Prepare the PXE environment

PXE (Preboot eXecution Environment) is a technology that allows you to boot an operating system from another computer on the local network. The process of booting the operating system in PXE technology is simplified as follows:

- the computer's BIOS calls up the network adapter's BIOS
- the NIC BIOS tries to find a DHCP server on the local network
- from the DHCP server, the client receives its IP address and the IP address of the TFTP server and the name of the file to be loaded, usually pxelinux.0 (PXE) or gpxelinux.0 (gPXE)
- the client computer downloads the bootloader file and other files that result directly from the PXE / gPXE configuration In the original PXE, all files are transferred using the TFTP protocol, which, unfortunately, is quite slow. For this reason, a third, secondary server is usually also used, usually NFS, or possibly HTTP or FTP, which is used to transfer the largest files. You can also use the extended version of PXE - gPXE, which itself supports the HTTP protocol

Client configuration

The configuration of the client computer is limited to the following steps:

- settings in the BIOS to boot from the local network (LAN) first,
- connecting the computer's network card to the router's LAN port.

PXE technology dates back to 1999, so even old computers support it. In fact, in order to boot the operating system over PXE, the client computer does not need to have any data storage devices, such as hard drives, flash drives, optical drives or floppy drives, because the operating system is loaded directly into RAM. It is worth mentioning the biggest disadvantage of PXE / gPXE, which is the memory requirement. It results from the needs of the operating system itself and the additional memory needed to support the bootloader, and additional memory is needed to load the operating system image.

PXE server installation and configuration based on Linux - Ubuntu 16.04

Initial assumptions for the example configuration:

- the server has two network cards
- the first card is enp0s3, which will be connected to the local network
    - the IP address of the card: 10.0.0.1/24
- the second network card is enp0s8, the card is directly connected to another card that provides access to the Internet (wan, lan), there is a bridge connection,

- the card has an assigned IP address: 192.168.1.171/24
- gateway settings: 192.168.1.1

The main task of the server is to provide a mechanism for automatic configuration of network cards in the local network, and additionally the ability to run and automatically install selected operating systems, using only a network connection.

Installing the tftpd server

```
apt-get install -y tftpd-hpa
```

The default tftp server directory is /var/lib/tftpboot. It's a good idea to create a local variable tftpboot_dir pointing to the home directory of the tftp server to make the next installation steps easier.

```
tftpboot_dir = "/var/lib/tftpboot"
```

The quotation marks is not necessary, but the variable is safe when path contains spaces.

dhcpd server installation

```
apt-get install -y isc-dhcp-server
```

The default dhcp server home directory is in the **/etc/dhcp** directory. We modify the default configuration of the dhcp server to the one shown below, the changes should be made in the **dhcpd.conf** file:

```
subnet 10.0.0.0 netmask 255.255.255.0 {
range 10.0.0.10 10.0.0.200;
option subnet-mask 255.255.255.0;
option broadcast-address 10.0.0.255;
option domain-name "moj_serwer.com";
option domain-name-servers 192.168.1.1;
option routers 10.0.0.1;
filename "pxelinux.0";
}

host ubuntu-16.04-pxeboot {
hardware ethernet 08:00:27:FE:46:C0;
fixed-address 10.0.0.3;
}
```

and then we enable and restart the server:

- `systemctl enable isc-dhcp-server`
- `systemctl restart isc-dhcp-server`

Configuration of netboot sever

We execute the command (must be entered in one line):

- `$ wget http://archive.ubuntu.com/ubuntu/ubuntu/dists/xenial/main/installer-amd64/current/images/netboot/netboot.tar.gz -O ubuntu-16.04-netboot.tar.gz`
- `$ mkdir ubuntu-16.04-netboot`
- `$ tar zxf ubuntu-16.04-netboot.tar.gz -C ubuntu-16.04-netboot`
- `$ sudo cp -a ubuntu-16.04-netboot/ubuntu-installer ${tftpboot_dir}`

The tftpboot_dir variable should point to the home directory of the tftp server.

Pxelinux configuration

The home directory of the tftp server contains the following files:

- pxelinux.0 - bootable image file
- ldlinux.c32 - library used by pxelinux.0
- pxelinux.cfg/default - symbolic link to syslinux.cfg
- boot-screens/syslinux.cfg - pxelinux.0 configuration file
- boot-screens/menu.cfg - boot menu
- boot-screens/vesamenu.c32 - VESA image handler
- boot-screens/libcom32.c32 - library used by vesamenu.c32
- boot-screens/libutil.c32 - library used by vesamenu.c32
- ubuntu-installer - linux system installer - Ubuntu 16.04 distribution

Executing the following commands will cause the boot-screens/syslinux.cfg and boot-screens/menu.cfg files to be created and the rest of the files to be copied from the netboot images:

- $ sudo cp ubuntu-16.04-netboot/ubuntu-installer/amd64/pxelinux.0 ${tftpboot_dir}/
- $ sudo cp ubuntu-16.04-netboot/ubuntu-installer/amd64/boot-screens/ldlinux.c32 ${tftpboot_dir}/
- $ sudo mkdir ${tftpboot_dir}/boot-screens
- $ sudo cp ubuntu-16.04-netboot/ubuntu-installer/amd64/boot-screens/libcom32.c32 ${tftpboot_dir}/boot-screens
- $ sudo cp ubuntu-16.04-netboot/ubuntu-installer/amd64/boot-screens/libutil.c32 ${tftpboot_dir}/boot-screens
- $ sudo cp ubuntu-16.04-netboot/ubuntu-installer/amd64/boot-screens/vesamenu.c32 ${tftpboot_dir}/boot-screens
- $ sudo mkdir ${tftpboot_dir}/pxelinux.cfg

- $ cd ${tftpboot_dir}/pxelinux.cfg
- $ sudo ln -s ../boot-screens/syslinux.cfg default

boot-screens/syslinux.cfg

```
path boot-screens
include boot-screens/menu.cfg
default boot-screens/vesamenu.c32
prompt 0
timeout 100
```

boot-screens/menu.cfg

Displays the following menu:

Ubuntu 16.04 automated install
Ubuntu 16.04 manual install

Items installed automatically use properly prepared "preseed" files.

Example extended version of the menu.cfg

```
menu hshift 13
menu width 49
menu margin 8
menu tabmsg
menu title Installer boot menu
label auto-debian-8
menu label ^Debian 8 automated install
kernel debian-installer/amd64/linux
append auto=true priority=critical vga=788 initrd=debian-
installer/amd64/initrd.gz
preseed/url=tftp://10.0.0.1/preseed/debian-8-preseed.cfg
label auto-ubuntu-16.04
menu label ^Ubuntu 16.04 automated install
kernel ubuntu-installer/amd64/linux
append auto=true priority=critical vga=788 initrd=ubuntu-
installer/amd64/initrd.gz
preseed/url=tftp://10.0.0.1/preseed/ubuntu-16.04 preseed.cfg
preseed/interactive=false
menu begin debian-8
menu title Debian 8
label mainmenu
menu label ^Back..
menu exit
include debian-installer/amd64/boot-screens/menu.cfg
menu end
menu begin ubuntu-16.04
menu title Ubuntu 16.04
```

```
label mainmenu
menu label ^Back..
menu exit
include ubuntu-installer/amd64/boot-screens/menu.cfg
menu end
```

Preseed configuration files

The preseed configuration files should be placed in a subdirectory of the tftp server, ${tftpboot_dir}/preseed.

Command for making empty directory:

```
sudo mkdir ${tftpboot_dir}/preseed
```

Preseed configuration for Ubuntu 16.04

The main parameters of the configuration file:

- root user password - "ubuntu",
- regular user, login - ubuntu, password - ubuntu

File content: ${tftpboot_dir}/preseed/ubuntu-16.04-preseed.cfg

```
d-i debian-installer/locale string en_US
d-i debian-installer/language string en
d-i debian-installer/country string JP
d-i keyboard-configuration/xkb-keymap select jp106
d-i passwd/user-fullname string
d-i passwd/username string ubuntu
d-i passwd/root-password password ubuntu
d-i passwd/root-password-again password ubuntu
d-i passwd/user-password password ubuntu
d-i passwd/user-password-again password ubuntu
d-i user-setup/allow-password-weak boolean true
d-i netcfg/choose_interface select auto
d-i netcfg/get_hostname string unassigned-hostname
d-i netcfg/get_domain string unassigned-domain
d-i mirror/country string manual
d-i mirror/http/hostname string http://jp.archive.ubuntu.com
d-i mirror/http/directory string /ubuntu
d-i mirror/http/proxy string
d-i clock-setup/utc boolean true
d-i clock-setup/ntp boolean true
d-i time/zone string Asia/Tokyo
d-i partman/confirm boolean true
d-i partman/choose_partition select finish
d-i partman/confirm_nooverwrite boolean true
d-i partman-auto/disk string /dev/[sv]da
```

```
d-i partman-auto/method string lvm
d-i partman-auto/choose_recipe select atomic
d-i partman-lvm/device_remove_lvm boolean true
d-i partman-lvm/confirm boolean true
d-i partman-lvm/confirm_nooverwrite boolean true
d-i partman-auto-lvm/guided_size string max
d-i partman-partitioning/confirm_write_new_label boolean true
d-i grub-installer/grub2_instead_of_grub_legacy boolean true
d-i grub-installer/only_debian boolean true
d-i grub-installer/bootdev string /dev/[sv]da
d-i pkgsel/update-policy select none
d-i finish-install/reboot_in_progress note
```

Configuration of the network gateway and the NAT mechanism

If you want to provide Internet access to computers in the local network, you must configure the gateway and the NAT address translation mechanism.

For this task, you can use the functionalities provided by the available firewall, in this case it is the iptables program.

Example gateway and NAT configuration:

```
route add -net 10.0.0.0/24 dev enp0s3
sysctl -w net.ipv4.ip_forward=1
iptables -F
iptables -t nat -F
iptables -t nat -A POSTROUTING ! -d 10.0.0.0/24 -o enp0s8 -j SNAT --
to-source 192.168.1.171
```

Attention !!! - Availability of package repositories for a given distribution is usually limited in time. In case of difficulties with access to installation packages, it is recommended to update the distribution to a newer version of the system.