LXC container as a 44Net VPN & Subnet Router

Mark Phillips, NI2O 06/2024 V1.2

Contents

Abstract

This document aims to walk the reader through the setting up of an LXC container for use as a 44Net VPN Subnet Router. It is a continuation of the “LXC container as a 44Net VPN Router/Firewall” document. While this document speaks to ProxMox and LXC systems it may be useful for other Linux/Mac implementations as the steps and commands are almost identical.

Assumptions

Familiarity with ProxMox Virtual Environment
Familiarity with Linux (Debian) networking
Networking basics

Requirements

Functioning Proxmox LXC environment
Functioning LAN with Internet connectivity
Functioning LXC VPN Router/Firewall container

Apply for a subnet

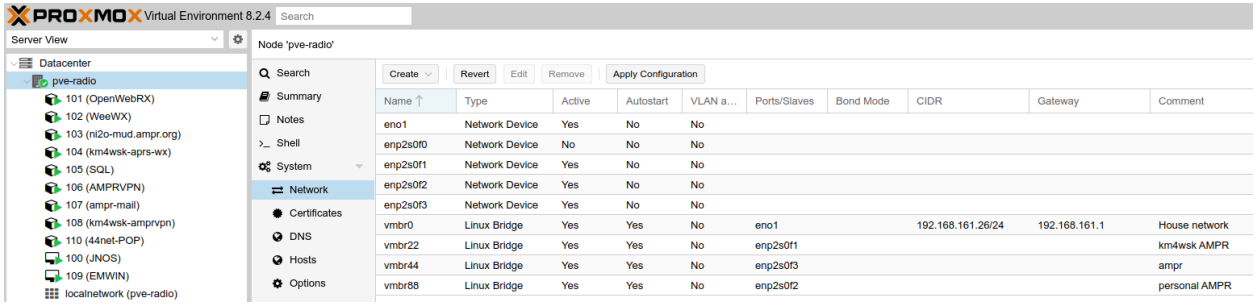
Using the pop.44net.cloud portal apply for a subnet suitable for your purposes. In the case of this document we will be using the 44.33.2.104/29 subnet. This will allow us a range of 8 IP addresses with 6 usable. But it's really 5 usable as we must take one of the usable addresses to be the router IP.

Modify the LXC container

To turn our container into a router we need to make at least the following modifications. A router must have an input port, output port and possibly several local ports. Each port must have a unique IP address. To this end add a second ethernet port to your container. This can be a real Ethernet port (assuming you have other physical machines to add to the subnet) or a virtual Ethernet port (assuming your machines are all virtual) or a mix of both. We'll be doing the mix.

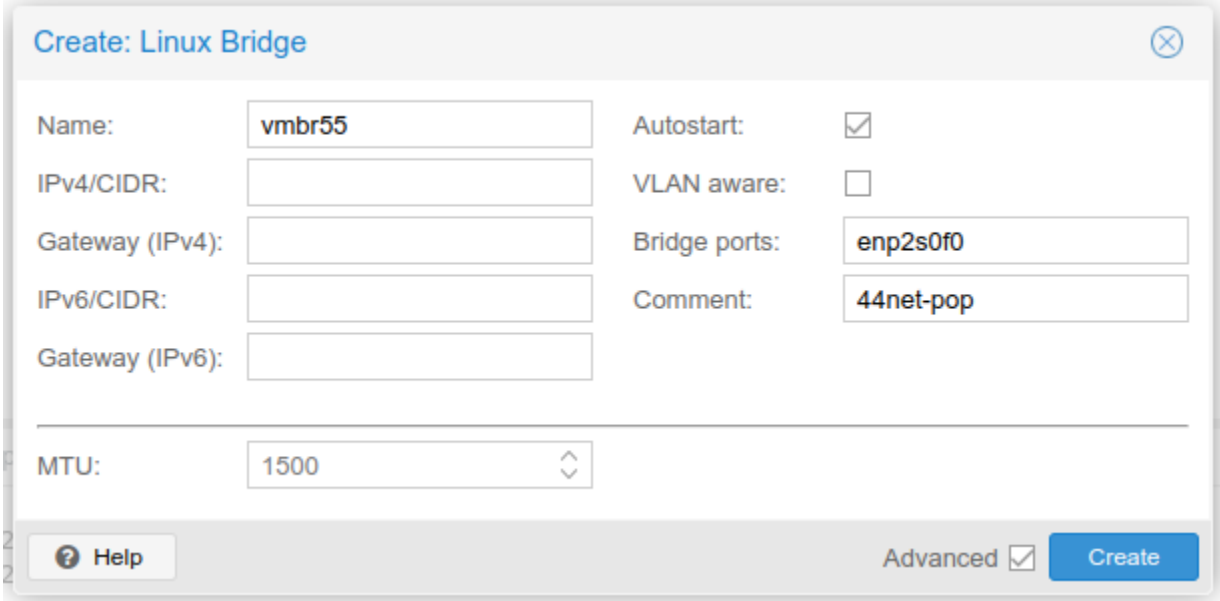
Add Bridged Ethernet port

First we must set up a bridge Ethernet port to the underlying Proxmox server. Follow the steps below;



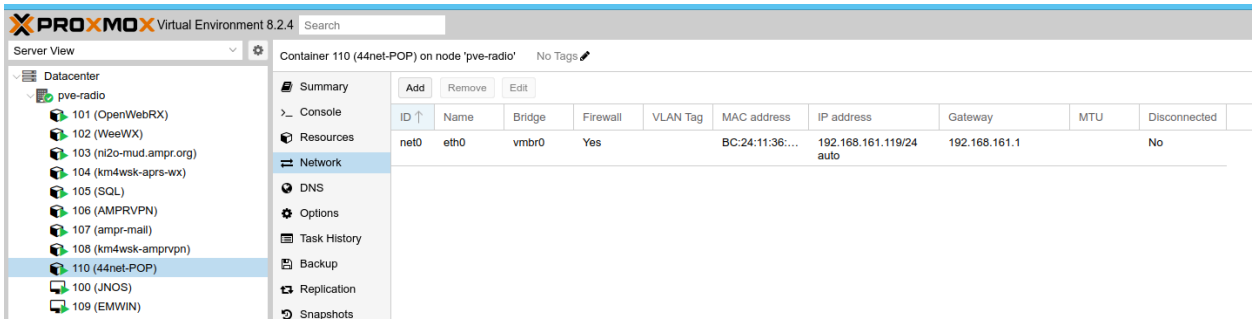
From the network panel as seen above create a Linux Bridge. If you want the resulting network to appear on an actual Ethernet port for use with physical machines, select an unused physical port on the server. Otherwise leave the “Bridge ports:” box empty for virtual LAN only.

Create > Linux Bridge (don’t forget to click “Apply configuration” afterwards!)

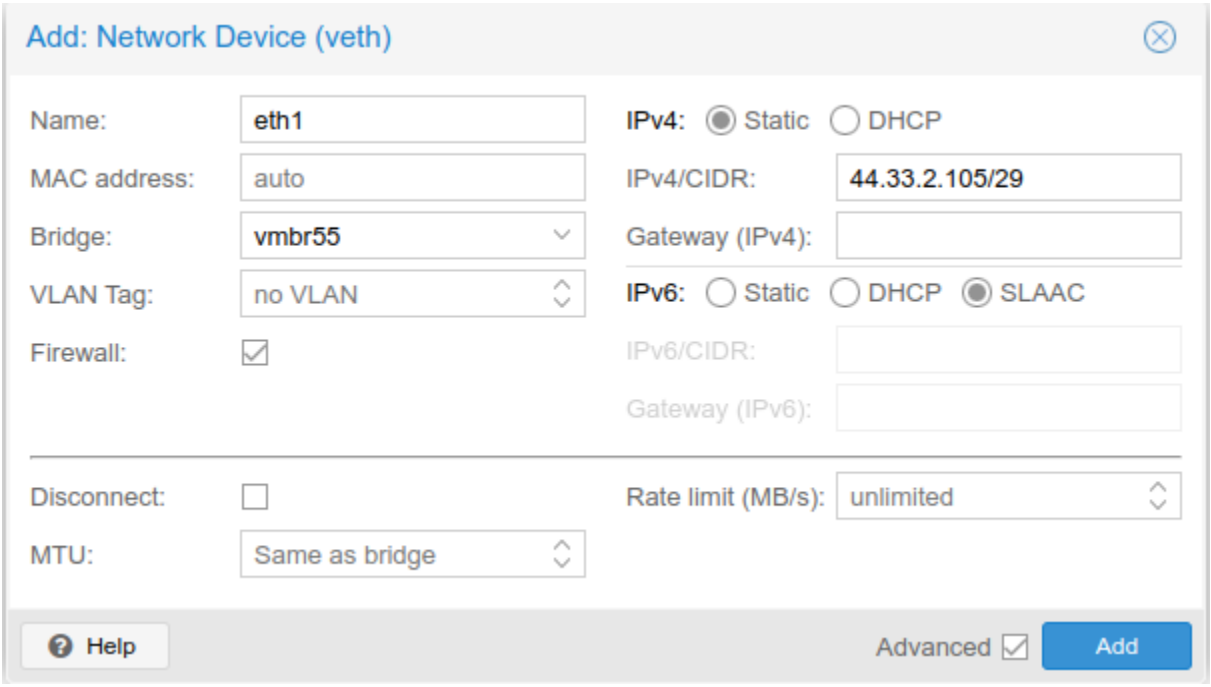


Add a virtual Ethernet interface

Now move on to the actual container itself. We previously set up a LAN interface that connects to our house LAN. We need to add a second one for our 44Net Subnet



Populate the fields as below. Do not populate the “Gateway” box.



On the console, perform an “ifconfig” to see if the new network interface has appeared

```
ifconfig
```

```
root@44net-POP:~# ifconfig
44net-pop: flags=209<UP,POINTOPOINT,RUNNING,NOARP>  mtu 1420
        inet 44.33.1.32  netmask 255.255.255.255  destination 44.33.1.32
        inet6 2a0a:bb06:1::d  prefixlen 128  scopeid 0x0<global>
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 64917  bytes 6534364 (6.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 55247  bytes 7123048 (6.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.161.119  netmask 255.255.255.0  broadcast 192.168.161.255
        inet6 2600:4040:7c7a:c738:be24:11ff:fe36:a48e  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::be24:11ff:fe36:a48e  prefixlen 64  scopeid 0x20<link>
        ether bc:24:11:36:a4:8e  txqueuelen 1000  (Ethernet)
        RX packets 568086  bytes 69423485 (66.2 MiB)
        RX errors 0  dropped 113  overruns 0  frame 0
        TX packets 58381  bytes 9605405 (9.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 44.33.2.105  netmask 255.255.255.248  broadcast 0.0.0.0
        inet6 fe80::be24:11ff:fe70:3acd  prefixlen 64  scopeid 0x20<link>
        ether bc:24:11:70:3a:cd  txqueuelen 1000  (Ethernet)
        RX packets 2  bytes 180 (180.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23  bytes 1258 (1.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

A final check that the interface is in fact installed requires us to see if there's a route to the subnet we added to the interface

```
route -n
```

```
root@44net-POP:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.161.1   0.0.0.0         UG    0      0        0 eth0
44.33.2.104     0.0.0.0         255.255.255.248 U     0      0        0 eth1
192.168.161.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

If you have any physical machines that are to be connected to this network/subnet connect them to the physical network port we attached the Bridge to. Issue the physical machines with IP addresses, subnets AND gateways. The Gateway address for our demonstration network is the address of the port we added to the VPN container (44.33.2.105/29).

Perform system updates

Now that we have a new interface installed it would be a good idea to make sure our system is up-to-date with all the relevant patches and upgrades that may have occurred since we last built the container.

```
apt update && apt upgrade -y
```

Route incoming connections

If you did not do it previously you will need to instruct Linux to forward the data it gets in from the VPN to the relevant place. This will be determined by the routing table which in our case has 3 routes: the internal network, the new subnet and to the VPN.

```
sysctl -w net.ipv4.ip_forward=1
```

Firewall considerations

It is highly recommended that you should firewall your VPN link. It is not within the scope of this document to explain the ins and outs of any firewall but the below settings for the UFW firewall package will allow a basic secure firewall on your VPN container. It will allow traffic to be firewalled ONLY to the VPN container itself. Traffic forwarded to the subnet will require firewalling on those devices too. Further information about the Uncomplicated Firewall (UFW) can be found here https://manpages.ubuntu.com/manpages/trusty/man8/ufw.8.html

Firewall incoming connections

By default, all incoming connections will be stopped by the UFW firewall application. To allow incoming connections to the new subnet add the below instructions. They will allow traffic from

the VPN and also from your lan to the new subnet. Make sure to change the IP ranges accordingly.

```
ufw route deny in on 44net-pop to 192.168.161.0/24
ufw route deny in on eth1 to 192.168.161.0/24
ufw route allow in on eth0 to 44.33.2.104/29
ufw route allow in on 44net-pop to 44.33.2.104/29
ufw route allow in from 44.33.2.104/29 to 44net-pop
```

```
root@44net-POP:~# ufw status
Status: active

To                         Action      From
--                         ------      ----
192.168.161.0/24           DENY FWD    Anywhere on 44net-pop
192.168.161.0/24           DENY FWD    Anywhere on eth1
44.33.2.104/29             ALLOW FWD   Anywhere on eth0
44.33.2.104/29             ALLOW FWD   Anywhere on 44net-pop
Anywhere                   ALLOW FWD   44.33.2.104/29
```

The above rules did the following ..

Denied all incoming access from the VPN (44net-pop) to the LAN (eth0)
Denied all packets from the new interface (eth1) to the LAN (eth0)
Allowed LAN (eth0) access to the new subnet (eth1)
Allowed VPN (44net-pop) access to the new subnet (eth1)
Allowed the new subnet (eth1) out over the VPN (44net-pop)

NAT outgoing connections (optional)

Your VPN can be used as both a router for allowing access to/from your subnet and as a gateway for your personal desktop equipment. For your subnet, no NAT is required but for your personal equipment that is connected to your house LAN you will need to NAT those devices.

We need to make sure that we NAT only those devices that appear on a particular network interface. If you built your container using the “LXC Container as a 44Net VPN Router/Firewall” document AND you enabled the NAT features then you need do no more.

To enable this feature it is necessary to further edit the VPN config file;

```
nano /etc/wireguard/44net-pop.conf
```

Add the following lines below the last line in the [Interface] section;

```
PostUp = iptables -A FORWARD -i eth0 -j ACCEPT; iptables -t nat -A
POSTROUTING -o 44net-pop -j MASQUERADE

PostDown = iptables -D FORWARD -i eth0 -j ACCEPT; iptables -t
nat -D POSTROUTING -o 44net-pop -j MASQUERADE
```

As shown in the example below;

```
[Interface]
PrivateKey = 8H8ey27QlQq68/H5bOyfICzDMbxlE2fWo6t1mo4nfnU=
Address = 44.33.1.32/32, 2a0a:bb06:1::d/128
DNS = 1.1.1.1, 1.0.0.1
PostUp = iptables -A FORWARD -i eth0 -j ACCEPT; iptables -t nat
-A POSTROUTING -o 44net-pop -j MASQUERADE
PostDown = iptables -D FORWARD -i eth0 -j ACCEPT; iptables -t
nat -D POSTROUTING -o 44net-pop -j MASQUERADE

[Peer]
PublicKey = CCK2lhIIo1BpAzqfQeUjPhVjFyslZ/R9Vh0AU1LJ218=
PresharedKey = WY8OUQRiErfe7glOHzwon8oXGj5EvSFj1J9wt0Zfdpo=
Endpoint = 45.32.220.92:12345
PersistentKeepalive = 10
AllowedIPs = 0.0.0.0/0, ::/0
```

Restart the VPN and also the firewall (if any)

```
wg-quick down 44net-pop
wg-quick up 44net-pop
ufw reload
```