserite questo comando:

```
lxc image list images: | more
```

Digitare la barra spaziatrice per sfogliare l'elenco. Questo elenco di container e macchine virtuali continua a crescere.

L'**ultima** cosa che si vuole fare è cercare un'immagine del contenitore da installare, soprattutto se si conosce l'immagine che si vuole creare. Modificare il comando per mostrare solo le opzioni di installazione di Rocky Linux:

```
lxc image list images: | grep rocky
```

In questo modo si ottiene un elenco molto più gestibile:

```
| rockylinux/8 (3 more)                    | 0ed2f148f7c6 | yes    |
Rockylinux 8 amd64 (20220805_02:06)          | x86_64       | CONTAINER
| 128.68MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/8 (3 more)                    | 6411a033fdf1 | yes    |
Rockylinux 8 amd64 (20220805_02:06)          | x86_64       | VIRTUAL-MACHINE
| 643.15MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/8/arm64 (1 more)              | e677777306cf | yes    |
Rockylinux 8 arm64 (20220805_02:29)          | aarch64      | CONTAINER
| 124.06MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/8/cloud (1 more)              | 3d2fe303afd3 | yes    |
Rockylinux 8 amd64 (20220805_02:06)          | x86_64       | CONTAINER
| 147.04MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/8/cloud (1 more)              | 7b37619bf333 | yes    |
Rockylinux 8 amd64 (20220805_02:06)          | x86_64       | VIRTUAL-MACHINE
| 659.58MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/8/cloud/arm64                 | 21c930b2ce7d | yes    |
```

```
Rockylinux 8 arm64 (20220805_02:06)          | aarch64      | CONTAINER
| 143.17MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/9 (3 more)                     | 61b0171b7eca | yes    |
Rockylinux 9 amd64 (20220805_02:07)          | x86_64       | VIRTUAL-MACHINE
| 526.38MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/9 (3 more)                     | e7738a0e2923 | yes    |
Rockylinux 9 amd64 (20220805_02:07)          | x86_64       | CONTAINER
| 107.80MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/9/arm64 (1 more)               | 917b92a54032 | yes    |
Rockylinux 9 arm64 (20220805_02:06)          | aarch64      | CONTAINER
| 103.81MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/9/cloud (1 more)               | 16d3f18f2abb | yes    |
Rockylinux 9 amd64 (20220805_02:06)          | x86_64       | CONTAINER
| 123.52MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/9/cloud (1 more)               | 605eadf1c512 | yes    |
Rockylinux 9 amd64 (20220805_02:06)          | x86_64       | VIRTUAL-MACHINE
| 547.39MB  | Aug 5, 2022 at 12:00am (UTC)  |
| rockylinux/9/cloud/arm64                  | db3ce70718e3 | yes    |
Rockylinux 9 arm64 (20220805_02:06)          | aarch64      | CONTAINER
| 119.27MB  | Aug 5, 2022 at 12:00am (UTC)  |
```

7.2 Installare, rinominare ed elencare le immagini

Per il primo contenitore, si utilizzerà "rockylinux/8". Per installarlo, *si può* usare:

```
lxc launch images:rockylinux/8 rockylinux-test-8
```

Questo creerà un contenitore basato su Rocky Linux chiamato "rockylinux-test-8". È possibile rinominare un contenitore dopo averlo creato, ma prima è necessario arrestare il contenitore, che si avvia automaticamente alla sua creazione.

Per avviare manualmente il container, utilizzare:

```
lxc start rockylinux-test-8
```

Per rinominare l'immagine (non lo faremo qui, ma ecco come farlo), prima di tutto fermate il contenitore:

```
lxc stop rockylinux-test-8
```

Usare il comando `move` per cambiare il nome del contenitore:

```
lxc move rockylinux-test-8 rockylinux-8
```

Se avete seguito comunque questa istruzione, fermate il contenitore e riportatelo al nome originale per continuare a seguirlo.

Ai fini di questa guida, per ora installate altre due immagini:

```
lxc launch images:rockylinux/9 rockylinux-test-9
```

e

```
lxc launch images:ubuntu/22.04 ubuntu-test
```

Esaminate ciò che avete elencando le vostre immagini:

```
lxc list
```

che restituirà questo:

```
+-------------------+---------+----------------------+------+-----------
+-----------+
|       NAME        |  STATE  |         IPV4         | IPV6 |   TYPE    |
SNAPSHOTS |
+-------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-8 | RUNNING | 10.146.84.179 (eth0) |      | CONTAINER |
0         |
+-------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-9 | RUNNING | 10.146.84.180 (eth0) |      | CONTAINER |
0         |
+-------------------+---------+----------------------+------+-----------
+-----------+
| ubuntu-test       | RUNNING | 10.146.84.181 (eth0) |      | CONTAINER |
0         |
+-------------------+---------+----------------------+------+-----------
+-----------+
```

8. Capitolo 6: Profili

In tutto questo capitolo sarà necessario eseguire i comandi come utente non privilegiato ("lxdadmin" se si segue dall'inizio di questo libro).

Quando si installa LXD si ottiene un profilo predefinito, che non può essere rimosso o modificato. Detto questo, è possibile utilizzare il profilo predefinito per creare nuovi profili da utilizzare con i propri container.

Se si esamina l'elenco dei container, si noterà che l'indirizzo IP in ogni caso proviene dall'interfaccia bridged. In un ambiente di produzione, si potrebbe voler usare qualcos'altro. Potrebbe trattarsi di un indirizzo assegnato via DHCP dall'interfaccia LAN o anche di un indirizzo assegnato staticamente dalla WAN.

Se si configura il server LXD con due interfacce e si assegna a ciascuna un IP sulla WAN e sulla LAN, è possibile assegnare gli indirizzi IP del container in base all'interfaccia verso cui il container deve essere rivolto.

A partire dalla versione 9.0 di Rocky Linux (e in realtà qualsiasi copia di Red Hat Enterprise Linux con bug per bug) il metodo per assegnare gli indirizzi IP staticamente o dinamicamente con i profili non funziona.

Ci sono modi per aggirare questo problema, ma è fastidioso. Questo sembra avere a che fare con le modifiche apportate a Network Manager che influiscono su macvlan. macvlan consente di creare molte interfacce con indirizzi Layer 2 diversi.

Per il momento, è bene sapere che questo comporta degli svantaggi quando si scelgono immagini di container basate su RHEL.

8.1 Creazione di un profilo macvlan e sua assegnazione

Per creare il profilo macvlan, utilizzare questo comando:

```
lxc profile create macvlan
```

Se si dispone di una macchina con più interfacce e si desidera più di un modello macvlan in base alla rete che si desidera raggiungere, si potrebbe usare "lanmacvlan" o "wanmacvlan" o qualsiasi altro nome che si desidera usare per

identificare il profilo. L'uso di "macvlan" nella dichiarazione di creazione del profilo dipende totalmente da voi.

Si vuole cambiare l'interfaccia macvlan, ma prima è necessario sapere qual è l'interfaccia principale del nostro server LXD. Si tratta dell'interfaccia che ha un IP assegnato alla LAN (in questo caso). Per individuare l'interfaccia, utilizzare:

```
ip addr
```

Cercare l'interfaccia con l'assegnazione IP LAN nella rete 192.168.1.0/24:

```
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 40:16:7e:a9:94:85 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.106/24 brd 192.168.1.255 scope global dynamic noprefixroute
enp3s0
       valid_lft 4040sec preferred_lft 4040sec
    inet6 fe80::a308:acfb:fcb3:878f/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

In questo caso, l'interfaccia è "enp3s0".

Quindi cambiare il profilo:

```
lxc profile device add macvlan eth0 nic nictype=macvlan parent=enp3s0
```

Questo comando aggiunge al profilo macvlan tutti i parametri necessari per l'uso.

Esaminare ciò che questo comando ha creato, utilizzando il comando:

```
lxc profile show macvlan
```

Il risultato sarà simile a questo:

```
config: {}
description: ""
devices:
  eth0:
    nictype: macvlan
    parent: enp3s0
    type: nic
```

```
name: macvlan
used_by: []
```

È possibile utilizzare i profili per molte altre cose, ma l'assegnazione di un IP statico a un contenitore o l'utilizzo di un server DHCP sono esigenze comuni.

Per assegnare il profilo macvlan a rockylinux-test-8 è necessario procedere come segue:

```
lxc profile assign rockylinux-test-8 default,macvlan
```

Fare la stessa cosa per rockylinux-test-9:

```
lxc profile assign rockylinux-test-9 default,macvlan
```

Questo dice che si vuole il profilo predefinito e che si vuole applicare anche il profilo macvlan.

8.2 Rocky Linux macvlan

Nelle distribuzioni e nei cloni di RHEL, Network Manager è in costante mutamento. Per questo motivo, il modo in cui funziona il profilo `macvlan` non funziona (almeno rispetto ad altre distribuzioni) e richiede un po' di lavoro aggiuntivo per assegnare gli indirizzi IP da DHCP o staticamente.

Ricordate che tutto questo non ha nulla a che fare con Rocky Linux in particolare, ma con l'implementazione dei pacchetti upstream.

Se si desidera eseguire i container Rocky Linux e utilizzare macvlan per assegnare un indirizzo IP dalle reti LAN o WAN, il processo è diverso a seconda della versione del container del sistema operativo (8.x o 9.x).

8.2.1 Rocky Linux 9.x macvlan - la soluzione DHCP

Innanzitutto, illustriamo cosa succede quando si arrestano e si riavviano i due container dopo l'assegnazione del profilo macvlan.

L'assegnazione del profilo, tuttavia, non modifica la configurazione predefinita, che è DHCP per impostazione predefinita.

Per verificarlo, procedere come segue:

```
lxc restart rocky-test-8
lxc restart rocky-test-9
```

Elencare nuovamente i container e notare che rockylinux-test-9 non ha più un indirizzo IP:

```
lxc list
```

```
+------------------+---------+----------------------+------+-----------
+-----------+
|       NAME       |  STATE  |         IPV4         | IPV6 |   TYPE    |
SNAPSHOTS |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-8 | RUNNING | 192.168.1.114 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-9 | RUNNING |                      |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| ubuntu-test       | RUNNING | 10.146.84.181 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
```

Come si può vedere, il nostro contenitore Rocky Linux 8.x ha ricevuto l'indirizzo IP dall'interfaccia LAN, mentre il contenitore Rocky Linux 9.x no.

Per dimostrare ulteriormente il problema, è necessario eseguire `dhclient` sul contenitore Rocky Linux 9.0. Questo mostrerà che il profilo macvlan *è stato* effettivamente applicato:

```
lxc exec rockylinux-test-9 dhclient
```

Un altro elenco di container mostra ora quanto segue:

```
+------------------+---------+----------------------+------+-----------
+-----------+
```

```
|       NAME        |  STATE  |         IPV4         | IPV6 |   TYPE    |
SNAPSHOTS |
+-------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-8 | RUNNING | 192.168.1.114 (eth0) |      | CONTAINER |
0         |
+-------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-9 | RUNNING | 192.168.1.113 (eth0) |      | CONTAINER |
0         |
+-------------------+---------+----------------------+------+-----------
+-----------+
| ubuntu-test       | RUNNING | 10.146.84.181 (eth0) |      | CONTAINER |
0         |
+-------------------+---------+----------------------+------+-----------
+-----------+
```

Ciò sarebbe dovuto accadere con l'arresto e l'avvio del contenitore, ma non è così. Supponendo di voler utilizzare sempre un indirizzo IP assegnato da DHCP, si può risolvere il problema con una semplice voce di crontab. Per farlo, è necessario ottenere l'accesso al container tramite shell, inserendo:

```
lxc exec rockylinux-test-9 bash
```

Quindi, determiniamo il percorso di `dhclient`. Per fare ciò, poiché questo container proviene da un'immagine minimale, è necessario prima installare `which`:

```
dnf install which
```

quindi eseguire:

```
which dhclient
```

che restituirà:

```
/usr/sbin/dhclient
```

Successivamente, modificare il crontab di root:

```
crontab -e
```

Aggiungere questa riga:

```
@reboot    /usr/sbin/dhclient
```

Il comando crontab inserito utilizza *vi*. Per salvare le modifiche e uscire, utilizzare ⇧ Shift + : + w + q.

Uscire dal container e riavviare rockylinux-test-9:

```
lxc restart rockylinux-test-9
```

Un altro elenco rivelerà che al contenitore è stato assegnato un indirizzo DHCP:

```
+--------------------+---------+----------------------+------+-----------
+-----------+
|       NAME         |  STATE  |         IPV4         | IPV6 |   TYPE    |
SNAPSHOTS |
+--------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-8 | RUNNING | 192.168.1.114 (eth0) |      | CONTAINER |
0         |
+--------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-9 | RUNNING | 192.168.1.113 (eth0) |      | CONTAINER |
0         |
+--------------------+---------+----------------------+------+-----------
+-----------+
| ubuntu-test       | RUNNING | 10.146.84.181 (eth0) |      | CONTAINER |
0         |
+--------------------+---------+----------------------+------+-----------
+-----------+
```

8.2.2 Rocky Linux 9.x macvlan - La soluzione per l'IP statico

Per assegnare staticamente un indirizzo IP, le cose si fanno ancora più complicate. Poiché gli `network-scripts` sono ormai deprecati in Rocky Linux 9.x, l'unico modo per farlo è l'assegnazione statica e, a causa del modo in cui i container utilizzano la rete, non è possibile impostare la route con una normale istruzione di `route ip`. Il problema è che l'interfaccia assegnata quando si applica il profilo macvlan (eth0 in questo caso) non è gestibile con Network Manager. La soluzione consiste nel rinominare l'interfaccia di rete sul contenitore dopo il riavvio e assegnare l'IP

statico. È possibile farlo con uno script ed eseguirlo (di nuovo) nel crontab di root. Per farlo, utilizzare il comando `ip`.

Per farlo, è necessario ottenere nuovamente l'accesso al contenitore:

```
lxc exec rockylinux-test-9 bash
```

Successivamente, si creerà uno script bash in `/usr/local/sbin` chiamato "static":

```
vi /usr/local/sbin/static
```

Il contenuto di questo script non è difficile:

```
#!/usr/bin/env bash

/usr/sbin/ip link set dev eth0 name net0
/usr/sbin/ip addr add 192.168.1.151/24 dev net0
/usr/sbin/ip link set dev net0 up
/usr/sbin/ip route add default via 192.168.1.1
```

Cosa stiamo facendo qui?

- rinominare eth0 con un nuovo nome che possiamo gestire ("net0")
- si assegna il nuovo IP statico che abbiamo allocato per il nostro contenitore (192.168.1.151)
- si apre la nuova interfaccia "net0"
- è necessario aggiungere la route predefinita per la nostra interfaccia

Rendere il nostro script eseguibile con:

```
chmod +x /usr/local/sbin/static
```

Aggiungerlo al crontab di root per il contenitore con il @reboot time:

```
@reboot     /usr/local/sbin/static
```

Infine, uscire dal container e riavviarlo:

```
lxc restart rockylinux-test-9
```

Aspettate qualche secondo e elencate di nuovo i contenitori:

```
lxc list
```

Il successo dovrebbe essere visibile:

```
+------------------+---------+----------------------+------+-----------
+-----------+
|       NAME       |  STATE  |         IPV4         | IPV6 |   TYPE    |
SNAPSHOTS |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-8 | RUNNING | 192.168.1.114 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-9 | RUNNING | 192.168.1.151 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| ubuntu-test       | RUNNING | 10.146.84.181 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
```

8.3 Ubuntu macvlan

Fortunatamente, nell'implementazione di Ubuntu di Network Manager, lo stack macvlan NON è rotto. È molto più facile da distribuire!

Proprio come nel caso del contenitore rockylinux-test-9, è necessario assegnare il profilo al nostro contenitore:

```
lxc profile assign ubuntu-test default,macvlan
```

Per scoprire se il DHCP assegna un indirizzo al contenitore, interrompere e riavviare il contenitore:

```
lxc restart ubuntu-test
```

Elencare nuovamente i contenitori:

```
+------------------+---------+----------------------+------+-----------
+-----------+
|       NAME       |  STATE  |         IPV4         | IPV6 |   TYPE    |
SNAPSHOTS |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-8 | RUNNING | 192.168.1.114 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-9 | RUNNING | 192.168.1.151 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| ubuntu-test       | RUNNING | 192.168.1.132 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
```

Riuscito!

La configurazione dell'IP statico è leggermente diversa, ma non è affatto difficile. È necessario modificare il file .yaml associato alla connessione del contenitore (`10-lxc.yaml`). Per questo IP statico si utilizzerà 192.168.1.201:

```
vi /etc/netplan/10-lxc.yaml
```

Cambiare ciò che c'è con quanto segue:

```
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: false
      addresses: [192.168.1.201/24]
      gateway4: 192.168.1.1
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
```

Salvare le modifiche e uscire dal container.

Riavviare il container:

```
lxc restart ubuntu-test
```

Quando si elencano nuovamente i containeri, si vedrà il proprio IP statico:

```
+------------------+---------+----------------------+------+-----------
+-----------+
|       NAME       |  STATE  |         IPV4         | IPV6 |   TYPE    |
SNAPSHOTS |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-8 | RUNNING | 192.168.1.114 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| rockylinux-test-9 | RUNNING | 192.168.1.151 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
| ubuntu-test       | RUNNING | 192.168.1.201 (eth0) |      | CONTAINER |
0         |
+------------------+---------+----------------------+------+-----------
+-----------+
```

Riuscito!

Negli esempi utilizzati in questo capitolo, è stato scelto intenzionalmente un container difficile da configurare e due meno difficili. Ci sono molte altre versioni di Linux disponibili nell'elenco delle immagini. Se ne avete uno preferito, provate a installarlo, assegnate il modello macvlan e impostate gli IP.

9. Capitolo 7: Opzioni di configurazione del container

Nel corso di questo capitolo sarà necessario eseguire i comandi come utente non privilegiato ("lxdadmin" se si segue dall'inizio di questo libro).

Esistono numerose opzioni per configurare il container dopo l'installazione. Prima di vederli, però, esaminiamo il comando `info` per un container. In questo esempio, si utilizzerà il container ubuntu-test:

```
lxc info ubuntu-test
```

Verrà visualizzato quanto segue:

```
Name: ubuntu-test
Location: none
Remote: unix://
Architecture: x86_64
Created: 2021/04/26 15:14 UTC
Status: Running
Type: container
Profiles: default, macvlan
Pid: 584710
Ips:
  eth0:    inet    192.168.1.201    enp3s0
  eth0:    inet6    fe80::216:3eff:fe10:6d6d    enp3s0
  lo:    inet    127.0.0.1
  lo:    inet6    ::1
Resources:
  Processes: 13
  Disk usage:
    root: 85.30MB
  CPU usage:
    CPU usage (in seconds): 1
  Memory usage:
    Memory (current): 99.16MB
    Memory (peak): 110.90MB
  Network usage:
    eth0:
      Bytes received: 53.56kB
      Bytes sent: 2.66kB
      Packets received: 876
      Packets sent: 36
    lo:
      Bytes received: 0B
```

```
        Bytes sent: 0B
        Packets received: 0
        Packets sent: 0
```

Ci sono molte informazioni utili, dai profili applicati, alla memoria in uso, allo spazio su disco in uso e altro ancora.

9.1 Una parola sulla configurazione e su alcune opzioni

Per impostazione predefinita, LXD assegna al container la memoria di sistema, lo spazio su disco, i core della CPU e altre risorse necessarie. Ma se si vuole essere più specifici? È assolutamente possibile.

Tuttavia, questo comporta degli svantaggi. Per esempio, se si assegna la memoria di sistema e il container non la usa tutta, la si sottrae a un altro container che potrebbe averne bisogno. Tuttavia, può accadere anche il contrario. Se un container vuole usare più della sua quota di memoria, può impedire agli altri container di averne a sufficienza, riducendo così le loro prestazioni.

Ricordate che ogni azione compiuta per configurare un contenitore *può* avere effetti negativi da qualche altra parte.

Piuttosto che scorrere tutte le opzioni di configurazione, utilizzare il completamento automatico delle schede per visualizzare le opzioni disponibili:

```
lxc config set ubuntu-test
```

e Tab ⇥.

Mostra tutte le opzioni per la configurazione di un container. Se avete domande su cosa fa una delle opzioni di configurazione, visitate la documentazione ufficiale di LXD e fate una ricerca per il parametro di configurazione, oppure cercate su Google l'intera stringa, ad esempio `lxc config set limits.memory` ed esaminate i risultati della ricerca.

Di seguito esaminiamo alcune delle opzioni di configurazione più utilizzate. Ad esempio, se si vuole impostare la quantità massima di memoria che un container può utilizzare:

```
lxc config set ubuntu-test limits.memory 2GB
```

Questo dice che se la memoria è disponibile per l'uso, ad esempio se ci sono 2 GB di memoria disponibile, il container può effettivamente usare più di 2 GB se è disponibile. Si tratta di un limite morbido, ad esempio.

```
lxc config set ubuntu-test limits.memory.enforce 2GB
```

Ciò significa che il container non può mai utilizzare più di 2 GB di memoria, indipendentemente dal fatto che sia attualmente disponibile o meno. In questo caso si tratta di un limite rigido.

```
lxc config set ubuntu-test limits.cpu 2
```

Questo dice di limitare a 2 il numero di core della CPU che il container può utilizzare.

Nota

Quando questo documento è stato riscritto per Rocky Linux 9.0, il repository ZFS per il 9 non era disponibile. Per questo motivo tutti i nostri contenitori di prova sono stati costruiti usando "dir" nel processo di init. Per questo motivo l'esempio seguente mostra un pool di archiviazione "dir" invece di "zfs".

Ricordate quando avete impostato il nostro pool di archiviazione nel capitolo ZFS? Il pool è stato chiamato "storage", ma avrebbe potuto essere chiamato in qualsiasi modo. Se si desidera esaminarlo, si può usare questo comando, che funziona ugualmente bene anche per gli altri tipi di pool (come mostrato per dir):

```
lxc storage show storage
```

Questo mostra quanto segue:

```
config:
  source: /var/snap/lxd/common/lxd/storage-pools/storage
description: ""
name: storage
driver: dir
used_by:
- /1.0/instances/rockylinux-test-8
- /1.0/instances/rockylinux-test-9
- /1.0/instances/ubuntu-test
```

```
- /1.0/profiles/default
status: Created
locations:
- none
```

Questo mostra che tutti i container utilizzano il pool di archiviazione dir. Quando si usa ZFS, si può anche impostare una quota disco su un container. Ecco come appare il comando, che imposta una quota disco di 2 GB sul container ubuntu-test:

```
lxc config device override ubuntu-test root size=2GB
```

Come detto in precedenza, usare le opzioni di configurazione con parsimonia, a meno che non si abbia un container che vuole usare molto più della sua quota di risorse. LXD, nella maggior parte dei casi, gestisce l'ambiente in modo autonomo.

Esistono molte altre opzioni che potrebbero essere di interesse per alcuni. La ricerca personale vi aiuterà a scoprire se uno di questi elementi è utile nel vostro ambiente.

10. Capitolo 8: istantanee del contenitore

Nel corso di questo capitolo sarà necessario eseguire i comandi come utente non privilegiato ("lxdadmin" se si è seguito dall'inizio di questo libro).

Le istantanee dei container, insieme a un server di istantanee (di cui si parlerà più avanti), sono probabilmente l'aspetto più importante dell'esecuzione di un server LXD di produzione. Le istantanee garantiscono un ripristino rapido. È una buona idea usarli come sicurezza quando si aggiorna il software principale che gira su un particolare contenitore. Se durante l'aggiornamento accade qualcosa che interrompe l'applicazione, è sufficiente ripristinare l'istantanea per tornare operativi con un tempo di inattività di pochi secondi.

L'autore ha utilizzato i container LXD per i server PowerDNS rivolti al pubblico e il processo di aggiornamento di queste applicazioni è diventato meno preoccupante, grazie alla creazione di istantanee prima di ogni aggiornamento.

È possibile eseguire lo snapshot di un contenitore anche quando è in esecuzione.

10.1 Il processo di snapshot

Iniziate ottenendo un'istantanea del container ubuntu-test con questo comando:

```
lxc snapshot ubuntu-test ubuntu-test-1
```

Qui lo snapshot viene chiamato "ubuntu-test-1", ma si può chiamare in qualsiasi modo. Per assicurarsi di uno snapshot, eseguire un'`lxc info` del container:

```
lxc info ubuntu-test
```

Avete già guardato una schermata informativa. Se si scorre fino in fondo, ora si vede:

```
Snapshots:
  ubuntu-test-1 (taken at 2021/04/29 15:57 UTC) (stateless)
```

Riuscito! Il nostro snapshot è in posizione.

Entrate nel container ubuntu-test:

```
lxc exec ubuntu-test bash
```

Creare un file vuoto con il comando *touch*:

```
touch this_file.txt
```

Uscite dal container.

Prima di ripristinare il container com'era prima della creazione del file, il modo più sicuro per ripristinare un container, in particolare se ci sono state molte modifiche, è quello di fermarlo prima:

```
lxc stop ubuntu-test
```

Ripristino:

```
lxc restore ubuntu-test ubuntu-test-1
```

Avviare nuovamente il container:

```
lxc start ubuntu-test
```

Se si torna di nuovo nel container e si guarda, il "this_file.txt" creato non c'è più.

Quando non si ha più bisogno di uno snapshot, è possibile eliminarlo:

```
lxc delete ubuntu-test/ubuntu-test-1
```

Attenzione

È sempre consigliabile eliminare gli snapshot con il container in esecuzione. Perché? Il comando *lxc delete* funziona anche per eliminare l'intero contenitore. Se avessimo accidentalmente premuto invio dopo "ubuntu-test" nel comando precedente, E, se il container fosse stato fermato, il container sarebbe stato cancellato. Non viene dato alcun avviso, fa semplicemente quello che gli si chiede.

Se il container è in esecuzione, tuttavia, viene visualizzato questo messaggio:

```
Error: The instance is currently running, stop it first or pass --force
```

Pertanto, eliminare sempre gli snapshot con il container in funzione.

Nei capitoli che seguono, vi saranno:

- impostare il processo di creazione automatica degli snapshot
- impostare la scadenza di uno snapshot in modo che scompaia dopo un certo periodo di tempo
- impostare l'aggiornamento automatico degli snapshot al server snapshot

11. Capitolo 9: server snapshot

In questo capitolo si utilizza una combinazione di utenti privilegiati (root) e non privilegiati (lxdadmin), a seconda dei compiti che si stanno eseguendo.

Come indicato all'inizio, il server snapshot per LXD deve essere in tutto e per tutto un mirror del server di produzione. Il motivo è che potrebbe essere necessario passare alla produzione in caso di guasto dell'hardware del server principale, e disporre di backup e di un modo rapido per riavviare i container di produzione consente di ridurre al minimo le telefonate e i messaggi di panico dell'amministratore di sistema. QUESTO è SEMPRE un bene!

Il processo di creazione del server snapshot è identico a quello del server di produzione. Per emulare completamente la nostra configurazione del server di produzione, eseguite nuovamente tutti i **capitoli da 1 a 4** sul server snapshot e, una volta completati, tornate a questo punto.

Sei tornato!!! Congratulazioni, questo significa che l'installazione di base del server snapshot è stata completata con successo.

11.1 Impostazione del rapporto tra server primario e snapshot

Prima di continuare è necessario fare un po' di pulizia. Innanzitutto, se si opera in un ambiente di produzione, probabilmente si ha accesso a un server DNS che si può utilizzare per impostare la risoluzione IP-nome.

Nel nostro laboratorio non abbiamo questo lusso. Forse anche voi avete lo stesso scenario. Per questo motivo, si aggiungeranno gli indirizzi IP e i nomi di entrambi i server al file /etc/hosts sul server primario e sul server snapshot. È necessario farlo come utente root (o *sudo*).

Nel nostro laboratorio, il server LXD primario è in esecuzione su 192.168.1.106 e il server LXD snapshot è in esecuzione su 192.168.1.141. Accedere a ciascun server tramite SSH e aggiungere quanto segue al file /etc/hosts:

```
192.168.1.106 lxd-primary
192.168.1.141 lxd-snapshot
```

Successivamente, è necessario consentire tutto il traffico tra i due server. A tale scopo, occorre modificare le regole di `firewalld`. Per prima cosa, sul server lxd-primario, aggiungere questa riga:

```
firewall-cmd zone=trusted add-source=192.168.1.141 --permanent
```

e sul server snapshot aggiungere questa regola:

```
firewall-cmd zone=trusted add-source=192.168.1.106 --permanent
```

poi ricaricare:

```
firewall-cmd reload
```

Successivamente, come utente non privilegiato (lxdadmin), è necessario impostare la relazione di fiducia tra le due macchine. Per farlo, eseguire il seguente comando su lxd-primary:

```
lxc remote add lxd-snapshot
```

Visualizza il certificato da accettare. Accettate e vi verrà richiesta la password. Si tratta della "trust password" impostata durante la fase di inizializzazione di LXD. Si spera che teniate traccia di tutte queste password in modo sicuro. Quando si inserisce la password, si riceve questo messaggio:

```
Client certificate stored at server:  lxd-snapshot
```

Non fa male avere anche la retromarcia. Ad esempio, impostare la relazione di fiducia anche sul server lxd-snapshot. In questo modo, se necessario, il server lxd-snapshot può inviare le istantanee al server lxd-primario. Ripetere i passaggi e sostituire "lxd-primary" con "lxd-snapshot"

11.1.1 Migrazione del primo snapshot

Prima di poter migrare il primo snapshot, è necessario che su lxd-snapshot vengano creati i profili che sono stati creati su lxd-primary. Nel nostro caso, si tratta del profilo "macvlan".

È necessario crearlo per lxd-snapshot. Tornare al Capitolo 6 e creare il profilo "macvlan" su lxd-snapshot, se necessario. Se i due server hanno gli stessi nomi di interfaccia madre ("enp3s0", ad esempio), è possibile copiare il profilo "macvlan" in lxd-snapshot senza ricrearlo:

```
lxc profile copy macvlan lxd-snapshot
```

Dopo aver impostato tutte le relazioni e i profili, il passo successivo è l'invio effettivo snapshot da lxd-primary a lxd-snapshot. Se avete seguito esattamente la procedura, probabilmente avete cancellato tutti i vostri snapshot. Create un'altro snapshot:

```
lxc snapshot rockylinux-test-9 rockylinux-test-9-snap1
```

Se si esegue il comando "info" per `lxc`, si può vedere l'istantanea in fondo al nostro elenco:

```
lxc info rockylinux-test-9
```

Che mostrerà qualcosa di simile in basso:

```
rockylinux-test-9-snap1 at 2021/05/13 16:34 UTC) (stateless)
```

Ok, incrociamo le dita! Proviamo a migrare in nostro snapshot:

```
lxc copy rockylinux-test-9/rockylinux-test-9-snap1 lxd-snapshot:rockylinux-
test-9
```

Questo comando dice che, all'interno del container rockylinux-test-9, si vuole inviare lo snapshot rockylinux-test-9-snap1 a lxd-snapshot e chiamarla rockylinux-test-9.

Dopo poco tempo, la copia sarà completa. Volete scoprirlo con certezza? Eseguire un `lxc list` sul server lxd-snapshot. Che dovrebbe restituire quanto segue:

```
+-------------------+---------+------+------+-----------+-----------+
|    NAME           |  STATE  | IPV4 | IPV6 |   TYPE    | SNAPSHOTS |
+-------------------+---------+------+------+-----------+-----------+
```

```
| rockylinux-test-9 | STOPPED |      |      | CONTAINER | 0         |
+-------------------+---------+------+------+-----------+-----------+
```

Riuscito! Provate ad avviarlo. Poiché lo stiamo avviando sul server lxd-snapshot, è necessario fermarlo prima sul server lxd-primario per evitare un conflitto di indirizzi IP:

```
lxc stop rockylinux-test-9
```

E sul server lxd-snapshot:

```
lxc start rockylinux-test-9
```

Supponendo che tutto questo funzioni senza errori, arrestare il container su lxd-snapshot e riavviarlo su lxd-primary.

11.2 Impostazione di boot.autostart su off per i container

Gli snapshot copiate su lxd-snapshot saranno inattivi durante la migrazione, ma se si verifica un evento di alimentazione o se è necessario riavviare il server di snapshot a causa di aggiornamenti o altro, si verificherà un problema. Questi container cercheranno di avviarsi sul server snapshot creando un potenziale conflitto di indirizzi IP.

Per eliminare questo problema, è necessario impostare i container migrati in modo che non vengano avviati al riavvio del server. Per il nostro container rockylinux-test-9 appena copiato, lo si farà con:

```
lxc config set rockylinux-test-9 boot.autostart 0
```

Eseguire questa operazione per ogni snapshot sul server lxd-snapshot. Lo "0" nel comando assicura che `boot.autostart` sia disattivato.

11.3 Automatizzazione del processo di snapshot

È fantastico poter creare snapshot quando è necessario, ma a volte *è* necessario creare manualmente uno snapshot. Si potrebbe anche copiare manualmente su lxd-snapshot. Ma per tutte le altre volte, in particolare per molti container in

esecuzione sul server lxd-primario, l'**ultima** cosa da fare è passare un pomeriggio a cancellare le istantanee sul server snapshot, creare nuove istantanee e inviarle al server snapshot. Per la maggior parte delle operazioni, è preferibile automatizzare il processo.

La prima cosa da fare è pianificare un processo per automatizzare la creazione di snapshot su lxd-primary. Questa operazione viene eseguita per ogni container sul server lxd-primary. Una volta completata, si occuperà di questo aspetto in futuro. Per farlo, si utilizza la seguente sintassi. Si noti la somiglianza con una voce di crontab per il timestamp:

```
lxc config set [container_name] snapshots.schedule "50 20 * * *"
```

Ciò significa che bisogna eseguire un'istantanea del nome del container ogni giorno alle 20:50.

Per applicare questo al nostro container rockylinux-test-9:

```
lxc config set rockylinux-test-9 snapshots.schedule "50 20 * * *"
```

È inoltre necessario impostare il nome dello snapshot in modo che sia significativo per la nostra data. LXD utilizza ovunque UTC, quindi la cosa migliore per tenere traccia delle cose è impostare il nome del container con una data e un'ora in un formato più comprensibile:

```
lxc config set rockylinux-test-9 snapshots.pattern "rockylinux-
test-9{{ creation_date|date:'2006-01-02_15-04-05' }}"
```

GRANDE, ma di certo non volete una nuova istantanea ogni giorno senza sbarazzarvi di quella vecchia, giusto? L'unità si riempirebbe di istantanee. Per risolvere questo problema si esegue:

```
lxc config set rockylinux-test-9 snapshots.expiry 1d
```

12. Capitolo 10: Automatizzare

Durante tutto questo capitolo dovrai essere radice o in grado di `sudo` per diventare root.

L'automazione del processo di snapshot rende le cose molto più facili.

12.1 Automatizzazione del Processo di Copia di Istantanee

Questo processo viene eseguito su lxd-primario. La prima cosa da fare è creare uno script che verrà eseguito da cron in /usr/local/sbin chiamato "refresh-containers" :

```
sudo vi /usr/local/sbin/refreshcontainers.sh
```

Lo script è piuttosto semplice:

```
#!/bin/bash
# This script is for doing an lxc copy --refresh against each container,
copying
# and updating them to the snapshot server.

for x in $(/var/lib/snapd/snap/bin/lxc ls -c n --format csv)
        do echo "Refreshing $x"
        /var/lib/snapd/snap/bin/lxc copy --refresh $x lxd-snapshot:$x
        done
```

Rendetelo eseguibile:

```
sudo chmod +x /usr/local/sbin/refreshcontainers.sh
```

Cambiare la proprietà di questo script all'utente e al gruppo lxdadmin:

```
sudo chown lxdadmin.lxdadmin /usr/local/sbin/refreshcontainers.sh
```

Impostare il crontab per l'utente lxdadmin per l'esecuzione di questo script, in questo caso alle 10 di sera:

```
crontab -e
```

La voce avrà il seguente aspetto:

```
00 22 * * * /usr/local/sbin/refreshcontainers.sh > /home/lxdadmin/refreshlog
2>&1
```

Salvare le modifiche e uscire.

In questo modo si creerà un registro nella home directory di lxdadmin chiamato "refreshlog" che permetterà di sapere se il processo ha funzionato o meno. Molto importante!

La procedura automatica a volte fallisce. Questo accade generalmente quando un particolare container non riesce ad aggiornarsi. È possibile eseguire manualmente l'aggiornamento con il seguente comando (assumendo rockylinux-test-9 qui, come nostro contenitore):

```
lxc copy --refresh rockylinux-test-9 lxd-snapshot:rockylinux-test-9
```

13. Appendice A - Configurazione Workstation

Anche se non fa parte dei capitoli per un server LXD, questa procedura aiuterà coloro che vogliono avere un ambiente di laboratorio, o un sistema operativo e un'applicazione semi-permanente, in esecuzione su una workstation o un notebook Rocky Linux.

13.1 Prerequisiti

- a proprio agio alla riga di comando
- in grado di utilizzare correntemente un editor a riga di comando, come `vi` o `nano`
- disposti a installare `snapd` per installare LXD
- necessità di un ambiente di test stabile da utilizzare quotidianamente o secondo necessità
- in grado di diventare root o di avere privilegi `sudo`

13.2 Installazione

Dalla riga di comando, installare il repository EPEL:

```
sudo dnf install epel-release
```

Al termine dell'installazione, eseguire l'aggiornamento:

```
sudo dnf upgrade
```

Installare `snapd`

```
sudo dnf install snapd
```

Abilitare il servizio `snapd`

```
sudo systemctl enable snapd
```

Riavviare il notebook o la workstation

Installare lo snap per LXD:

```
sudo snap install lxd
```

13.3 Inizializzazione di LXD

Se avete letto i capitoli sul server di produzione, questa procedura è quasi identica alla procedura di avvio del server di produzione.

```
sudo lxd init
```

Si aprirà una finestra di dialogo con domande e risposte.

Ecco le domande e le nostre risposte per lo script, con una piccola spiegazione dove necessario:

```
Would you like to use LXD clustering? (yes/no) [default=no]:
```

Se siete interessati al clustering, fate ulteriori ricerche su Linux container qui.

```
Do you want to configure a new storage pool? (yes/no) [default=yes]:
Name of the new storage pool [default=default]: storage
```

Facoltativamente, è possibile accettare l'impostazione predefinita.

```
Name of the storage backend to use (btrfs, dir, lvm, ceph) [default=btrfs]: dir
```

Si noti che `dir` è leggermente più lento di `btrfs`. Se si ha l'accortezza di lasciare un disco vuoto, si può usare quel dispositivo (ad esempio: /dev/sdb) per il dispositivo `btrfs` e poi selezionare `btrfs`, ma solo se il computer host ha un sistema operativo che supporta `btrfs`. Rocky Linux e qualsiasi clone di RHEL non supporteranno `btrfs` - non ancora, comunque. la `dir` funzionerà bene in un ambiente di laboratorio.

```
Would you like to connect to a MAAS server? (yes/no) [default=no]:
```

Il Metal As A Service (MAAS) non rientra nell'ambito di questo documento.

```
Would you like to create a new local network bridge? (yes/no) [default=yes]:
What should the new bridge be called? [default=lxdbr0]:
What IPv4 address should be used? (CIDR subnet notation, “auto” or “none”)
[default=auto]:
What IPv6 address should be used? (CIDR subnet notation, “auto” or “none”)
[default=auto]: none
```

Se si desidera utilizzare IPv6 sui propri contenitori LXD, è possibile attivare questa opzione. Questo dipende da voi.

```
Would you like the LXD server to be available over the network? (yes/no)
[default=no]: yes
```

Questa operazione è necessaria per eseguire lo snapshot della workstation. Rispondere "sì" in questo caso.

```
Address to bind LXD to (not including port) [default=all]:
Port to bind LXD to [default=8443]:
Trust password for new clients:
Again:
```

Questa password di fiducia è il modo in cui ci si connette al server snapshot o si torna indietro dal server snapshot. Impostate qualcosa che abbia senso nel vostro ambiente. Salvare questa voce in un luogo sicuro, ad esempio in un gestore di password.

```
Would you like stale cached images to be updated automatically? (yes/no)
[default=yes]
Would you like a YAML "lxd init" preseed to be printed? (yes/no) [default=no]:
```

13.4 Privilegi dell'utente

La prossima cosa da fare è aggiungere l'utente al gruppo lxd. Anche in questo caso, è necessario usare `sudo` o essere root:

```
sudo usermod -a -G lxd [username]
```

dove [username] è l'utente del sistema.

A questo punto, sono state apportate diverse modifiche. Prima di proseguire, riavviare il computer.

13.5 Verifica dell'installazione

Per assicurarsi che `lxd` sia stato avviato e che il vostro utente abbia i privilegi, dal prompt della shell fate:

```
lxc list
```

Si noti che qui non è stato usato `sudo`. L'utente ha la possibilità di inserire questi comandi. Vedrete qualcosa di simile a questo:

```
+------------+---------+----------------------+------+-----------+-----------+
|    NAME    |  STATE  |         IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+------------+---------+----------------------+------+-----------+-----------+
```

Se lo fate, avete un bell'aspetto!

13.6 Il resto

Da questo punto, si possono facilmente utilizzare i capitoli del nostro "LXD Production Server" per proseguire. Tuttavia, ci sono alcuni aspetti della configurazione di una workstation a cui dobbiamo prestare meno attenzione. Ecco i capitoli consigliati per iniziare:

- Capitolo 5 - Impostazione e gestione delle immagini
- Capitolo 6 - Profili
- Chapter 8 - Container Snapshots

13.7 Ulteriori informazioni

- Guida per principianti di LXD che vi permetterà di iniziare a usare LXD in modo produttivo.
- Panoramica e documentazione ufficiale di LXD

13.8 Conclusione

LXD è uno strumento potente che può essere utilizzato su workstation o server per aumentare la produttività. Su una workstation, è ottimo per i test di laboratorio, ma può anche mantenere snapshot semi-permanenti di sistemi operativi e applicazioni disponibili nel loro spazio privato.