# LXD
# containers for your home lab

Michal Halenka

@HalisCz

prgcont.cz | github.com/prgcont/talks | @prgcont

2018-03-22

# ~~Pre-flight~~ Font size check

```
# prepare working directory
mkdir prgcont
cd prgcont

# get Vagrantfile
wget https://github.com/prgcont/talks/raw/master/2018-03-22/Vagrantfile

# Fedora porn
export VAGRANT_DEFAULT_PROVIDER=virtualbox

# run VM to get OS images upfront
vagrant up
```

https://slides.com/michalhalenka/prgcont-lxd/live

# Hands-on preparation

```
# prepare working directory
mkdir prgcont
cd prgcont

# get Vagrantfile
wget https://github.com/prgcont/talks/raw/master/2018-03-22/Vagrantfile

# Fedora porn
export VAGRANT_DEFAULT_PROVIDER=virtualbox

# run VM to get OS images upfront
vagrant up
```

# Flight plan

- Theory, history, features
- LXD 101 with hands-on
- Use-cases, examples

# Theory, history, features

# Theory

- system containers (not application containers)
- based on top of LXC
- by Canonical
- ... but intuitive (at least compared to LXC commands)
- Go
- secure (unpriviledged containers)
- resource control (cpu, ram, net I/O, block I/O, disk usage, ...)
- device passthrough (USB, GPU, unix char, unix block, NIC, paths, ...)
- network management
- storage management
- cross-host live-migration with CRIU
- REST API
- integrates with OpenStack

# History

2008 - initial release of LXC

2013 - first versions of docker running on LXC

2014 - LXC 1.0 introduces unpriviledged mode

2015 - LXD 0.1

2016 - LXC rewritten in C, compatible with LXC 1.x

2017 - LXD 2.0.0

# Moving parts

- LXD - privileged daemon
- LXC - CLI client for LXD
- LXCFS - FUSE for overlay of cpuinfo, meminfo, cgroupfs, ...
- LXD container
- LXD snapshot
- LXD image

# Security

- *kernel namespaces* to keep everything separate from the rest of the system
- *seccomp* to filter dangerous system calls
- *AppArmor* to provide restrictions on mounts, socket, file access, ...
- *capabilities* to prevent loading kernel modules, altering time, ...
- *cgroups* to restrict usage of resources
- *TLS 1.2* to secure client <=> server communication

# LXD container

- rootfs filesystem
- configuration, resource limit, env, ...
- devices (disks, NICs, ...)
- profiles (inherited configuration)
- properties (architecture, persistency, name)
- runtime state (when using CRIU)

# LXD snapshot

- immutable LXD container
- ... can be renamed

# LXD image

- clean Linux distribution image
- sha256hash
- aliases
- preconfigured image servers

# LXD profile

- container configuration and devices in one place
- inheritance
- predefined "default" and "docker"

# LXD remote

- another LXD instance acting as source of images
- predefined: local, ubuntu, ubuntu-daily, images

https://images.linuxcontainers.org/

# Storage backend

(from basic to the most powerfull)

- directory (no quotas, no instant cloning)
- LVM
- Btrfs
- ZFS
- CEPH

https://github.com/lxc/lxd/blob/master/doc/storage.md#storage-backends-and-supported-functions

... buckle up, the boring theory is over

# LXD 101

# Installation

| | |
|---|---|
| Alpine | apk add lxd |
| ArchLinux | AUR |
| Fedora | COPR * |
| Gentoo | emerge lxd |
| Ubuntu 16.04+ | apt install lxd lxd-client |
| Snap | snap install lxd |
| MacOS (client only) | brew install lxc |

(done in vagrant)

# Initialization

```
##### with wizard

root@ubuntu-artful:~# lxd init

Do you want to configure a new storage pool (yes/no) [default=yes]?
Name of the storage backend to use (dir or zfs) [default=dir]:
Would you like LXD to be available over the network (yes/no) [default=no]?

Do you want to configure the LXD bridge (yes/no) [default=yes]? no

LXD has been successfully configured.



##### without wizard
root@ubuntu-artful:~# lxd init --auto
```

(done in vagrant)

# Basic LXD container

```
##### create container
root@ubuntu-artful:~# lxc launch images:debian/9 prgcont1
Creating prgcont1

The container you are starting does not have any network attached to it.
  To create a new network, use: lxc network create
  To attach a network to a container, use: lxc network attach

Starting prgcont1

##### list containers
root@ubuntu-artful:~# lxc list
+----------+---------+------+------+------------+-----------+
|   NAME   |  STATE  | IPV4 | IPV6 |    TYPE    | SNAPSHOTS |
+----------+---------+------+------+------------+-----------+
| prgcont1 | RUNNING |      |      | PERSISTENT | 0         |
+----------+---------+------+------+------------+-----------+
```

# Basic LXD container

```
##### run something in container
root@ubuntu-artful:~# lxc exec prgcont1 bash

root@prgcont1:~# cat /etc/debian_version
9.4

##### delete container
root@ubuntu-artful:~# lxc delete prgcont1 --force

##### rinse, repeat, ...
root@ubuntu-artful:~# lxc launch debian/9 prgcont1
```


ONE DOES NOT SIMPLY
USE CONTAINERS JUST FOR BASH
imgflip.com

# Container's configuration

```
root@ubuntu-artful:~# lxc config show prgcont1
architecture: x86_64
config:
  image.architecture: amd64
  image.description: Debian stretch amd64 (20180314_22:42)
  image.os: Debian
  image.release: stretch
  image.serial: "20180314_22:42"
  volatile.base_image: 4eb18a0eb154cc07f96996aba6b9b1ce71a461389ec254d3da4577248586c729
  volatile.idmap.base: "0"
  volatile.idmap.next: '[{"Isuid":true,"Isgid":false,"Hostid":100000,"Nsid":0,"Maprange":65536},{
  volatile.last_state.idmap: '[{"Isuid":true,"Isgid":false,"Hostid":100000,"Nsid":0,"Maprange":65
  volatile.last_state.power: RUNNING
devices: {}
ephemeral: false
profiles:
- default
stateful: false
description: ""
```

# Profile configuration

```
##### list profiles
root@ubuntu-artful:~# lxc profile list
+---------+---------+
|  NAME   | USED BY |
+---------+---------+
| default | 1       |
+---------+---------+

##### show profile
root@ubuntu-artful:~# lxc profile show default
config: {}
description: Default LXD profile
devices:
  root:
    path: /
    pool: default
    type: disk
name: default
used_by:
- /1.0/containers/prgcont1
```

# NAT network for all containers

```
##### create bridge
root@ubuntu-artful:~# lxc network create testbr0
# OR # lxc network create testbr0 ipv6.address=none ipv4.address=10.0.3.1/24 ipv4.nat=true

##### show bridge
root@ubuntu-artful:~# lxc network show testbr0
config:
  ipv4.address: 10.230.85.1/24
  ipv4.nat: "true"
  ipv6.address: fd42:f774:512f:b682::1/64
  ipv6.nat: "true"
description: ""
name: testbr0
type: bridge
used_by: []
managed: true

##### include bridge into default profile
root@ubuntu-artful:~# lxc network attach-profile testbr0 default eth0
##### reload network in container and enjoy
root@prgcont1:~# service networking restart && apt update
```

# File transfer

```
##### get file from container
root@ubuntu-artful:~# lxc file pull <container>/<path> <dest>


##### get file to stdout
root@ubuntu-artful:~# lxc file pull <container>/<path> -
# i.e. # lxc file pull prgcont1/etc/resolv.conf -

##### get file to container
root@ubuntu-artful:~# lxc file push <source> <container>/<path>


##### edit file in container
root@ubuntu-artful:~# lxc file edit <container>/<path>
```

# Snapshot

```
##### create snapshot
root@ubuntu-artful:~# lxc snapshot prgcont1 my-precious-snapshot

##### list snapshots counts
root@ubuntu-artful:~# lxc list

##### list snapshots of container
root@ubuntu-artful:~# lxc info prgcont1

##### restore snapshot
root@ubuntu-artful:~# lxc restore prgcont1 my-precious-snapshot

##### restore snapshot to new container
root@ubuntu-artful:~# lxc copy prgcont1/my-precious-snapshot prgcont1.1

##### delete snapshot
root@ubuntu-artful:~# lxc delete prgcont1/my-precious-snapshot
```

# Limits

```
##### limit all containers to 1 CPU
root@ubuntu-artful:~# lxc profile set default limits.cpu 1

##### limit prgcont1 to 4 CPU but only for 10 %
root@ubuntu-artful:~# lxc config set prgcont1 limits.cpu 4
root@ubuntu-artful:~# lxc config set prgcont1 limits.cpu.allowance 10%

##### limit prgcont1 to 256MB RAM
root@ubuntu-artful:~# lxc config set prgcont1 limits.memory 256MB

##### limit disk
root@ubuntu-artful:~# lxc config set prgcont1 root limits.read 10MB
root@ubuntu-artful:~# lxc config set prgcont1 root limits.write 10IOps

##### get limits and usage
root@ubuntu-artful:~# lxc info prgcont1
```

# Live migration (preparation)

```
##### get criu

root@ubuntu-artful:~# apt install criu
```

# Live migration (intro)

```
##### create statefull snapshot

root@ubuntu-artful:~# lxc snapshot prgcont1 stateful_snap --stateful

##### stateful stop/start
root@ubuntu-artful:~# lxc stop prgcont1 --stateful

# examine the state in image files
root@ubuntu-artful:~# tree /var/lib/lxd/containers/prgcont1/rootfs/state/
```

# Live migration (in theory)

```
##### live migrate to another lxd remote

root@ubuntu-artful:~# lxc move prgcont1 remote_lxd_server:migrated_prgcont1
```

- still in progress
- basic containers should live migrate in Ubuntu 16.04

# Thank you for your attention

# Any questions?