LXC

```
apt-get install lxc
```

Container erstellen

```
lxc-create -n <name> -t debian -- -r buster #stretch
#/usr/share/lxc/templates/lxc-debian -r stretch -a armhf
```

```
lxc-create -n <name> -t ubuntu -- -r bionic
#/usr/share/lxc/templates/lxc-debian -r bionic -a armhf
```

alternativ vorhandenen Container (Ordner mit Namen in /var/lib/lxc) kopieren und config anpassen (lxc.rootfs, lxc.utsname, lxc.mount.entry)

nach dem Erstellen muss das Root-Passwort gesetzt werden

```
lxc-start -n <name>
lxc-attach -n <name> passwd
```

Konfiguration

/var/lib/lxc/systemname/config

```
# Template used to create this container: /usr/share/lxc/templates/lxc-
debian
# Parameters passed to the template: -r stretch -a armhf
# Template script checksum (SHA-1): 127e2020d76da79709d5e4e0c7e347f40a6a793b
# For additional config options, please look at lxc.container.conf(5)

# Uncomment the following line to support nesting containers:
#lxc.include = /usr/share/lxc/config/nesting.conf
# (Be aware this has security implications)

#lxc.network.type = empty
lxc.rootfs = /var/lib/lxc/stretch-web/rootfs
lxc.rootfs.backend = dir

# Common configuration
lxc.include = /usr/share/lxc/config/debian.common.conf

# Container specific configuration
lxc.tty = 4
lxc.utsname = stretch
lxc.arch = armhf
```

```
lxc.start.auto = 1

#lxc.start.delay = 0 (in seconds)
lxc.start.delay = 5
#lxc.start.order = 0 (higher means earlier)
#lxc.start.order = 0

lxc.network.type = veth
lxc.network.link = lxcbr0
lxc.network.flags = up
lxc.network.ipv4 = 10.0.3.10/24
lxc.network.ipv4.gateway = auto

#/var/www
#lxc.mount.entry = /path/to/folder/on/host /path/to/mount/point none bind 0
0
lxc.mount.entry = /var/www /var/lib/lxc/stretch-web/rootfs/var/www/ none
bind 0 0
```

bootstrapped Dateisystem in /var/lib/lxc/systemname/rootfs/ (wenn nicht über Template erzeugt)

in die /etc/network/interfaces:

```
auto eth0
iface eth0 inet manual
```

LXCBR0

/etc/network/interfaces:

```
auto lxcbr0
iface lxcbr0 inet static
        bridge_ports none
        bridge_fd 0
        bridge_maxwait 0
        address 10.0.3.1
        netmask 255.255.255.0
```

Portforwarding

```
${ipt} -t nat -A PREROUTING ! -i ppp0 -m addrtype --dst-type LOCAL -p tcp --
dport 80 -j DNAT --to-destination 10.0.3.10:80
```

dies leitet den Port 80 bei Zugriff auf alle Schnittstellen (außer ppp0) auf die IP-Adresse des LXC-Containers weiter zum Vergleich...alle Ports (inkl. ppp0) hier für https:

```
${ipt} -t nat -A PREROUTING -m addrtype --dst-type LOCAL -p tcp --dport 443
-j DNAT --to-destination 10.0.3.10:443
```

Bedienung

starten/stoppen

```
lxc-start -n <name>
lxc-stop -n name
```

betreten / Befehle ausführen

```
lxc-console -n <name>
#strg+a,q zum verlassen
lxc-attach -n <name> -- <befehl>
```

Kommunikation container/host

dieses habe ich via ssh-pubkey realisiert

```
#Erstellen eines neuen Schlüssels in dem container
ssh-keygen -b 4096
#public-key auf den host übertragen
ssh-copy-id -i .ssh/id_rsa.pub user@host
#testen vom container aus
ssh -i .ssh/key_rsa user@host
```

Host

/home/frank/.ssh/authorized_keys:

```
command="/home/frank/hostinfo.sh" ssh-rsa AAAAB3N...
```

/home/frank/hostinfo.sh:

```
#!/bin/bash

line=$1
read line

case $line in
  "info")
    hostname
  ;;
  "diskspace")
    df -h |grep -v tmpfs
  ;;
  #...
```

```
esac
```

Container

/usr/local/bin/hostinfo.sh

```
#!/bin/bash
#echo $0 $1
SSHOPT=""
if [ "$USER"=="www-data" ];then
  SSHOPT="-o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no"
fi

if [ -n "$1" ]; then
  echo "$1" | ssh -vvv ${SSHOPT} frank@192.168.0.10 -i /etc/ssh/id_rsa_cmd
fi
```

Webserver-Integration

/etc/sudoers (damit web-user hostinfo als user frank ausführen kann (sonst funktioniert ssh nicht))

```
www-data ALL=(frank) NOPASSWD: /usr/local/bin/hostinfo.sh
```

index.php

```
$ret="";
echo "<pre>";
$lastline=system ("sudo -u frank /usr/local/bin/hostinfo.sh info",$ret);
//alternativ
$command="sudo -u frank /usr/local/bin/hostinfo.sh diskspace";
$output = shell_exec($command);
$output = explode(PHP_EOL, $output);
print_r($output);
//  echo "$lastline,$ret<br>";
echo "</pre>";
```

Autologin (qnap)

/share/Virtual/container-station-data/lib/lxc/debian-stretch/rootfs/lib/systemd/system/container-getty@.service

```
[Service]
ExecStart=-/sbin/agetty -a root --noclear --keep-baud pts/%I
115200,38400,9600 $TERM
```

lxc2 => lxc3

bei Fehlermeldung

Unknown configuration key „lxc.rootfs“

```
lxc-update-config -c /var/lib/lxc/buster-web/config
```

grundsätzlich ändert sich folgendes:

```
< lxc.rootfs = /var/lib/lxc/buster-web/rootfs
< lxc.rootfs.backend = dir
---
> lxc.rootfs.path = /var/lib/lxc/buster-web/rootfs

< lxc.tty = 4
< lxc.utsname = buster
---
> lxc.tty.max = 4
> lxc.uts.name = buster

< lxc.network.type = veth
< lxc.network.link = lxcbr0
< lxc.network.flags = up
< lxc.network.ipv4 = 10.0.3.10/24
< lxc.network.ipv4.gateway = auto
---
> lxc.net.0.type = veth
> lxc.net.0.link = lxcbr0
> lxc.net.0.flags = up
> lxc.net.0.ipv4.address = 10.0.3.10/24
> lxc.net.0.ipv4.gateway = auto
```

debug

```
lxc-start -Fn buster-web -o debug -l debug
```

foreground-modus (-F) und debug

kein login-prompt mit lxc-console

beim Erstellen eines bullseye-containers ist mir das aufgefallen

mit lxc-attach (lxc-attach -n NAME – login) oder „lxc-console -n NAME -t 0“ funktionierte es

```
# mknod /dev/tty0 c 4 0
# systemctl restart getty@tty1.service
```

```
# systemctl status getty@tty1.service
```

https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=907615#53