LXC Classic Linux containers

START HERE

`lxc-create -n <name> -t download`
Create from an image

`lxc-info -n <name>`
State, PID, IPs

`lxc-start -n <name>`
Boot it in background

`lxc-stop -n <name>`
Clean shutdown

`lxc-ls -f`
List with state and IPs

`lxc-snapshot -n <name>`
Save a restore point

`lxc-attach -n <name>`
Root shell inside

`lxc-destroy -n <name>`
Delete it

Template download, busybox → `lxc-create` → **Container dir** config + rootfs → `lxc-start` → **RUNNING** init + namespaces → `lxc-stop` → **STOPPED** still defined → `lxc-destroy` → **Gone** dir removed

CONCEPTS

Every tool is a separate lxc-* program taking -n <name>; there is no single lxc command in the classic tools. The lxc client of LXD/Incus is a different thing: see the Incus sheet.

Unprivileged containers (run as your user, uid-mapped) need /etc/subuid, /etc/subgid and an /etc/lxc/lxc-usernet quota; root-run containers are privileged by default.

Containers live in /var/lib/lxc/<name>/ (root) or ~/.local/share/lxc/ (unprivileged); -P <path> points any tool at another store.

CREATE AND DESTROY

`lxc-create -n <name> -t download`
Create from the image server (asks distro)
Add -- -d <dist> -r <release> -a <arch> to skip the prompts.

`lxc-create -n <name> -t download -- -d debian -r bookworm -a amd64`
Create non-interactively

`lxc-create -n <name> -t busybox`
Tiny test container

`lxc-create -n <name> -t download -f <config>`
Create with a config file

`lxc-create -n <name> -t download -B btrfs`
Pick the backing store
Types include dir, btrfs, zfs, lvm, loop, rbd; lvm and loop also take --fstype and --fssize.

`lxc-create -t download -h`
Show template options

`lxc-destroy -n <name>`
Delete a stopped container

`lxc-destroy -n <name> -f`
Stop and delete in one go

`lxc-destroy -n <name> -s`
Delete including snapshots

RUN AND STOP

`lxc-start -n <name>`
Start in the background

`lxc-start -n <name> -F`
Start in foreground, see boot output

RUN AND STOP (cont.)

`lxc-start -n <name> -l DEBUG -o <file>`
Log a failing start
First stop when a container will not boot; add -F to see the console too.

`lxc-stop -n <name>`
Request clean shutdown

`lxc-stop -n <name> -k`
Kill instead of shutting down

`lxc-stop -n <name> -t <sec>`
Hard-stop after a timeout

`lxc-stop -n <name> -r`
Reboot the container

`lxc-freeze -n <name>`
Pause all processes

`lxc-unfreeze -n <name>`
Resume a frozen container

`lxc-wait -n <name> -s RUNNING`
Block until a state is reached
States can be OR-ed: -s 'STOPPED|FROZEN'; add -t <sec> for a timeout.

`lxc-autostart -a`
Start all lxc.start.auto containers

INSPECT

`lxc-ls -f`
List containers, state, IPs

`lxc-ls --running`
Only running containers

`lxc-ls --stopped`
Only stopped containers

`lxc-ls --filter <regex>`
Filter names by regex

`lxc-ls -F NAME,STATE,IPV4`
Pick the fancy columns

`lxc-info -n <name>`
Show state, PID, IPs, usage

`lxc-info -n <name> -iH`
Only the IP addresses, plain

`lxc-info -n <name> -s`
Only the state

`lxc-info -n <name> -S`
Usage stats (CPU, memory, I/O)

`lxc-top`
Live per-container usage

`lxc-checkconfig`
Check the kernel supports LXC

GET INSIDE

`lxc-attach -n <name>`
Shell inside the container

`lxc-attach -n <name> -- <cmd>`
Run one command inside

GET INSIDE (cont.)

`lxc-attach -n <name> --clear-env -- <cmd>`
Run with a clean environment

`lxc-attach -n <name> -s 'NETWORK' -- <cmd>`
Enter only some namespaces
Implies elevated privileges and can leak host rights into the command. Use with care.

`lxc-console -n <name>`
Attach to the container console
Detach with Ctrl+a q.

`lxc-device -n <name> add /dev/<dev>`
Hot-add a host device

`lxc-device -n <name> add <iface>`
Move a host interface inside

CLONES AND SNAPSHOTS

`lxc-snapshot -n <name>`
Create next snapshot (snap0, snap1...)

`lxc-snapshot -n <name> -L`
List snapshots

`lxc-snapshot -n <name> -r <snap>`
Restore a snapshot

`lxc-snapshot -n <name> -r <snap> -N <new>`
Restore into a new container

`lxc-snapshot -n <name> -d <snap>`
Delete a snapshot

`lxc-copy -n <name> -N <new>`
Clone a container

`lxc-copy -n <name> -N <new> -s`
Clone as copy-on-write snapshot
Needs a backing store that supports it (btrfs, zfs, lvm, overlay).

`lxc-copy -n <name> -e`
Start a throwaway ephemeral copy

`lxc-copy -n <name> -N <new> -R`
Rename a container

CONFIG AND LIMITS

`lxc.start.auto = 1`
Start at host boot

`lxc.uts.name = <host>`
Container hostname

`lxc.net.0.type = veth`
Virtual NIC type

`lxc.net.0.link = <bridge>`
Bridge to attach to

CONFIG AND LIMITS (cont.)

`lxc.net.0.flags = up`
Bring the NIC up

`lxc.net.0.ipv4.address = <ip>/<mask>`
Static IPv4 address

`lxc.mount.entry = <src> <dst> none bind 0 0`
Bind-mount a host dir
The destination is relative to the rootfs, without a leading slash.

`lxc.cgroup2.memory.max = <bytes>`
Memory limit (cgroup v2)

`lxc.cgroup2.cpu.max = <quota> <period>`
CPU limit (cgroup v2)

`lxc.idmap = u 0 100000 65536`
Map container uids to host uids

`lxc-cgroup -n <name> <key> <value>`
Change a limit on a running container

FILES AND PATHS

`/var/lib/lxc/<name>/config`
Per-container config (root)

`/var/lib/lxc/<name>/rootfs`
Container filesystem on the host

`~/.config/lxc/default.conf`
Defaults for new unprivileged containers

`/etc/lxc/default.conf`
Defaults for new root containers

`/etc/lxc/lxc-usernet`
Per-user veth and bridge quota

RECIPES

Unprivileged container as your user

```
# one-time: ranges and NIC quota
echo "$USER veth lxcbr0 10" | sudo tee -a /etc/lxc/lxc-usernet
mkdir -p ~/.config/lxc
cp /etc/lxc/default.conf ~/.config/lxc/default.conf
# add the idmap lines for your /etc/subuid and /etc/subgid range, then:
lxc-create -n web -t download -- -d debian -r bookworm -a amd64
lxc-start -n web && lxc-ls -f
```

Safe experiment with rollback

```
lxc-stop -n web
lxc-snapshot -n web
lxc-start -n web
# ... break things ...
lxc-stop -n web -k
lxc-snapshot -n web -r snap0
```