Deploying Linux using PXE

What is PXE?

PXE (Preboot eXecution Environment) is a method of deploying an operating system to a computer over the network. I think it's great: you don't need to burn CD's or DVD's anymore. The only thing you have to do is changing the boot order in the BIOS, and when the computer starts (mostly) press F12. The syslinux PXE also allows you to automate (basic) configuration during the deployment.

On this moment I already have a Windows virtual machine for deploying Windows to other computers, I'd like to have the same for CentOS8. In this article, I'll create a new virtual machine (VM) in Hyper-V, with CentOS8, to deploy CentOS to other machines. I'll do this in the GUI first, and script it afterwards. The scripts can be found on my git environment [1].

Let's start!

To get this environment, you'll need a virtual machine. Use the wizard to create a new VM in Hyper-V:
- Give it a name (f.e. LinuxPxe) and change the directory if necessary
- It is a generation 1 VM.
- Give it 2048 MB of startup memory, and checked the “Use Dynamic Memory for this virtual machine” checkbox,
- In the connection tab, give it a connection to the network (external switch).
- Create a virtual harddisk for this VM,
- Install an operating system from a bootable CD/DVD-ROM (I used CentOS-8-x86_64-1905-dvd1.iso for that, which is the latest version via the default download link for CentOS [2]).
- After clicking on Finish, right-click on the VM and change the number of CPU’s from 1 to 2 and click Ok.

<<Picture>>

After that, start the VM. In the configuration screens of CentOS,
- change the Time & Date to your own time zone, in my case f.e. Europe/Amsterdam
- change the Software Selection, choose Server (a GUI isn't needed for this VM)
- in Installation Destination, click on the disk of this VM twice, so you see a blue box under Local Standard Disks and you see a check mark on the disk
- in Network & Host Name, switch the ethernet (eth0) to On and change the hostname to LinuxPxe.

After that, click on "begin the installation". I never configure the root password in this screen, I just create a local user (in my case f.e. frederique) with administrator privileges. When the installation asks to do a reboot, turn off the VM and change the Hyper-V settings of this VM: under BIOS, change the boot order so that IDE comes first. After starting the system again, log on, sudo into root and do a yum update:

```
$ sudo -i
# yum update -y.
```

PXE under the hood

To understand why we need the software that we are configuring, it is useful to first look how PXE works. When you press F12 during the boot of a machine to start a network install, the BIOS will do a DHCP request on the network. The DHCP servers on the network will respond, each with their own DHCP options. One of those options might be the Full Qualified Domain Name (FQDN) or IP-address of the PXE server. The BIOS will accept or deny the DHCP offer based on these options.

When one of the DHCP offers suits the needs of the BIOS, that offer will be accepted. The DHCP server will confirm that the offer can be used and the BIOS will configure the network card to use the IP-address of the DHCP offer.

After that, the BIOS will start a Trivial File Transfer Protocol (TFTP) session with the PXE server. It is important to know, that TFTP uses a different protocol, with different port numbers than standard FTP [3]. The boot files are copied to the server and the BIOS will start the boot software. A menu will be presented, and after that the Linux kernel will be started.

The Linux kernel is just enough to download and start the installation software. The installation software is downloaded via FTP, and after downloading the software the installation is started, just as it would when it was started using an installation DVD.

It is also possible to use an extra parameter that tells where to download the anaconda configuration file, where Linux installation options can be used to configure Linux during the installation.

<<Picture>>

Installing software
Some of the software is only available via epel release, so first install the epel-release software and after that install the other packages:

```
# yum install epel-release -y
# yum install dhcp-server ftp vsftpd syslinux tftp tftp-server -y
```

DHCP
I choose to use a subset of the IP addresses that are used by the DHCP server that is part of my router. I know that this DHCP server will use addresses in the192.168.2 network, starting from one. I think I will be safe to use the numbers 200-210 in this network. The advantage of using the same network addresses as the network you are in, is that you don't need to use an extra router to use the internet during the installation.

The documentation in /usr/share/doc/syslinux has an example configuration, that I have adjusted to my environment. I didn't use a group for PXE servers (this DHCP server will only be used for PXE, so the options for PXE can be global). I also don't use fixed addresses (for setting up and testing), I used a range statement instead. The next-server address is the IP address of the LinuxPxe server. If you follow allong, you should replace the address that this server has in my environment (192.168.2.131) by the IP address that the LinuxPxe server has in your own environment. You can find it by using the command

```
# ip address show
```

I added the following content to the /etc/dhcp/dhcpd.conf file. 192.168.2.254 is the address of my home router, with a built-in DNS. Please check the bold options and change them if needed for your own environment:

allow booting;
allow bootp;

option domain-name "mydomain";
option subnet-mask **255.255.255.0**;
option broadcast-address **192.168.2.255**;

```
option domain-name-servers **192.168.2.254**;
option routers **192.168.2.254**;

next-server **192.168.2.131**;
filename "pxelinux.0";
subnet **192.168.2.0** netmask **255.255.255.0** {
        range dynamic-bootp **192.168.2.200 192.168.2.210**;
}
```

TFTP
Copy the startup files to the default tftp directory:

```
# cp /usr/share/syslinux/pxelinux.0 /var/lib/tftpboot
# cp /usr/share/syslinux/memdisk  /var/lib/tftpboot
# cp /usr/share/syslinux/menu.c32 /var/lib/tftpboot
# cp /usr/share/syslinux/mboot.c32 /var/lib/tftpboot
# cp /usr/share/syslinux/chain.c32 /var/lib/tftpboot
# cp /usr/share/syslinux/ldlinux.c32 /var/lib/tftpboot
# cp /usr/share/syslinux/libutil.c32 /var/lib/tftpboot
```

Use the following commands to mount the DVD and copy the kernel files from the DVD to the TFTP environment:

```
# mount /dev/cdrom /media

# mkdir /var/lib/tftpboot/networkboot
# cp /media/images/pxeboot/vmlinuz /var/lib/tftpboot/networkboot
# cp /media/images/pxeboot/initrd.img  /var/lib/tftpboot/networkboot
```

FTP
In the configuration file /etc/vsftpd/vsftpd.conf, allow anonymous ftp by configuring the keyword anonymous_enable=YES. Disable local users with local_enable=NO and don't allow updates in any form:

```
anonymous_enable=YES
local_enable=NO
write_enable=NO
```

In some cases, you might need to change the ports which are used by FTP for passive connections, because these ports have to be known by a firewall in your network. When this is the case, add the following lines and configure the firewall in your network to open the ports 8000-9000:

```
# Portnumbers used for passive ftp:
pasv_min_port=8000
pasv_max_port=9000
```

When something doesn't work as expected, it can be useful to use the debugging within VSFTP. Change the default xferlog_std_format to NO and add a line for log_ftp_protocol. When you configure this, you will see all ftp commands in the file /var/log/vsftpd.log .

```
xferlog_std_format=NO
log_ftp_protocol=YES
```

Now, copy the installation DVD to the /var/ftp/CentOS8 directory:

```
mkdir /var/ftp/CentOS8
cp -a /media/*            /var/ftp/CentOS8
cp -a /media/.treeinfo /var/ftp/CentOS8
```

PXE
The next step is to configure PXE. Create a directory (!) /var/lib/tftpboot/pxelinux.cfg . Within this directory, it is possible to create configuration files based on UUID, Mac address or (parts of the) IP address. In my case, I'll use the hexadecimal value of my network 192.168.2 without dots, so C0A802, as the name of the configfile [4]. In this way, this configfile will be used for all PXE installations by this server. When you want to do this differently, read the documentation of pxelinux (/usr/share/doc/syslinux).

```
# mkdir /var/lib/tftpboot/pxelinux.cfg
# vi /var/lib/tftpboot/pxelinux.cfg/C0A802
```

Add the following content to this file (mind that the ks=... part is behind the APPEND keyword, not on a new line):

```
DEFAULT menu.c32
PROMPT 0
TIMEOUT 30
LABEL CentOS 8 Server
MENU centos8s
KERNEL /networkboot/vmlinuz
APPEND initrd=/networkboot/initrd.img inst.repo=ftp://192.168.2.131/CentOS8
ks=ftp://192.168.2.131/centos8.cfg
```

Kickstart / Anaconda
The ks option in the PXE configuration file refers to the anaconda file. Fortunately, there is an anaconda file that was used by the configuration of the LinuxPxe virtual machine. This can be used as a base for the configuration file that is used in the PXE install. The default permissions for this file is readable and writable for user root and no permissions for group or others, this has to be changed so that anonymous users also can read it:

```
# cp ~root/anaconda-ks.cfg /var/ftp/centos8.cfg
# chmod a+r /var/ftp/centos8.cfg
```

There are some options that have to be changed in this file. We installed the LinuxPxe from DVD/CD-Rom and when we use PXE we want to install from the network. In newer versions of Anaconda it is no longer required to add a base url to the AppStream repo. This, however, still leads to an error, which can easily be solved by leaving the line repo --name=AppStream out. Last but not least, one of the nice features of Anaconda is to get the newest versions of the packages during the installation. To make this possible, change the /var/ftp/centos8.cfg file:

```
repo --name="AppStream" --baseurl=file:///run/install/repo/AppStream
# Use CDROM installation media
cdrom
```

into

```
repo --name=centos-updates --mirrorlist=http://mirrorlist.centos.org/?
release=$releasever&arch=$basearch&repo=BaseOS --cost=1000
repo --name=appstream-updates --mirrorlist=http://mirrorlist.centos.org/?
release=$releasever&arch=$basearch&repo=AppStream --cost=1000
repo --name=extras-updates --mirrorlist=http://mirrorlist.centos.org/?
release=$releasever&arch=$basearch&repo=Extras --cost=1000
# Use ftp installation media
url --url=ftp://192.168.2.131/CentOS8
```

It is also smart to add the options clearpart and zerombr, to be sure that the original disk will be wiped before an installation. Change the original clearpart statement: change

```
clearpart --none --initlabel
```

into:

```
clearpart --all
zerombr
```

You might find, that the network card of a newly installed system is disabled after reboot, even though you configured the card to be on. This can be solved by changing the configuration file /etc/sysconfig/network-scripts/ifcfg-eth0: the last parameter ONBOOT=no should be ONBOOT=yes. After this change, you can use **systemctl restart NetworkManager** to enable the network card.

When this is the case, the anaconda.cfg has the parameter --onboot=off on the line with the network parameters. You have to delete this parameter: change

```
network   --bootproto=dhcp --device=eth0 --onboot=off  --ipv6=auto --activate
```

into:

```
network   --bootproto=dhcp --device=eth0 --ipv6=auto --activate
```

Firewall
There have to be changes to the firewall to allow tftp and ftp traffic. When you use this PXE server for mutiple networks and you use a DHCP-proxy, also add the proxy-dhcp service:

```
# firewall-cmd --add-service=tftp --permanent
# firewall-cmd --add-service=ftp --permanent
# setsebool -P allow_ftpd_full_access 1

# firewall-cmd --add-service=proxy-dhcp --permanent

# firewall-cmd --reload
```

Services
To allow the tftp software to start, we need to configure tftp as a service. Add the file /etc/systemd/system/tftpd.service with the following content:

```
[Unit]
Description=TFTP Service
```

```
Documentation=man:in.tftpd

[Service]
ExecStart=/usr/sbin/in.tftpd -s -vvv -l -a :69 -r blksize -P /var/run/tftpd.pid /var/lib/tftpboot
ExecStop=/bin/kill -15 $MAINPID
PIDFile=/var/run/tftpd.pid

[Install]
WantedBy=multi-user.target
```

The parameters of in.tftpd are:

-s	secure: change rootdirectory to the directory in the commandline before allowing tftp
-vvv	very verbose logging
-l -a :69	listen on port 69 on all network adapters
-r blksize	refuse to accept the blksize option negotiation.

After that, start and enable the services:

```
# systemctl daemon-reload
# systemctl restart dhcpd
# systemctl enable dhcpd
# systemctl start tftpd
# systemctl enable tftpd
# systemctl restart vsftpd
# systemctl enable vsftpd
```

Testing...
Now, let's see if this works. First, try to get the file menu.c32 via tftp:

```
# tftp localhost
get menu.c32
quit
```

This should work well. Now, try ftp:

```
# ftp localhost
(using user-id ftp, password ftp)
```

When this works well, you can use cd CentOS8 to go into the CentOS8 directory and see the files with ls.

```
cd CentOS8
ls
get TRANS.TBL
quit
```

This also should work.

Now it's time to try PXE: create a new VM in Hyper-V, using the same Hyper-V parameters as before except for the name (use f.e. TestPxe) and the DVD drive: in the installation options, don't use a DVD drive, but select the last option ("Install an operating system from a network-based installation server") instead. In the settings of the newly created VM, not only change the number of

CPU's from 1 to 2, but also add an extra network card. The PXE starting is done using a so called "legacy network card", all the other networking is done using the newly created network adapter. This is something weird in the Hyper-V configuration: the legacy network adapter is not recognized when ftp is used to get the files from the ftp server. Connect this new network adapter also to the network (external switch).

Start the VM. When you configured everything as described above, there shouldn't be errors.

If there are errors, you might do some simple checks:
- Did you change the IP-address 192.168.2.131 and the network 192.168.2 in the configuration of the DHCP configuration file to addresses in your own network?
- Did you change the IP-address 192.168.2.131 in the configuration file /var/ftp/centos8.cfg?
- Did you change the name of the file C0A802 (which is hexadecimal for 192.168.2) into the hexadecimal value of your own network?
- Did you add an extra network card in the Hyper-V node TestPxe?
- Is there text in /var/log/vsftpd.log?
- The command journalctl -xe might also give more information

When the installation software asks to reboot, stop the VM and change the settings of the VM. In the BIOS settings, set the order to boot from IDE first. After that, start up the VM.

After installation
First, check that there are no updates available:

```
$ sudo -i
# yum update -y
```

I expect that no updates are available, because updates are downloaded during the installation (the three repo lines in the Anaconda-configuration in /var/ftp/centos8.cfg).

After the installation, the name of the server is the same as in the configurationfile. In our example, it will be LinuxPxe. You might want to change this, using the hostname command. This will only work until the next reboot. To change it permanently, change the content of the file /etc/hostname.

```
# hostname newhostname
# vi /etc/hostname
newhostname
```

Alternatively, you can change the following line in /var/ftp/centos8.cfg before starting pxe:

```
network --bootproto=dhcp --hostname=LinuxPxe
```

When you enabled the logging of the ftp protocol in VSFTP, you might want to switch this off after everything works as expected: set xferlog_std_format=YES and comment out the line with log_ftp_protocol=YES. You might also change or delete the very verbose logging switch -vvv for tftp.

You might want to remove the Legacy Network Adapter from your VM after installing CentOS8.

Scripting
The last part in any installation is (or: should be) scripting your solution. This is, because you might have to roll out the same configuration later again. As I wrote before, you can download the scripts

for creating the PXE server from github[1]. In this repository are also the configurationfiles that are the result of the changes that are described above.

Conclusion
To be honest, I was surprised that installing and configuring PXE under Linux was so difficult. When I remember my “good old Windows-days”, I could just install WDS and DHCP on the same server, configure DHCP and WDS, add a boot program, add an OS and everything was working. I was a little bit disappointed that I had to do a lot of searching and reading lots of information to be able to set PXE up on Linux. The advantage of LinuxPxe is, that downloading updates can be done during the installation of the OS. It is also possible to do some post-configuration in the Anaconda file [9].

I want to thank everybody who put his/her information on the internet - it really helped to get PXE working on CentOS8 (see f.e. the footnotes [3]-[9]). None of these website had the whole solution, I therefore wrote this article for the AMIS technology blog site.

Foot notes:
[1] Git with scripts for PXE Linux
[2] https://www.centos.org/download/
[3] https://www.rfwireless-world.com/Terminology/FTP-vs-TFTP.html
[4] You might want to use the decimal to hexadecimal converter on https://www.binaryhexconverter.com/decimal-to-hex-converter to convert decimal values to hexadecimal values.
[5] https://unix.stackexchange.com/questions/392731/how-can-i-include-all-recent-updates-during-kickstart-install-on-fedora-centos-w
[6] https://www.linuxtechi.com/configure-pxe-installation-server-centos-7/
[7] https://www.vultr.com/docs/how-to-setup-vsftpd:-ftp-server-on-centos-7
[8] https://www.thegeekdiary.com/how-to-enable-verbose-logging-for-vsftpd/
[9] https://docs.centos.org/en-US/centos/install-guide/Kickstart2/#sect-kickstart-syntax

<u>P/M: nice things to write a blog about</u>
- KVM
- Local yum repository https://linuxtechlab.com/offline-yum-repository-for-lan/ . It seems that you need a server where you have all software installed (!) that you want to update?!
- Use a spreadsheet with settings and use that in the deployment

- Routers: in Windows easy to setup, how is this done in Linux? (see also: chapter in lpic2.unix.nl, routed deamon?)
- Deployment of updates to a non-public environment: in Windows done via WSUS, how in Linux?
- Is it possible to do a “kickstart” alike deployment with Windows versions? F.e. Win 2016 and AD?
- Local time server?
- SSSD (LDAP)
- Monitoring?
- fail2ban on all nodes
- firewalld settings on all nodes: only open ports when they need to be open.
- snort on special node that is able to see all network traffic
- OpenVAS: https://www.linuxincluded.com/installing-openvas-on-centos-7/

- Mail Postfix: use DANE (lpic2.unix.nl h 12)
- Squid: enforcement of use of Squid (OBA)
- DHCP over OpenVPN (for the Ubuntu-server in the home network)
- DHCP incl. Dynamic updates of DNS
- iptables didn’t work for routing (PREROUTING and POSTROUTING were allowed to configure, but not recognized with -L, possible cause: running in a Hyper-V VM? - mac address spoofing on, private network off – no difference)

- Check for correct functioning environment: open a port and just say OK or FAILED (to prevent the use of ssh where this isn’t needed). Check both for “functional” checks (is the dhcp address in the right network?) and for security items (which ports should be open/are open?)
- Check for systems that are up and check them to a list with systems that are supposed to be up (spreadsheet?)
- SSH only when announced (something like passwd vault → combination with snort possible?)
- firewall-cmd to use services, always look to the ports that are used!

===
Docker/Kubernettes: the masterpiece
- Every week: start a container that will produce new docker images. And distribute them only if the link to the old images is to the most current version (and don’t deploy anything and just give a warning when the link isn’t to the most recent version)