Linux Containers with LXC/LXD

Unlike VMWare/Virtualbox virtualisation, containers wraps up individual workloads (instead of the entire OS and kernel) and their dependencies into relatively tiny containers (or “jails” if youre talking FreeNAS/AIX, or “zones” if you’re talking Solaris, “snaps” if you’re talking Ubuntu Core). There are many solutions to linux containerisation in the marketplace at present, and LXC is free.

This post serves as a go-to reference page for examples of all the most commonly used lxc commands when dealing with linux containers. I highly recommend completing all the sections below, running all the commands on the test server available at linuxcontainers.org, to fully appreciate the context of what you are doing. This post is a work in progress and will likely be augmented over time with examples from my own lab, time permitting.

Your first container

LXD is image based, however by default no images are loaded into the image store as can be seen with:

lxc image list

LXD knows about 3 default image servers:

ubuntu: (for Ubuntu stable images)
ubuntu-daily: (for Ubuntu daily images)
images: (for a bunch of other distributions)

The stable Ubuntu images can be listed with:

lxc image list ubuntu: | less

To launch a first container called “first” using the Ubuntu 16.04 image, use:

lxc launch ubuntu:16.04 first

Your new container will now be visible in:

lxc list

Running state details and configuration can be queried with:

lxc info first
lxc config show first

Limiting resources

By default your container comes with no resource limitation and inherits from its parent environment. You can confirm it with:

free -m
lxc exec first – free -m

To apply a memory limit to your container, do:

lxc config set first limits.memory 128MB

And confirm that it’s been applied with:

lxc exec first – free -m

Snapshots

LXD supports snapshoting and restoring container snapshots. Before making a snapshot, lets do some changes to the container, for example, updating it:

lxc exec first – apt-get update
lxc exec first – apt-get dist-upgrade -y
lxc exec first – apt-get autoremove –purge -y

Now that the container is all updated and cleaned, let’s make a snapshot called “clean”:

lxc snapshot first clean

Let’s break our container:

lxc exec first — rm -Rf /etc /usr

Confirm the breakage with (then exit):

lxc exec first — bash

And restore everything to the snapshotted state: (be sure to execute these from the container host, not from inside the container or it won’t work.

lxc restore first clean

And confirm everything’s back to normal (then exit):

lxc exec first — bash

Creating images

As your probably noticed earlier, LXD is image based, that is, all containers must be created from either a copy of an existing container or from an image.

You can create new images from an existing container or a container snapshot.

To publish our “clean” snapshot from earlier as a new image with a user friendly alias of “clean-ubuntu”, run:

lxc publish first/clean —alias clean-ubuntu

At which point we won’t need our “first” container, so just delete it with:

lxc stop first
lxc delete first

And lastly we can start a new container from our image with:

lxc launch clean-ubuntu second

Accessing files from the container

To pull a file from the container you can use the “lxc file pull” command:

lxc file pull second/etc/hosts .

Let’s add an entry to it:

echo “1.2.3.4 my-example” >> hosts

And push it back where it came from:

lxc file push hosts second/etc/hosts

You can also use this mechanism to access log files:

lxc file pull second/var/log/syslog – | less

We won’t be needing that container anymore, so stop and delete it with:

lxc delete –force second

Use a remote image server

The lxc client tool supports multiple “remotes”, those remotes can be read-only image servers or other LXD hosts.

LXC upstream runs one such server at https://images.linuxcontainers.org which serves a set of automatically generated images for various Linux

distributions.

It comes pre-added with default LXD but you can remove it or change it if you don’t want it.

You can list the available images with:

lxc image list images: | less

And spawn a new Centos 7 container with:

lxc launch images:centos/7 third

Confirm it’s indeed Centos 7 with:

lxc exec third — cat /etc/redhat-release

And delete it:

lxc delete -f third

The list of all configured remotes can be obtained with:

lxc remote list

Interact with remote LXD servers

For this step, you’ll need a second demo session, so open a new one here

Copy/paste the “*lxc remote add*” command from the top of the page of that new session into the shell of your old session. Then confirm the server fingerprint for the remote server.

Note that it may take a few seconds for the new LXD daemon to listen to the network, just retry the command until it answers.

At this point you can list the remote containers with:

lxc list tryit:

And its images with:

lxc image list tryit:

Now, let’s start a new container on the remote LXD using the local image we created earlier.

lxc launch clean-ubuntu tryit:fourth

You now have a container called “fourth” running on the remote host “tryit”. You can spawn a shell inside it with (then exit):

lxc exec tryit:fourth bash

Now let’s copy that container into a new one called “fifth”:

lxc copy tryit:fourth tryit:fifth

And just for fun, move it back to our local lxd while renaming it to “sixth”:

lxc move tryit:fifth sixth

And confirm it’s all still working (then exit):

lxc start sixth
lxc exec sixth — bash

Then clean everything up:

lxc delete -f sixth
lxc delete -f tryit:fourth
lxc image delete clean-ubuntu