Comparing LXD vs. LXC

allquixotic (Sean McNamara) #1 April 25, 2017, 7:33pm

An **extremely common confusion** is the distinction between **LXD** ("LX Daemon" / pronounced "lex-dee") and **LXC** (Linux Containers). This post attempts to clear that up.

Overview

From the **official site**:

> LXD isn't a rewrite of LXC, in fact it's building on top of LXC to provide a new, better user experience. Under the hood, LXD uses LXC through liblxc and its Go binding to create and manage the containers.

> It's basically an alternative to LXC's tools and distribution template system with the added features that come from being controllable over the network.

And from **here**:

> LXD is the new LXC experience. It offers a completely fresh and intuitive user experience with a single command line tool to manage your containers. Containers can be managed over the network in a transparent way through a REST API. It also works with large scale deployments by integrating with OpenStack.

> LXD was announced in early November 2014 and is still under very active development.

Distinguishing them on the command line

lxd

`lxd` is the LXD **daemon**. For interacting with the daemon (to create and manage containers, for instance), you want to use the `lxc` command. You generally don't want to invoke `lxd` directly – unless you need to run `lxd init` or something; check `man lxd` or `lxd --help` for more info on what you can do with running `lxd` directly, but once you get it running from your init system, you probably won't need to invoke it directly again unless you are debugging LXD itself.

The `lxc` command is the **LXD** front-end ("LXD Client" is how I think of it).

However, if you're trying to use **LXD**, you should avoid using any commands that start with `lxc-` (that's `lxc`, followed by a short hyphen)! These commands are associated with **LXC**.

lxc

LXC commands start with `lxc-` (that's `lxc` followed by a short hyphen). If there's no hyphen, just the literal command `lxc`, that's associated with **LXD**.

Packaging

The way LXD and LXC are packaged is distro-specific, but here's some info on the Ubuntu packages.

From **@stgraber** **here**:

> The packaging in Ubuntu is a bit confusing unfortunately…
> "lxc1" is the "LXC 1.0 user experience" which is what normal people call LXC
> "lxc2" is the "LXC 2.0 user experience" which is what normal people call LXD
> So if you install "lxc1", you'll get the usual LXC tools
> "lxc-create/lxc-start/…" and all of those will be at version 2.0.7,
> getting all the bugfix and security updates and supported through 2021.

Mixing and matching

It's generally a bad idea to mix and match LXC and LXD on the same system, in my opinion, because you are likely to get confused, or LXC and LXD might get *themselves* confused with sharing resources like namespaces, etc. I am not aware of any good use case for using both, so you should really decide on which one to use, and stick with it.

Even better, uninstall the command-line interface for the one you aren't using, so you don't accidentally run a command that's not applicable to the LX{D,C} you intend to use.

Deciding on which to use

It is this user's opinion that all "green field" (new user/new server) deployments looking at LXC or LXD as a solution should, in 99% of cases, just use LXD. This is especially true if your container host OS is Ubuntu 16.04 or later; you'll have the most secure, most streamlined experience on this specific OS distro/version combo.

Outside of Ubuntu, it's a bit easier to deploy LXC than LXD on distros completely outside the Debian/Ubuntu ecosystem, because it has fewer dependencies on kernel features and patches. However, with the proper kernel, it *is* possible to get a fully working LXD on non-Ubuntu, non-Debian distros. Support for LXD out of the box may improve in the future on some non-Ubuntu distros, but we'll have to wait and see.

Security

LXC and LXD use the AppArmor LSM and the Secure Computing (seccomp) facility of the Linux kernel for the following purposes:

- To limit the extent to which processes in an unprivileged container can "break out" and affect system-

wide resources on the host OS
- To minimize the impact of compromised processes within a container
- To isolate containers' resources from the host OS while allowing them to access varying levels of resources provided by the host, depending on what the user (from the host side) chooses to allow

Although there are pretty good "default" security measures in place for both LXC and LXD, the isolation is a bit more streamlined and easier to set up from a user perspective with LXD, in my opinion. Security has also been significantly improved over time, so using a patched/supported version of LXD is a very good idea.

On non-Ubuntu systems, the AppArmor profiles might not work (if the kernel lacks certain AppArmor features), which significantly weakens the security of either LXC or LXD. Additionally, Ubuntu ships with a bunch of AppArmor profiles out of the box that other distros might not provide.

Features

Both lxd and lxc have the concept of unprivileged vs. privileged containers. An unprivileged container is designed to be as isolated as possible from the host OS; a privileged container basically implies that with little effort, a root user in the container can "break out" into the host OS, so the barrier between the container and the host is mostly to prevent accidents and encourage good software configuration management practices (similar to Docker).

Unprivileged containers have relatively fewer features than privileged containers in terms of what they can do with the OS and direct hardware access, but that's a *good thing*, because many "direct access" type features only make sense when the user working with them is a trusted owner/administrator of the server. You certainly wouldn't want to give tenants on a multi-tenant container-based VPS hosting box a privileged container. They could easily spy on and interfere with other tenants.

Maintaining this post

Did you know that this post is a Wiki article? This means, once you've earned the "Basic" badge, you can *edit* this post to improve it. If you see an error, or want to add different perspectives or resources, please feel free – but try to keep it on the topic of LXD vs. LXC. For detailed specifics or how-tos for a specific product (LXC or LXD), you are encouraged to create your own Wiki articles.

To learn more about trust levels and badges on Discourse, **click here**.

20 Likes

Deploying django applications

Helper Function for opening a terminal in containers

Question about cpu.limits

LXC : ping host with 2 containers and access internet

System requirements?

Centos 7 vs ubuntu 18 missing /etc/ssh/sshd_config?

Two subnet, two interfaces... using macvlan?

How add another interface to lxc container with static/public ip ovh

Is it possible to run xwindow in a container?

Generate ubuntu18 and ubuntu18.root files from ubuntu:18.04 LXC Image

Name and package confusion between lxc and lxd

Multiple newbie questions

Where can I find config file for Ubuntu 18.04 LXC container?

stgraber (Stéphane Graber) pinned globally #2 April 25, 2017, 7:03pm

stgraber (Stéphane Graber) #3 April 25, 2017, 7:07pm

Thanks for the very useful post, I pinned it so that everyone can see it very easily.

Also, with the link limit now bumped to 5 for new users, you should be able to edit it and add the remaining link.

1 Like

allquixotic (Sean McNamara) #4 April 25, 2017, 7:10pm

Do you know if there's a way for either admins or other trusted users to edit this post? While I don't think it's necessarily desirable for the Discourse to be a general wiki where anyone can edit anything, having *a few* articles that are in Wiki-like format (where at least a subset of trusted/admin users can edit them) is desirable.

In this case, I'm happy to edit and maintain this document, but can't guarantee it'll always be up to date and the most accurate. Additionally, as other community members and developers, Canonical employees, etc. continue to write new blog posts and documentation for LXD and LXC, we'll want to have a Resources section at the bottom of the article that links to them.

BTW, for anyone else who's itching to write a post like this that is presentable as a pinned FAQ/Article, you should really consider writing one on LXD unprivileged container networking, with a focus on the most common scenarios as observed on the ML (for instance, giving containers a static IP via macvlan). Start

with high-level basics and gradually work into the command-line, explaining networking concepts as you go. If nobody posts one in the next week or two, I might work on it if I get some time.

---

stgraber (Stéphane Graber) #5 April 25, 2017, 7:15pm

I just marked that post as a wiki now, so anyone of trust level 1 or above can now edit it.
Anyone who's got trust level 2 or higher can make new wiki posts and admins can turn existing posts into wikis as needed (which is what I did here).

3 Likes

---

cryptofuture (cryptofuture) #6 November 6, 2017, 5:39pm

Basically I could say there no single situation where person should prefer LXD over LXC, beside situations where complex or heavy solution like proxmox used or environment created with LXC in mind.
LXD is more simple even for small tasks like running two tests containers on Desktop. And a lot more simpler in maintenance if you running big amounts of the containers.
I seen many times how people spending hours configuring network to make less robust solution, that I can setup with with lxc network in several seconds. More above with pure LXC I didn't even seen similar network setups, every person doing its own.
And thats even without talking about LXD REST api.
Sad part about it many people even not knowing that they using LXD, and still call it LXC.

---

thewyzard (Sean Zachariasen) #7 November 25, 2017, 3:34am

I'm battling that one in my chef environment… there are several resources out there that call themselves 'lxd' but they are simple wrappers around around the lxc CLI tool and are not actually LXD. There's little to no understanding of the distinction in the general market, but I would soooo love to have their namespaces because I am trying to do it right, by giving the consumer options between the CLI and the REST api.

---

angristan #8 February 19, 2018, 11:02am

Thanks for the wiki page. LXD is a nice project and allows to use LXC containers very easily! The only downside is that it only works perfectly on Ubuntu Server.

stgraber (Stéphane Graber) #9 February 19, 2018, 4:03pm

Have you tried the LXD snap on other distros? If so, what kind of problem did you run into?

We have CI in place for the LXD snap on a number of other Linux distributions and aren't aware of anything being broken, though some features will be affected by exactly what the distribution's kernel supports.

angristan #10 February 19, 2018, 9:11pm

Hi Stéphane and thanks for your work.

Yes I'm using the snap package on Debian, although I would prefer to have it in the Debian repo. At least I have an up-to-date version to play with. 🙂

Indeed, after using it a bit more, most of the apparent drawbacks I read on the web about using LXD trough the snap package aren't true anymore. What a great experience!

I have issues with CRIU (**https://stgraber.org/2016/04/12/lxd-2-0-remote-hosts-and-container-migration-612/**) but I also get them on Ubuntu Server so I guess they're not related.

stgraber (Stéphane Graber) #11 February 19, 2018, 11:00pm

Yeah, CRIU is pretty hard to get working and very easy to break once it does 🙂

I've added experimental support for CRIU in the snap package too, but you have to manually opt into that given how unreliable CRIU is:

```
snap set lxd criu.enable=true
systemctl reload snap.lxd.daemon
```

1 Like

casperghst42 #13 August 27, 2018, 5:10pm

Interesting information, but just a point; work also very nicely with SELinux.

I personally don't some of the decisions made with LXD and how lxc init is working. The "defaults" for things like storage path is an absolute pain to change, and should be looked at.

simos #14 August 27, 2018, 8:46pm

You probably mean `lxd init` (and not `lxc init`, which creates a container but does not launch it).

`lxd init` is sort of a wizard that helps to configure LXD with some sane defaults.
If you do not like the defaults, you can use a *preseed* file instead to configure LXD in one go.
Alternatively, you can avoid `lxd init` altogether and run the set of configuration `lxc` commands manually to your liking.

Indeed, an issue with `lxd init` is that as a wizard, it applies each choice as you go along. If it aborts further down the line, the previous configuration (for example, about storage) has already been applied.

casperghst42 #15 August 28, 2018, 9:36am

Yes, true “lxd init”.

Thanks, I need to have a look at it.

Skaperen (Phil Howard) #16 October 24, 2018, 1:33am

IMHO, along the lines of *this* post/article, another useful one (or two) would be to list some typical use cases for choosing either **lxc** or **lxd**.

i have not made a decision, yet. but i am leaning towards lxc. there are two reasons i am coming to this conclusion. the 1st is that i’m mostly interested personally in doing system level things with containers. the 2nd is the i am wanting to learn about containers the way i would have learned had i not been misdirected away when they first came out years ago. i am doing catch-up now. i would have learned lxc 1st because at one time that’s all there was. then i want to learned lxd in terms of already knowing lxc. and somewhere in their i want to also learn the API in terms of programming in C and in Python. i don’t think my use cases are typical, though. and they might well be confusing to typical users who only learn lxd (or Docker).

one of my goals is to set up a dual-distro system on my laptop with Ubuntu and Slackware (maybe also some others like Centos, Debian, and/or Fedora). and part of that goal is to have both Ubuntu and Slackware each running in their own container with the host system minimized to run containers and suitable system administrator tools. and i also want to look into building “distros” targeted to only be container images.

pdosch (Pete Dosch) #18 December 13, 2018, 4:48pm

Can someone expand on the security dependencies of LXD on apparmor? I want to run LXD containers, but enable SELinux security, which I believe removes apparmor. Is this safe to do, and does anyone know of articles on how to secure an LXD container with SELinux?

---

stgraber (Stéphane Graber) #19 December 13, 2018, 5:09pm

LXD doesn't require apparmor, it will happily run on systems that have it missing or disabled.

With AppArmor disabled, privileged containers should be considered as entirely unsafe. While we don't consider them to be root safe when apparmor is present, we also don't know of a trivial way to escape in that case, but without apparmor it's downright trivial.

Unprivileged containers (default) should be perfectly safe as apparmor only acts as a safety net there with the user namespace acting as the main security barrier.

---

pdosch (Pete Dosch) #20 December 14, 2018, 1:09pm

Thanks for the confirmation!

Skaperen (Phil Howard) #21 December 15, 2018, 5:55am

fyi, i have decided to go with LXD. i just haven't installed it, yet, because my needs are not urgent. but there is a new need that i might like to do sooner, if it is doable.

---

sairoop10 (SS Roop) #22 February 21, 2019, 9:05am

> **LXC container APIs vs LXC command line utilites**
>
> Using LXC containers I'm trying to transform it into a NAT like network topology. For this I succeeded creating NAT like container with iptables, dnsmasq etc., like services running in it and also made it flexible enough to move from one part of the network to other. But now I tried of recreating the whole construct using LXC container API's. This time my code is not working properly like, I modified /var/lib /lxc/nat/config for creating/adding a new interface to the container and when started …

Can you please solve this issue. I've been waiting for the reply for a long time

next page →

Powered by Discourse, best viewed with JavaScript enabled