LXD

[lxd]

install

```
apt remove lxd lxd-client
snap install lxd

# create zfs dataset on pool rpool
zfs create rpool/lxd

# create lxd storage called zfspool using previous defined dataset
lxc storage create zfspool zfs source=rpool/lxd

# define default storage pool
lxc profile device add default root disk path=/ pool=zfspool

# initialize network
sudo lxd init
```

Because group membership is only applied at login, you then either need to close and re-open your user session or use the “newgrp lxd” command in the shell you're going to interact with lxd from

```
newgrp lxd
```

lxc-prepare (chmod +x)

```
#!/bin/bash

NAME=$1
ALIAS=$2
ALIAS=${ALIAS:=xenial}

lxc image show $ALIAS >/dev/null 2>&1
if [ ! $? = 0 ]; then
    echo lxc image copy images:ubuntu/xenial/amd64 local: --alias
xenial
    exit 0
fi

if [ ! -f /etc/apt/apt.conf.d/proxy.conf ]; then
    sudo apt install apt-cacher-ng
    PROXY=$( lxc network show lxdbr0 | sed -n 's/\s\+ipv4.address:
\([0-9\.]\+\).*/\1/p' )
    echo "Acquire::http::Proxy \"http://$PROXY:3142\";" | sudo tee
/etc/apt/apt.conf.d/proxy.conf
    echo "PfilePattern = .*" | sudo tee -a /etc/apt-cacher-ng/acng.conf
```

```
    echo "PassThroughPattern: .*" | sudo tee -a /etc/apt-cacher-
ng/acng.conf
    systemctl restart apt-cacher-ng
fi


lxc info $NAME >/dev/null 2>&1
if [ ! $? = 0 ]; then
    lxc launch $ALIAS $NAME
fi


if [ -f /etc/apt/apt.conf.d/proxy.conf ]; then
    lxc file push /etc/apt/apt.conf.d/proxy.conf
$NAME/etc/apt/apt.conf.d/
fi

lxc file push /etc/inputrc $NAME/etc/
```

basic

list remote images

```
lxc image list images:
```

auto update remote images

```
lxc config set images.auto_update_cached true
```

import image

```
lxc image copy images:ubuntu/xenial/amd64 local: --alias xenial
```

create profile

```
lxc profile create juju-default
cat profile.yaml | lxc profile edit juju-default
```

profile.yaml

```
name: juju-default
config:
  boot.autostart: "true"
  security.nesting: "true"
  security.privileged: "true"
  linux.kernel_modules: openvswitch,nbd,ip_tables,ip6_tables
devices:
```

```
  eth0:
    mtu: "9000"
    name: eth0
    nictype: bridged
    parent: br-mng
    type: nic
  kvm:
    path: /dev/kvm
    type: unix-char
  mem:
    path: /dev/mem
    type: unix-char
  root:
    path: /
    type: disk
  tun:
    path: /dev/net/tun
    type: unix-char
```

create container from local image

```
lxc image list
lxc launch xenial test1 --profile juju-default
```

create container from remote image

```
lxc launch images:ubuntu/xenial/amd64 xenial1
lxc config set xenial1 boot.autostart false
lxc list
```

create custom image from local container

```
lxc publish local-container --alias mycustomimage
```

create container from previous image

```
lxc launch mycustomimage newcontainer
```

bash inside

```
lxc exec trusty1 -- /bin/bash
```

stop and delete

```
lxc stop trusty1
lxc delete trusty1
```

autostart on host boot

```
lxc config set <name> boot.autostart true
```

show container configuration

```
lxc config show <name>
```

proxy

```
apt install apt-cacher-ng
NAME=x11test
lxc file push /etc/apt/apt.conf.d/proxy.conf $NAME/etc/apt/apt.conf.d/
```

/etc/apt/apt.conf.d/proxy

```
Acquire::http::Proxy "http://10.106.191.1:3142";
```

network

```
lxc network create br0
lxc network show br0
lxc network edit br0
```

static IP container

```
istance=c1

lxc stop $instance
lxc network attach lxdbr0 $istance eth0 eth0
lxc config device set $istance eth0 ipv4.address 10.99.10.42
lxc start $istance
```

servers

prepare lxd server

```
# bind to port 8443
lxc config set core.https_address "[::]"

# password
lxc config set core.trust_password some-password
```

from client add remote server

```
lxc remote add myserver <ip address or DNS>
```

run command

```
lxc exec myserver:trusty1 -- bash
```

xorg integration

- https://bitsandslices.wordpress.com/2015/12/08/creating-an-lxd-container-for-graphics-applications/

container

create container

```
NAME=x11test
lxc launch images:ubuntu/bionic/amd64 $NAME
```

install simpler X program

```
lxc exec $NAME -- apt install xterm
lxc exec $NAME bash
apt install mesa-utils x11-apps
```

```
NAME=nvidia-sdk-manager
# lxc config set $NAME environment.DISPLAY <ip-of-host-lxdbr0-bridge>:0
lxc config set $NAME environment.DISPLAY :0
lxc config device add $NAME X0 disk path=/tmp/.X11-unix/X0 source=/tmp/.X11-unix/X0
lxc config device add $NAME Xauthority disk path=/root/.Xauthority source=${XAUTHORITY}
```

on host

for gmd (ubuntu >= 17.10) or ...

/etc/gdm3/custom.conf

```
[security]
DisallowTCP=false

[xdmcp]
Enable=true
```

... or for lightdm

/etc/lightdm/lightdm.conf

```
xserver-allow-tcp=true
```

```
xserver-command=X -listen tcp
```

add ip of container on /etc/X0.hosts

```
NAME=x11test
lxc info $NAME | sed -n "s/\s*eth0:\s*inet\s\([0-9\.]*\).*/\1/p" >>
/etc/X0.hosts
```

launch X application in container

```
xhost +
lxc exec $NAME -- xterm
```

audio integration

- https://bitsandslices.wordpress.com/2015/12/10/using-audio-in-lxd-containers/

misc devices

```
lxc config device add <name> rfxcom unix-char path=/dev/ttyACM0
lxc config device set <name>  rfxcom  mode 666
```

share folder

```
# only first time
echo "root:$UID:1" | sudo tee -a /etc/subuid
echo "root:${id -d}:1" | sudo tee -a /etc/subgid
lxc profile set default security.privileged true

# for every share
# lxc init stretch giano
lxc config set gianocop security.privileged true
lxc config set giano raw.idmap "both $UID $UID"
# source is on host, path is inside container
lxc config device add giano develop disk source=/mnt/giano path=/mnt/giano
```

migration

on host-destination

```
lxc config set core.https_address 0.0.0.0:8443
lxc config set core.trust_password PASSWORDhere
```

on host-origin

```
# add destination lxd
lxc remote add other-server <ip-address>

# take snap0 on gianocop container
lxc snapshot gianocop snap0
lxc copy gianocop/snap0 other-server:gianocop --verbose
lxc delete gianocop/snap0
```

on host-destination delete volatile in “lxc config”

```
volatile.base_image:
6adc9ca1a1124ebd954ba787e83dd9318866fd0b9ddce1cffc612559cfe3bc88
  volatile.eth0.hwaddr: 00:16:3e:50:f6:e8
  volatile.eth0.name: eth0
  volatile.idmap.base: "0"
  volatile.idmap.next:
'[{"Isuid":true,"Isgid":false,"Hostid":165536,"Nsid":0,"Maprange":1000},{"Is
uid":true,"Isgid":true,"Hostid":1000,"Nsid":1000,"Maprange":1},{"Isuid":true
,"Isgid":false,"Hostid":166537,"Nsid":1001,"Maprange":64535},{"Isuid":false,
"Isgid":true,"Hostid":165536,"Nsid":0,"Maprange":1000},{"Isuid":true,"Isgid"
:true,"Hostid":1000,"Nsid":1000,"Maprange":1},{"Isuid":false,"Isgid":true,"H
ostid":166537,"Nsid":1001,"Maprange":64535}]'
  volatile.last_state.idmap:
'[{"Isuid":true,"Isgid":false,"Hostid":165536,"Nsid":0,"Maprange":1000},{"Is
uid":true,"Isgid":true,"Hostid":1000,"Nsid":1000,"Maprange":1},{"Isuid":true
,"Isgid":false,"Hostid":166537,"Nsid":1001,"Maprange":64535},{"Isuid":false,
"Isgid":true,"Hostid":165536,"Nsid":0,"Maprange":1000},{"Isuid":true,"Isgid"
:true,"Hostid":1000,"Nsid":1000,"Maprange":1},{"Isuid":false,"Isgid":true,"H
ostid":166537,"Nsid":1001,"Maprange":64535}]'
  volatile.last_state.power: STOPPED
```

export image from container

[wiki, lxd, 'profile, network', apache, vlan]

Vlan attach

```
apt-get install vlan
```

```
sudo modprobe 8021q
```

```
sudo vconfig add eth1 10
```

```
sudo ip addr add 10.0.0.1/24 dev eth1.10
```

```
ip addr del 10.22.30.44/16 dev eth0
```

```
sudo ip link set up eth1.10
```

```
 sudo su -c 'echo "8021q" >> /etc/modules'
```

```
auto eth1.10
iface eth1.10 inet static
    address 10.0.0.1
    netmask 255.255.255.0
    vlan-raw-device eth1
```

Send file to your new host

On image hosts

```
lxc publish --force 'name of container" --alias 'new name'
```

example

```
lxc publish --force 'lxc-limesurvey' --alias 'lxc-docuwiki'
```

Export image

```
lxc image  export 'new name'
```

Output is in efaa243331f0a7c175376edaf796545a01ad09bb47f25a297b798e09fe66ee66.tar.gz Show size of export

```
du -h
efaa243331f0a7c175376edaf796545a01ad09bb47f25a297b798e09fe66ee66.tar.gz
```

check sum of image

```
md5sum
efaa243331f0a7c175376edaf796545a01ad09bb47f25a297b798e09fe66ee66.tar.gz >
exportmd5.txt

cat exportmd5.txt | nc 10.18.49.73 1234

cat efaa243331f0a7c175376edaf796545a01ad09bb47f25a297b798e09fe66ee66.tar.gz
| nc 10.18.49.73 1234
```

NB: 10.18.49.73 is your new lxd host

1234 is a free port

Transfer image and checksum to new LXD host

```
nc -l 1234 >
efaa243331f0a7c175376edaf796545a01ad09bb47f25a297b798e09fe66ee66.tar.gz
nc -l 1234 > exportmd5.txt
```

check file

```
md5sum
efaa243331f0a7c175376edaf796545a01ad09bb47f25a297b798e09fe66ee66.tar.gz
md5sum -c exportmd5.txt
```

Import image to new LXD host

```
 lxc image import
efaa243331f0a7c175376edaf796545a01ad09bb47f25a297b798e09fe66ee66.tar.gz --
alias lxc-docuwiki
```

Transferring image: 100%

```
lxc launch image_name container_name
```

Creating container_name Starting container_name

In some instances the publish command may lead to a split xz tar-ball — but both formats are supported. Simply import the meta-data and rootfs components with

```
  lxc image import <metadata tarball> <rootfs tarball> --alias image_name
```

Edit LXD default profile: networking

Put lxc network interface to host network

```
lxc stop lxc-docuwiki
lxc profile device set default eth0 parent ens3
lxc profile device set  default eth0 nictype macvlan
service lxd restart
service lxd-containers restart
```

launch your container

```
lxc start lxc-docuwiki
lxc exec lxc-docuwiki /bin/bash
```