LXC

Florent Gluck - Florent.Gluck@hesge.ch

April 1, 2022

What is LXC?

- LXC = Linux Containers
  - low-level Linux container runtime
- Run multiple isolated Linux systems on a single host
- Provide OS level virtualization (not an hypervisor!)
  - provide virtual OS with own CPU, memory, I/O and filesystem
- Provide a user space API
- Use kernel-based isolation mechanisms (capabilities, namespaces, cgroups, seccomp)
- Containers share the same kernel as the host kernel!

LXC components

- LXC is very minimalistic: thin layer over Linux kernel features
- Only the following components are installed in a typical LXC installation:
  - Set of userspace tools
  - Templates
  - Libraries
  - Language bindings
- Make sure to use LXC version ≥ 2.0

Templates

- LXC uses templates to create different OS containers
- Templates = scripts to bootstrap specific OS
- Each Linux distribution supported by LXC has a script dedicated to it
- A generic script, `download`, allows to install various distributions using images of root file systems
- Check out https://images.linuxcontainers.org for available images

LXC API

```c
// API available at https://github.com/lxc/lxc
#include <lxc/lxccontainer.h>
#include <err.h>

int main() {
    struct lxc_container *c = lxc_container_new("apicontainer", NULL);
    if (!c) errx(1, "Failed to setup lxc_container struct");

    if (c->is_defined(c)) errx(1, "Container already exists");

    if (!c->createl(c, "download", NULL, NULL, LXC_CREATE_QUIET, "-d", "ubuntu",
          "-r", "focal", "-a", "amd64", NULL)) errx(1, "Failed to create
         container rootfs");

    if (!c->start(c, 0, NULL)) errx(1, "Failed to start the container");
    printf("Container state and PID: %s %d\n", c->state(c), c->init_pid(c));

    if (!c->shutdown(c, 30)) {
        printf("Failed to cleanly shutdown the container, forcing.\n");
        if (!c->stop(c)) errx(1, "Failed to kill the container");
    }

    if (!c->destroy(c)) errx(1, "Failed to destroy the container");
    return EXIT_SUCCESS;
}
```

Installation on Ubuntu

- Packages to install: `lxc`, `lxc-templates`
- Available commands:

`lxc-autostart`	`lxc-freeze`
`lxc-attach`	`lxc-info`
`lxc-cgroup`	`lxc-ls`
`lxc-checkconfig`	`lxc-monitor`
`lxc-checkpoint`	`lxc-snapshot`
`lxc-config`	`lxc-start`
`lxc-console`	`lxc-stop`
`lxc-copy`	`lxc-top`
`lxc-create`	`lxc-unfreeze`
`lxc-destroy`	`lxc-unshare`
`lxc-device`	`lxc-wait`
`lxc-execute`	`lxc-usernsexec`
`lxc-update-config`	

Kernel support?

```
~ $ lxc-checkconfig
LXC version 4.0.6
Kernel configuration not found at /proc/config.gz; searching...
Kernel configuration found at /boot/config-5.10.0-1029-oem
--- Namespaces ---
Namespaces: enabled
Utsname namespace: enabled
Ipc namespace: enabled
Pid namespace: enabled
User namespace: enabled
Network namespace: enabled

--- Control groups ---
Cgroups: enabled

Cgroup v1 mount points:
/sys/fs/cgroup/systemd
/sys/fs/cgroup/devices
/sys/fs/cgroup/cpu,cpuacct
/sys/fs/cgroup/cpuset
/sys/fs/cgroup/net_cls,net_prio
/sys/fs/cgroup/hugetlb
/sys/fs/cgroup/perf_event
/sys/fs/cgroup/rdma
...
```

Basic commands

`lxc-create`	create a container from a template script
`lxc-start`	start running a container
`lxc-ls`	list containers on the system
`lxc-attach`	start a process inside a running container
`lxc-console`	launch a console (login) for the specified container
`lxc-stop`	stop a running container
`lxc-destroy`	destroy a container

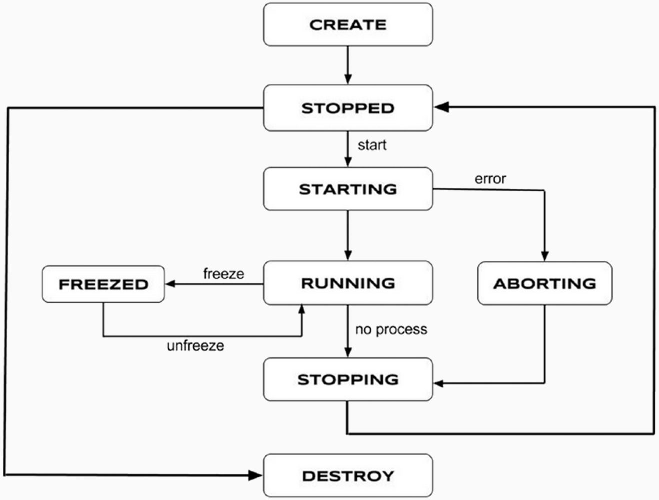
Command failure?

- LXC commands might fail without displaying any error message
- Most commands provide logging options often exposed as:

```
Common options :
  -o, --logfile=FILE        Output log to FILE instead
      of stderr
  -l, --logpriority=LEVEL   Set log priority to LEVEL
```

- The `logfile` will contain detailed messages according to the detail level specified in `logpriority`

LXC container life cycle



Container files and configuration

- Templates script found in:
  `/usr/share/lxc/templates/`
- Default configuration file:
  `/etc/lxc/default.conf`
- Container files in:
  `/var/lib/lxc/<container_name>/`
- Container configuration in:
  `/var/lib/lxc/<container_name>/config`
- LXC configuration parameters:
  `man lxc.container.conf`

Creating containers

- Creating a container from a template
  - Available templates found in `/usr/share/lxc/templates/`

```
lxc-create -n <name> -t <template> -- -r <release>
```

- Creating a container from an image
  - Must use the `download` template which installs distributions from root filesystem images
  - Available images listed at https://images.linuxcontainers.org

```
lxc-create -n <name> -t download -- -d <distrib> -r <release> -a <
     arch>
```

  - NOTE: on Ubuntu 20.04, the following env. var. is necessary:

```
export DOWNLOAD_KEYSERVER="hkp://keyserver.ubuntu.com"
```

Privileged containers

LXC allows creating privileged containers

- Created by root and running as root
- UID 0 in container mapped to UID 0 outside of container
- **Dangerous**: someone escaping the container will be root on the host!

Unprivileged containers (1/2)

LXC allows creating unprivileged[1] containers started by root

- Created by root, started by root, but not running as root
- Accomplished by using user namespaces
- UID 0 in container **not** mapped to UID 0 outside of container
- E.g. UID 0 $\rightarrow$ 65536 in container mapped to 100000 $\rightarrow$ 165536 on host
- **Much safer** than privileged containers!

[1] only download template can be used with unprivileged containers!

Unprivileged containers (2/2)

LXC allows creating **unprivileged** containers

- Created by non-root user and running as non-root user
- Requires a bit of system configuration
- **The safest** way of running containers, albeit with more limitations

Creating unprivileged containers as root

- Allocate UID and GID ranges to root in `/etc/subuid` and `/etc/subgid`, e.g.:

```
root:100000:65536
```

- Specify the range in `/etc/lxc/default.conf` using `lxc.idmap`, e.g.:

```
lxc.idmap = u 0 100000 65536
lxc.idmap = g 0 100000 65536
```

  this tells LXC to map UID 0 to UID 100000, etc.

- Allows everyone to go through `/var/lib/lxc`

```
chmod +x /var/lib/lxc
```

Limiting resources

- To limit resources at runtime (temporarily):
  `lxc-cgroup -n <container> <state-object> <value>`
- To limit resources persistently:
  - edit container's configuration
    `/var/lib/lxc/<container>/config`
- Examples:

```
lxc-cgroup -n mycontainer cpuset.cpus "0,1"
lxc-cgroup -n mycontainer memory.limit_in_bytes
    "64000000"
```

Snapshots

LXC support various types of snapshots through `lxc-copy`:

- Full copy snapshots
- Copy-on-write snapshots where only differences are written
  - requires a filesystem that supports it: btrfs, lvm, overlay, zfs
- Ephemeral snapshots are automatically destroyed on shutdown

Resources

- Man pages:
  - `man lxc`
  - `man lxc.container.conf`
- Linux container and virtualization tools
  https://linuxcontainers.org/
- Practical LXC and LXD “Linux Containers for Virtualization and Orchestration”, Senthil Kumaran S., Apress 2017