PXE server setup

This guide describes how to install and configure a preboot execution environment (PXE) server that you can use to deploy EVE-OS to a fleet of edge nodes.

Prerequisites and considerations

- Your PXE server's machine must be running Ubuntu 22.04 or later.
- We recommend that you run your PXE server on a network dedicated to imaging.

Setting up a PXE server for EVE-OS deployments

The following procedures are discrete, but each one sets up the next. If you haven't begun setting up your PXE server, complete them sequentially. Otherwise, you can skip to the procedure that you need.

After you start dnsmasq on your network in the last procedure, your PXE server will be ready.

Note that the following procedures use EVE version 8.12.0 only as an example. Substitute your preferred version where necessary. Refer to the list of EVE-OS releases for options.

Download the EVE-OS source files

In this procedure, you'll download EVE source files so that your PXE server will be able to deliver them without internet connectivity.

1. Make a new directory for your EVE files within a tftboot directory.

```
mkdir /tftpboot/eve
```

2. Copy the following code and paste it into a script file. For example, "get-eve-pxe-files.sh". This script will download all of the required files for the version of EVE specified in the EVE_VERSION variable.

```
#!/bin/bash

EVE_VERSION='10.4.8-lts'
EVE_FILES=('amd64.initrd.bits' 'amd64.initrd.img' 'amd64.installer.img'
'amd64.ipxe.efi' 'amd64.ipxe.efi.cfg' 'amd64.ipxe.efi.ip.cfg' 'amd64.kernel'
'amd64.rootfs.img')
```

```
if [ ! -d /tftpboot/eve/$EVE_VERSION ];
then
        sudo mkdir -p /tftpboot/eve/$EVE_VERSION
fi

for i in "${EVE_FILES[@]}"
do
        sudo wget [https://github.com/lf-
edge/eve/releases/download/$EVE_VERSION/$i](https://github.com/lf-
edge/eve/releases/download/$EVE_VERSION/$i) -P /tftpboot/eve/$EVE_VERSION
```

3. Change the permissions of the download script to make it executable.

```
chmod +x get-eve-pxe-files.sh
```

4. Run the download script.

```
./get-eve-pxe-files.sh
```

Set a static IP address for your PXE server

1. Create a config file called 00-installer-config.yaml

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

2. Paste the following code into your installer config file.

```
network:
  ethernets:
        ens160:
          dhcp4: true
  ens192:
        addresses:
          - 192.168.1.10/24
        nameservers:
          addresses: [8.8.8.8, 8.8.4.4]
version: 2
```

3. Apply your new network configuration.

```
sudo netplan apply
```

4. Run the following command.

```
ip addr sh
```

5. Use the output of the previous command to verify that your IP addresses are assigned to the correct interfaces and can ping each interface.

Configure your tftpboot directory structure

1. Install the ipxe server software.

```
sudo apt install ipxe –y
```

2. Create a tftp boot directory (if you don't already have one) for your PXE and EVE files.

```
sudo mkdir /tftpboot
```

3. Copy the tftpt boot files from the ipxe installation into your tftpboot directory.

```
sudo cp /usr/lib/ipxe/{undionly.kpxe,ipxe.efi} /tftpboot
```

4. Make a new directory called menu in your tftboot directory.

```
sudo mkdir /tftpboot/menu
```

5. Create your boot file.

```
sudo vi /tftpboot/menu/boot.ipxe
```

6. Copy the following code into your boot file.

```
#!ipxe
# dhcp
```

```
#
# Uncomment ntp lines for devices without RTC (RPI for example)
# echo Getting the current time from ntp...
# :retry_ntp
# ntp pool.ntp.org || goto retry_ntp
#
# you may want to add the following to the kernel command line arguments:
#   * eve_install_disk=XXX (e.g. XXX=mmcblk0)
#   * eve_install_server=XXX (e.g. XXX=zedcloud.hummingbird.zededa.net)
#   * eve_persist_disk=XXX (e.g. XXX=mmcblk0, you can set multiple values
#     here with comma delimiter to use multiple disks).
#
# chain --autofree https://github.com/lf-
edge/eve/releases/download/1.2.3/ipxe.efi.cfg
# set url https://foo.bar/
# set url https://github.com/lf-edge/eve/releases/download/8.12.0/amd64.

set url tftp://192.168.1.10/eve/8.12.0/amd64.
set console console=ttyS0 console=ttyS1 console=ttyS2 console=ttyAMA0
console=ttyAMA1 console=tty0
set eve_args eve_soft_serial=${mac:hexhyp} eve_reboot_after_install
set installer_args root=/initrd.image find_boot=netboot overlaytmpfs
fastboot

# a few vendor tweaks
iseq ${smbios/manufacturer} Huawei && set console console=ttyAMA0,115200n8
||
iseq ${smbios/manufacturer} Huawei && set platform_tweaks pcie_aspm=off
pci=pcie_bus_perf crashkernel=auto ||

:start
menu PXE Boot Options
item eve-8.12.0-amd64 EVE 8.12.0 AMD64
item shell iPXE shell
item exit  Exit to BIOS
choose --default eve-8.12.0-amd64 --timeout 10000 option && goto ${option}

:eve-8.12.0-amd64
kernel ${url}kernel ${eve_args} ${installer_args} ${console}
${platform_tweaks} initrd=amd64.initrd.img initrd=amd64.installer.img
```

```
initrd=amd64.initrd.bits initrd=amd64.rootfs.img initrd=initrd.bits
initrd=rootfs.img
initrd ${url}initrd.img
initrd ${url}installer.img
initrd ${url}initrd.bits
initrd ${url}rootfs.img

boot

:shell
shell

:exit
exit
```

Configure dnsmasq for your network

1. Install dnsmasq.

```
sudo apt install -y dnsmasq
```

2. Create the dnsmasq configuration file.

```
sudo vi /etc/dnsmasq.conf
```

3. Copy the following content into your configuration file.

```
# enable logs if required
#log-queries
#lo-dhcp

# disable DNS server
port=0

# listen on PXEBOOT
listen-address=192.168.1.10
interface=ens192

# enable built-in tftp server
```

```
enable-tftp
tftp-root=/tftpboot

# DHCP range 192.168.1.100 - 192.168.1.250
dhcp-range=192.168.1.100,192.168.1.250,255.255.255.0,24h

# Default gateway
dhcp-option=3,192.168.1.1

# Domain name - zededalab.net
dhcp-option=15,zededalab.net

# Broadcast address
dhcp-option=28,192.168.1.255

# Set interface MTU to 9000 bytes (jumbo frame)
# Enable only when your network supports it
# dhcp-option=26,9000

# Tag dhcp request from iPXE
dhcp-match=set:ipxe,175

# inspect the vendor class string and tag BIOS client
dhcp-vendorclass=BIOS,PXEClient:Arch:00000

# 1st boot file - Legacy BIOS client
dhcp-boot=tag:!ipxe,tag:BIOS,undionly.kpxe,192.168.1.10

# 1st boot file - EFI client
# at the moment all non-BIOS clients are considered

# EFI client
dhcp-boot=tag:!ipxe,tag:!BIOS,ipxe.efi,192.168.1.10

# 2nd boot file
dhcp-boot=tag:ipxe,menu/boot.ipxe
```

4. Start dnsmasq.

```
sudo systemctl dnsmasq
```