=== LXD containers get IP addresses from LAN DHCP Server & Static ===

https://www.ubuntupit.com/how-to-configure-and-use-network-bridge-in-ubuntu-linux/

https://blog.simos.info/how-to-make-your-lxd-containers-get-ip-addresses-from-your-lan-using-a-bridge/

https://fakrul.wordpress.com/2020/05/08/lxd-containers-get-ip-addresses-from-lan-dhcp-server/

https://www.cyberciti.biz/faq/install-lxd-on-ubuntu-20-04-lts-using-apt/

https://belajarlinux.id/cara-install-lxdui-di-ubuntu-20-04-lts/

{{:linux:lxc.png?800|}}

By default, all containers run hidden in a private network on the host. The containers are not accessible from the local network, nor from the Internet. However, they have network access to the Internet through the host.

It would be great to have LXC containers getting from local DHCP server so that anyone from the network can connect to the container.

1. First we will create a bridge interface and add our physical interface (in the example it's enp1s0) to the bridge:

sudo vi /etc/netplan/00-installer-config.yaml

```
network:
  version: 2
  renderer: networkd

  ethernets:
    enp1s0:
      dhcp4: false
      dhcp6: false

  bridges:
    bridge0:
      interfaces: [enp1s0]
      addresses: [192.168.99.252/24]
      gateway4: 192.168.99.1
      nameservers:
        addresses:
```

```
        - 1.1.1.1
        - 8.8.8.8
      parameters:
        stp: true
        forward-delay: 4
      dhcp4: no
```

Next apply the config

```
sudo netplan apply
```

2. Now install LXD

```
sudo apt install lxd
```

3. By default LXD will create a profile with name “default”. For us we will create a new profile name “bridgeprofile” and add the basic config:

```
sudo lxc profile create bridgeprofile
Profile bridgeprofile created

sudo cat <<EOF | lxc profile edit bridgeprofile
> description: Bridged networking LXD profile
> devices:
>   eth0:
>     name: eth0
>     nictype: bridged
>     parent: bridge0
>     type: nic
> EOF
```

Verify the profile:

```
 * # lxc profile list
 * +---------------+---------+
 * |     NAME      | USED BY |
 * +---------------+---------+
 * | bridgeprofile | 0       |
 * +---------------+---------+
 * | default       | 0       |
 * +---------------+---------+
```

4. Now generate lxd config using command “sudo lxd init” and select the appropriate options

```
r  * oot@lxd-home:/home/fakrul# lxd init
 * Would you like to use LXD clustering? (yes/no) [default=no]:
 * Do you want to configure a new storage pool? (yes/no) [default=yes]:
```

```
  * Name of the new storage pool [default=default]:
  * Name of the storage backend to use (btrfs, dir, lvm, zfs, ceph) [default=zfs]:
  * Create a new ZFS pool? (yes/no) [default=yes]:
  * Would you like to use an existing block device? (yes/no) [default=no]:
  * Size in GB of the new loop device (1GB minimum) [default=46GB]:
  * Would you like to connect to a MAAS server? (yes/no) [default=no]:
  * Would you like to create a new local network bridge? (yes/no) [default=yes]: no
  * Would you like to configure LXD to use an existing bridge or host interface? (yes/no)
[default=no]:
  * Would you like LXD to be available over the network? (yes/no) [default=no]:
  * Would you like stale cached images to be updated automatically? (yes/no) [default=yes]
  * Would you like a YAML "lxd init" preseed to be printed? (yes/no) [default=no]:
```

5. Edit the profile and add the storage pool. You can add any other cloud-init config if you want. For example I have added my ssh-public key and related information

```
sudo lxc profile show bridgeprofile
config:
  user.user-data: |
    #cloud-config
    users:
      - name: fakrul
        ssh_authorized_keys:
          - ssh-rsa
    *
*AAAAB3NzaC1yc2EAAAADAQABAAABAQCwkqkOEoYAolYMIkHozxc5eY6gjUhvqCbJj9BDRv
sWL26Rm2Yg2aCcAGaiGo+a1ADX8Ty6Ed0s+jZtEaoTEKVCSlCPZ5BB03B80Wao1/RjjoVho12
zf9r9lR5dz1n79fysCj43LRgJR7qb0Inpq4WyIx2mm4hq4kryWj5+cBm1LYnpIUf4LqCl3YzY2tKVX
rpVqbrJ5SIdy7FD6FxZIdpdwGJ7b+uUv0R/zSfVMZuU1+DIxyO9la0ZhFL4I/5H9Tb348tLCjR7X//
Tu+zSceTmg51ATWHvsSfRwOL/bUWRjMSr9swbXzwweGcw6k15VJ5sYiBFGDla5almjmXAfSx
D aminul@rajshahionline.com
        sudo: ALL=(ALL) NOPASSWD:ALL
        groups: sudo
        shell: /bin/bash
        passwd: $1$hUDF3Apy$jJztHAqAE8qxMX3yl0rnK/
        gecos: Fakrul Alam
    # Update timezone
    locale: en_US.UTF-8
    timezone: Australia/Sydney
    # Update apt database on first boot (run 'apt-get update').
    package_update: true
    # Install required packages
    packages:
      - whois
```

```
      - inetutils-traceroute
      - traceroute
      - net-tools
description: Bridged networking LXD profile

devices:
  eth0:
    name: eth0
    nictype: bridged
    parent: bridge0
    type: nic
  root:
    path: /
    pool: default
    type: disk
name: bridgeprofile
```

6. Now you can deploy instances with “bridgeprofile”

"lxc launch -p bridgeprofile ubuntu:18.04 graylog"

7. The instance “graylog” will get IP address from the LAN DHCP server.

This is an issue with the name resolutions (DNS configuration) in the container.

Can you verify that you have tried the above with the container images ubuntu:18.04 or ubuntu:20.04?
There are more container images in the images: repository, including Ubuntu images. If you have selected a Debian or Fedora container image, you would need to look above in this post the separate LXD profiles for them.

You can ping the Internet (the IP address for 1.1.1.1) but not hostnames (DNS issues). The cloud-config section above has a setting to enable DNS for you, and use one of the public DNS servers (8.8.8.8 is provided by Google).

First, can you run the following command in the container? It will try to make a name resolution using specifically the Google DNS server. If that works, then DNS resolutions are not blocked somehow. You should get the IP address of http://www.google.com in the output, both IPv4 and IPv6.

host www.google.com 8.8.8.8

Second, if the above works, verify that the cloud-init information managed to get parsed correctly, and can be found in the container.

```
$ cat /etc/netplan/50-cloud-init.yaml
# This file is generated from information provided by the datasource.  Changes
# to it will not persist across an instance reboot.  To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
    ethernets:
        eth0:
            addresses:
            - 192.168.1.200/32
            nameservers:
                addresses:
                - 8.8.8.8
                search: []
            routes:
            -   on-link: true
                to: 0.0.0.0/0
                via: 169.254.0.1
    version: 2
$
```

Third, verify that the network manager (in the case of Ubuntu it is NetworkManager) is aware of the DNS server. The last lines mention the public Google DNS server, 8.8.8.8.

```
$ systemd-resolve --status
...
Link 2 (eth0)
      Current Scopes: DNS
DefaultRoute setting: yes
       LLMNR setting: yes
MulticastDNS setting: no
  DNSOverTLS setting: no
      DNSSEC setting: no
    DNSSEC supported: no
  Current DNS Server: 8.8.8.8
         DNS Servers: 8.8.8.8
$
```