Virtualization with Linux Containers (LXC)

SUSE Linux Enterprise Server 11 SP2

SUSE Documentation Team

LXC is a lightweight “virtualization” method to run multiple virtual units (containers, akin to “chroot”) simultaneously on a single control host. Containers are sufficiently isolated with Kernel Control Groups (cgroups) to guarantee the required security.

LXC provides an operating system-level virtualization where the *Kernel* controls the isolated containers. With other full virtualization solutions like Xen, KVM, or `libvirt` the *processor* simulates a complete hardware environment and controls its virtual machines.

1 Terminology

chroot
: A *change root* (chroot, or change root jail) is a section in the filesystem which is isolated from the rest of the filesystem. For this purpose, the `chroot` command is used to change the root of the filesystem. A program which is executed in such a “chroot jail” cannot access files outside the designated directory tree.

cgroups
: Kernel Control Groups (commonly referred to as just “cgroups”) are a Kernel feature that allows aggregating or partitioning tasks (processes) and all their children into hierarchical organized groups to isolate resources.

Container
: A “virtual machine” on the host server that can run any Linux system, for example openSUSE, SUSE Linux Enterprise Desktop, or SUSE Linux Enterprise Server.

Container Name
: A name that refers to a container. The name is used by the `lxc` commands.

Kernel Namespaces
: A Kernel feature to isolate some resources like network, users, and others for a group of processes.

LXC Host Server
: The system that contains the LXC system and provides the containers and management control capabilities through cgroups.

2 Overview

Conceptually, LXC can be seen as an improved *chroot* technique. The difference is that a chroot environment separates only the file system, whereas LXC goes further and provides resource management and control via cgroups.

Benefits of LXC

- Isolating applications and operating systems through containers.
- Providing nearly native performance as LXC manages allocation of resources in real-time.
- Controlling network interfaces and applying resources inside containers through cgroups.

Limitations of LXC

- All LXC containers are running inside the host system's Kernel and not with a different Kernel.
- Only allows Linux “guest” operating systems.
- LXC is not a full virtualization stack like Xen, KVM, or `libvirt`.
- Security depends on the host system. LXC is not secure. If you need a secure system, use KVM.

3 Setting up an LXC Host

The LXC host provides the cgroups and controls all containers.

Procedure 1 *Preparing an LXC Host*

1 Install the following packages:

- lxc
- bridge-utils

2 Check if everything is prepared for LXC:

```
lxc-checkconfig
```

You should see the words `enabled` on each checked item.

3 If you want to access the virtual container's ethernet interface, create a network bridge. A network bridge allows to share the network link on the physical interface of the host (`eth0`):

3a Open YaST and go to *Network Device* > *Network Settings*.

3b Click *Add*.

3c Select *Bridge* as device type. Proceed with *Next*.

3d Activate *Dynamic Address* and select *DHCP*.

3e Choose your bridged device(s), usually `eth0`. Proceed with *Next*. Optionally check your devices with the `ifconfig` command. Close the *Network Settings* module.

4 If you have created a network bridge, assign its interface zone:

4a Start YaST and go to *Security & Users* > *Firewall*.

4b Open the *Interfaces* tab.

4c Select your bridge device (usually `br0`).

4d Click *Change...* and select *External Zone*. Proceed with *Ok*.

4e Finish with *Next*.

5 Make sure the cgroup service is started by running

```
/etc/init.d/boot.cgroup start
```

In order to automatically activate cgroups at boot time run the following command:

```
insserv boot.cgroup
```

The LXC host is now prepared for setting up containers.

4 Setting up LXC Containers with YaST

A container is a “virtual machine” that can be started, stopped, connected, or disconnected in YaST. The two last actions are only available in the GUI version, not when YaST running in text mode. If you use YaST in a text console, use the `lxc-console` command as described in Procedure 5, “Starting, Accessing, and Stopping Your Container Manually” (page 8).

To set up an LXC container with YaST, proceed as follows:

Procedure 2 *Creating a Container with YaST*

1 Open YaST and go to the LXC module.

2 Click *Create*.

3 Enter a name of your container in the *Name* field.

4 Select a Linux distribution (only SLES is supported) from the *Template* popup menu.

5 Enter the bridge for your LXC container. If you do not have a bridge, click *Configure Network...* to configure a bridge.

6 If needed, enter a password to log in to a LXC container. If you leave the password field empty, the standard password “root” is used for this container.

7 Finish with *Create* and YaST tries to prepare the container. This action takes some time.

8 After YaST has finished the preparation, click *Start* to launch the LXC container.

Procedure 3 *Starting, Accessing, and Stopping Your Container with YaST*

1 Select the container and click *Start*

2 Click the *Connect* button. A new terminal window opens.

3 Log in with user `root` and your password from Step 6 (page 5) of Procedure 2, "Creating a Container with YaST" (page 5). If you did not set a password, use "root".

4 Make your changes in your container.

5 When you are finished, save all your work and log out.

6 Click the *Disconnect* button to close the terminal. It is still possible to reconnect to your container by clicking *Connect*.

7 To shutdown the container entirely, click the *Stop* button.

5 Setting up LXC Containers Manually

A container is a "virtual machine" that can be started, stopped, frozen, or cloned (to name but a few tasks). To set up an LXC container, proceed as follows:

Procedure 4 *Creating a Container Manually*

1 Create a configuration file (name `lxc_vps0.conf` in this example) with the container name in it and edit it according to the following example:

```
lxc.utsname = vps0 ❶
lxc.network.type = veth ❷
lxc.network.flags = up ❸
lxc.network.link = br0 ❹
lxc.network.hwaddr = 00:30:6E:08:EC:80 ❺
lxc.network.ipv4 = 192.168.1.10 ❻
lxc.network.name = eth0 ❼
```

❶ Container name, should also be used when naming the configuration file

❷ Type of network virtualization to be used for the container. The option `veth` defines a peer network device. It is created with one side assigned to the container and the other side is attached to a bridge by the `lxc.network.link` option.

❸ Network actions. The value `up` in this case activates the network.

❹ Host network interface to be used for the container.

❺ Allocated MAC address of the virtual interface. This MAC address needs to be unique in your network and different from the host MAC address.

❻ IPv4 address assigned to the virtualized interface. Use the address `0.0.0.0` to make use of DHCP. Use `lxc.network.ipv6` if you need IPv6 support.

❼ Dynamically allocated interface name. This option will rename the interface in the container.

More example files can be found in `/usr/share/doc/packages/lxc/examples/`. Find details about all options in the `lxc.conf` man page.

2 Create a container by using the configuration file from Step 1 (page 6):

- For an openSUSE 12.1 "guest":

```
lxc-create -t opensuse -f lxc.conf -n CONTAINER
```

- For a SUSE Linux Enterprise Server 11 SP2 "guest":

```
lxc-create -t sles -f lxc.conf -n CONTAINER
```

CONTAINER needs to be replaced by the value you specified for `lxc.utsname` in the config file, `vps0`in this example.

Downloading and installing the base packages for openSUSE or SUSE Linux Enterprise Server will take some time. The container will be created in `/var/lib/lxc/`*CONTAINER*.

3 Finalize the configuration of the container:

3a Change the root path to the installed LXC container with the `chroot` command:

```
chroot /var/lib/lxc/CONTAINER_NAME/rootfs/
```

3b Change the password for user `root` with `passwd root`.

3c Create an `operator` user without `root` privileges:

```
useradd -m operator
```

3d Change the operator's password:

```
passwd operator
```

3e Leave the chroot environment with `exit`.

Procedure 5 *Starting, Accessing, and Stopping Your Container Manually*

1 Start the container:

```
lxc-start -n CONTAINER_NAME
```

2 Connect to the container and log in:

```
lxc-console -n CONTAINER_NAME
```

3 Stop and remove your container *always* with the two steps:

```
lxc-stop -n CONTAINER_NAME
lxc-destroy -n CONTAINER_NAME
```

6 Starting Containers at Boot Time

LXC containers can be started at boot time. However, you need to follow certain conventions. Every container has a subdirectory with its name in `/etc/lxc/`, for example, `/etc/lxc/my-sles`. This directory needs to be created once. There you place your configuration file (named `config`).

To set up the automatic start of LXC containers, proceed as follows:

1 Activate the cgroup service with `insserv boot.cgroup`. This has to be done only once to enable this service at boot time. The command will populate the `/sys/fs/cgroup` directory.

2 Create a directory `/etc/lxc/`*`CONTAINER`*.

3 Copy your configuration file to `/etc/lxc/`*`CONTAINER`*`/config`.

4 Run `/etc/init.d/boot.cgroup start` to set up cgroups properly.

5 Run `/etc/init.d/lxc start` to start your containers.

6 Wait a few seconds and run `/etc/init.d/lxc list` to print the state of all your containers.

After this procedure, your LXC containers are correctly configured. To start it automatically next time you boot your computer, use `insserv lxc`.

7 For More Information

LXC Home Page
http://lxc.sourceforge.net

Kernel Control Groups (cgroups)
http://www.suse.com/documentation/sles11/book_sle_tuning/data/cha_tuning_cgroups.html

Managing Virtual Machines with libvirt
http://www.suse.com/documentation/sles11/book_sles_kvm/data/part_managing_virtual.html

LXC Container Driver
http://libvirt.org/drvlxc.html

8 Legal Notice

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither Novell, Inc., SUSE LINUX Products GmbH, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.