PXELINUX

From Syslinux Wiki

Contents

- 1 Description
- 2 Working directory
- 3 Configuration
- 4 Options
  - 4.1 Option priority
  - 4.2 DHCP options
  - 4.3 Hardcoded options
- 5 HTTP and FTP
- 6 Filename syntax
- 7 Keeppxe
- 8 Examples
  - 8.1 Configuration filename
  - 8.2 TFTP servers
  - 8.3 DHCP config - Simple
  - 8.4 DHCP Config - PXE-1
  - 8.5 DHCP Config - Encapsulated
  - 8.6 DHCP Config - ISC dhcpd options
    - 8.6.1 vendor options
    - 8.6.2 vendor options - handcrafted
- 9 Known issues
  - 9.1 Broken PXE stacks
- 10 PXE stack on a floppy
- 11 Deploy Linux from Windows WDS/RIS server using PXELinux
- 12 Custom Menu Example with sub-menus
- 13 Notes
  - 13.1 Error recovery
  - 13.2 MTFTP
- 14 UEFI
- 15 Resources

Description

PXELINUX is a Syslinux derivative, for booting from a network server using a network ROM conforming to the Intel PXE (Pre-Execution Environment) specification. PXELINUX is not a program intended to be flashed or burned into a PROM on the network card. For such possibility, check out iPXE (http://ipxe.org/).

If you want to create PXE-compliant boot PROM for your network card (to use with PXELINUX, for example), check out NetBoot (http://netboot.sourceforge.net/).

Working directory

The initial Current Working Directory is either as supplied by DHCP option 210 (pxelinux.pathprefix), the hardcoded path-prefix or the parent directory of the PXELINUX file, as indicated by DHCP fields *sname* and *file* (sname="192.168.2.3" and file="boot/pxelinux.0" result in "tftp://192.168.2.3/boot/", or in "192.168.2.3::boot/" in older PXELINUX format) with the precedence as specified under the #Options section of this document.

All unqualified filenames are relative to the Current Working Directory.

Configuration

The basic configuration is the same for all Syslinux variants. This document explains only some of the differences specifically applicable to PXELINUX.

On the TFTP server, create the directory "`/tftpboot`", and copy "`pxelinux.0`" (from the Syslinux distribution) and any kernel or initrd images that you want to boot.

`[5.00+]` Also copy "`ldlinux.c32`" from the Syslinux distribution to the "`/tftpboot`" directory on the TFTP server.

Finally, create the directory "`/tftpboot/pxelinux.cfg`". The configuration file (equivalent of syslinux.cfg -- see the SYSLINUX FAQ for the options here) will live in this directory.

Because more than one system may be booted from the same server, the configuration file name depends on the IP address of the booting machine.

Before a generic explanation, let's see first an example. When:

- the bootloader file name is "`/mybootdir/pxelinux.0`"; and,
- the client UUID is "`b8945908-d6a6-41a9-611d-74a6ab80b83d`"; and,
- the `Ethernet` MAC address is "`88:99:AA:BB:CC:DD`"; and,
- the IP address is "`192.168.2.91`", or in uppercase hexadecimal, "`C0A8025B`";

then PXELINUX will try the following configuration files (in this order):

```
        /mybootdir/pxelinux.cfg/b8945908-d6a6-41a9-611d-74a6ab80b83d
        /mybootdir/pxelinux.cfg/01-88-99-aa-bb-cc-dd
        /mybootdir/pxelinux.cfg/C0A8025B
        /mybootdir/pxelinux.cfg/C0A8025
        /mybootdir/pxelinux.cfg/C0A802
        /mybootdir/pxelinux.cfg/C0A80
        /mybootdir/pxelinux.cfg/C0A8
        /mybootdir/pxelinux.cfg/C0A
        /mybootdir/pxelinux.cfg/C0
        /mybootdir/pxelinux.cfg/C
        /mybootdir/pxelinux.cfg/default
```

Let's see now what exactly the above example represents.

After attempting the file as specified in the DHCP or hardcoded options, PXELINUX will probe the following paths, prefixed with "`pxelinux.cfg/`", under the initial Working Directory.

- The client UUID, if provided by the PXE stack.

  Note that some BIOSes do not have a valid UUID, and it might end up reporting something like all 1's. This value is represented in the standard UUID format using lowercase hexadecimal digits, e.g. "`b8945908-d6a6-41a9-611d-74a6ab80b83d`".

- The hardware type (using its ARP type code) and address, all in lowercase hexadecimal with dash separators.

  For example, for an Ethernet (ARP type "`1`") with address "`88:99:AA:BB:CC:DD`", it would search for the filename "`01-88-99-aa-bb-cc-dd`".

- The client's own IPv4 address in uppercase hexadecimal, followed by removing hex characters, one at a time, from the end. For example, "`192.168.2.91`" → "`C0A8025B`".

  The included program, "`gethostip`", can be used to compute the hexadecimal IP address for any host.

- Lowercase "`default`".

Note that all references to filenames are relative to the directory in which "`pxelinux.0`" lives.

PXELINUX generally requires for filenames (including any relative path) to be 127 characters or shorter in length.

`[3.20+]` If PXELINUX cannot find a configuration file, it will reboot after the timeout interval has expired. This keeps a machine from getting stuck indefinitely due to a boot server failure.

Options

`[1.62+]` Depending on the specific DHCP server, the following nonstandard DHCP options might be available so to customize the specific behaviour of PXELINUX. See RFC 5071 for some additional information about these options. Options for PXELINUX can be specified by DHCP options, or hardcoded into the binary.

Option priority

Hardcoded "after-options" are applied after DHCP options (and overrride them) while hardcoded "before-options" are applied prior to DHCP options. The default behavior takes the lowest priority.

DHCP options

Option 208 ***pxelinux.magic***

Earlier versions of PXELINUX required this option to be set to `F1:00:74:7E` (241.0.116.126) for PXELINUX to be able to recognize any special DHCP options whatsoever. As of PXELINUX 3.55, this option is deprecated and is no longer required.

Option 209 ***pxelinux.configfile***

Specify the initial PXELINUX configuration file name, which may be qualified or unqualified.

Option 210 ***pxelinux.pathprefix***

Specify the PXELINUX common path prefix, instead of deriving it from the boot file name. This almost certainly needs to end in whatever character the TFTP server OS uses as a pathname separator, e.g. slash (/) for Unix.

Option 211 *pxelinux.reboottime*

Specify, in seconds, the time to wait before reboot in the event of TFTP failure. "`0`" (zero) means wait "forever" (in reality, it waits approximately 136 years).

Hardcoded options

`[3.83+]` The program "`pxelinux-options`" can be used to hard-code DHCP options into the "`pxelinux.0`" image file. This is sometimes useful when the DHCP server is under different administrative control.

Hardcoded options:

```
  6 => 'domain-name-servers',
 15 => 'domain-name',
 54 => 'next-server',
209 => 'config-file',
210 => 'path-prefix',
211 => 'reboottime'
```

HTTP and FTP

Older versions of PXELINUX supported HTTP by using a hybrid bootloader that also contained gPXE/iPXE, with such images named either gpxelinux.0 or ipxelinux.0.

Since version 5.10, a special PXELINUX binary, lpxelinux.0, natively supports HTTP and FTP transfers, greatly increasing load speed and allowing for standard HTTP scripts to present PXELINUX's configuration file. To use HTTP or FTP, use standard URL syntax as filename; use DHCP options to transmit a suitable URL prefix to the client, or use the "pxelinux-options" tool provided in the "utils" directory to program it directly into the lpxelinux.0 file.

While using HTTP/FTP (syntax), trying to use pxelinux.0 (i.e. without the letter "`l`" prefix) without iPXE/gPXE running underneath, will result in a "*file not found*" warning without any explanation as to the cause!

Example:

```
LABEL linux-http
LINUX http://boot-server/boot/mykernel
APPEND initrd=http://boot-server/boot/myinitrd
```

Filename syntax

PXELINUX supports the following special pathname conventions:

::filename

Suppress the common filename prefix, i.e. passes the string "filename" unmodified to the server.

IP address::filename

(e.g. 192.168.2.3::filename)
Suppress the common filename prefix, and send a request to an alternate TFTP server.
Instead of an IP address, a DNS name can be used.

It will be assumed to be fully qualified if it contains dots; otherwise the local domain as reported by the DHCP server (option 15) will be added.

The double colon symbol ("::") was chosen because it is unlikely to conflict with operating system usage. However, if you happen to have an environment for which the special treatment of "`::`" is a problem, please contact the Syslinux mailing list.

`[4.00+]`, PXELINUX also supports standard URL syntax.

Keeppxe

Normally, PXELINUX will unload the PXE and UNDI stacks before invoking the kernel. In special circumstances (for example, when using MEMDISK to boot an operating system with an UNDI network driver) it might be desirable to keep the PXE stack in memory. If the option "keeppxe" is given on the kernel command line, then PXELINUX will keep the PXE and UNDI stacks in memory. (If you don't know what this means, you probably don't need it.)

Examples

Configuration filename

For DHCP siaddr "192.168.2.3", file "mybootdir/pxelinux.0", client UUID "b8945908-d6a6-41a9-611d-74a6ab80b83d", `Ethernet` MAC address "88:99:AA:BB:CC:DD" and IPv4 address "192.168.2.91", the following files will be attempted in this order (after "config-file" options):

```
        mybootdir/pxelinux.cfg/b8945908-d6a6-41a9-611d-74a6ab80b83d
        mybootdir/pxelinux.cfg/01-88-99-aa-bb-cc-dd
        mybootdir/pxelinux.cfg/C0A8025B
        mybootdir/pxelinux.cfg/C0A8025
        mybootdir/pxelinux.cfg/C0A802
        mybootdir/pxelinux.cfg/C0A80
        mybootdir/pxelinux.cfg/C0A8
        mybootdir/pxelinux.cfg/C0A
        mybootdir/pxelinux.cfg/C0
        mybootdir/pxelinux.cfg/C
        mybootdir/pxelinux.cfg/default
```

TFTP servers

For best results, use a TFTP server with support for the "`tsize`" TFTP option (RFC 1784/RFC 2349).

Please check out the Hardware Compatibility reference page to see if your PXE stacks need any special workarounds.

Some TFTP servers that have been successfully used with PXELINUX include:

- The "tftp-hpa" TFTP server (highly portable and port of the BSD TFTP server code) supports options and is available at:

  http://www.kernel.org/pub/software/network/tftp/ or ftp://ftp.kernel.org/pub/software/network/tftp/

  and on any kernel.org mirror (http://www.kernel.org/mirrors/).

- Another TFTP server supporting options is "atftp" by Jean-Pierre Lefebvre:

ftp://ftp.mamalinux.com/pub/atftp/

atftp is likely going to perform better than "tftp-hpa" on a large boot server, but may not be quite as widely portable.

- If your boot server is running Windows (and you can't change that), try "tftpd32" by Philippe Jounin (you need version 2.11 or later; previous versions had a bug which made it incompatible with PXELINUX):

  http://tftpd32.jounin.net/

- Eric Cook of Intel also reports that the TFTPD server from Win2000 Server RIS can be used:

  The trick is to install RIS, but don't configure it with the GUI. Instead, do the following:
  In the registry, add the folder `\HKLM\System\CurrentControlSet\Services\TFTPD\Parameters`.
  In the Parameters folder, add a key called `Directory`, with a value of the TFTP root directory path.
  With the Services GUI, configure the TFTPD service for Automatic start and start it. If you DO configure the RIS in Win2k, you end up with the MS PXE stuff, which is ugly to get rid of.

  However, Christian "Dr. Disk" Hechelmann reports having success with using the Windows RIS as-is, and has sent a nice writeup on how to set it up. See Windows Remote Install System.

DHCP config - Simple

The PXE protocol uses a very complex set of extensions to DHCP or BOOTP. However, most PXE implementations -- this includes all Intel ones version 0.99n and later -- seem to be able to boot in a "conventional" DHCP/TFTP configuration. Assuming you don't have to support any very old or otherwise severely broken clients, this is probably the best configuration unless you already have a PXE boot server on your network.

A sample DHCP setup, using the "conventional TFTP" configuration, would look something like the following, using ISC dhcp (2.0 or later) "dhcpd.conf" syntax:

```
        allow booting;
        allow bootp;

        # Standard configuration directives...

        option domain-name "domain_name";
        option subnet-mask subnet_mask;
        option broadcast-address broadcast_address;
        option domain-name-servers dns_servers;
        option routers default_router;

        # Group the PXE bootable hosts together
        group {
                # PXE-specific configuration directives...
                next-server TFTP_server_address;
                filename "/tftpboot/pxelinux.0";

                # You need an entry like this for every host
                # unless you're using dynamic addresses
                host hostname {
                        hardware ethernet ethernet_address;
                        fixed-address hostname;
                }
        }
```

Note that if your particular TFTP daemon runs under chroot (tftp-hpa will do this if you specify the "`-s`" (secure) option; this is highly recommended), you almost certainly should not include the /tftpboot prefix in the filename statement.

DHCP Config - PXE-1

If the simple config does not work for your environment, you probably should set up a "PXE boot server" on port 4011 of your TFTP server; a free PXE boot server is available at: http://www.kano.org.uk/projects/pxe/

With such a boot server defined, your DHCP configuration should look the same except for an "`option dhcp-class-identifier`" (ISC dhcp 2) or "`option vendor-class-identifier`" (ISC dhcp 3):

```
        allow booting;
        allow bootp;

        # Standard configuration directives...

        option domain-name "domain_name";
        option subnet-mask subnet_mask;
        option broadcast-address broadcast_address;
        option domain-name-servers dns_servers;
        option routers default_router;

        # Group the PXE bootable hosts together
        group {
                # PXE-specific configuration directives...
                option dhcp-class-identifier "PXEClient";
                next-server pxe_boot_server_address;

                # You need an entry like this for every host
                # unless you're using dynamic addresses
                host hostname {
                        hardware ethernet ethernet_address;
                        fixed-address hostname;
                }
        }
```

Here, the boot file name is obtained from the PXE server.

DHCP Config - Encapsulated

If the "conventional TFTP" configuration doesn't work on your clients, and setting up a PXE boot server is not an option, you can attempt the following configuration. It has been known to boot some configurations correctly; however, there are no guarantees:

```
allow booting;
allow bootp;

# Standard configuration directives...

option domain-name "domain_name";
option subnet-mask subnet_mask;
option broadcast-address broadcast_address;
option domain-name-servers dns_servers;
option routers default_router;

# Group the PXE bootable hosts together
group {
  # PXE-specific configuration directives...
  option dhcp-class-identifier "PXEClient";
  option vendor-encapsulated-options 09:0f:80:00:0c:4e:65:74:77:6f:72:6b:20:62:6f:6f:74:0a:07:00:50:72:6f:6d:70:74
  next-server TFTP_server;
  filename "/tftpboot/pxelinux.0";
  # You need an entry like this for every host
  # unless you're using dynamic addresses
  host hostname {
    hardware ethernet ethernet_address;
    fixed-address hostname;
  }
}
```

Note that this will not boot some clients that will boot with the "conventional TFTP" configuration; Intel Boot Client 3.0 and later are known to fall into this category.

DHCP Config - ISC dhcpd options

ISC dhcp 3.0 supports a rather nice syntax for specifying custom options. The following syntax can be used in "dhcpd.conf" if you are running this version of dhcpd:

```
       option space pxelinux;
       option pxelinux.magic      code 208 = string;
       option pxelinux.configfile code 209 = text;
       option pxelinux.pathprefix code 210 = text;
       option pxelinux.reboottime code 211 = unsigned integer 32;
```

Note: In earlier versions of PXELINUX, this would only work as a "`site-option-space`". Since PXELINUX 2.07, this will work both as a "`site-option-space`" (unencapsulated) and as a "`vendor-option-space`" (type 43 encapsulated.) This may avoid messing with the "`dhcp-parameter-request-list`", as detailed below.

`[PXELINUX 2.07+]` This is supported both as a "`site-option-space`", and as a "`vendor-option-space`".

Inside your PXELINUX-booting group or class (wherever you have the PXELINUX-related options, such as the "filename" option), you would add, for example:

```
# Always include the following lines for all PXELINUX clients
site-option-space "pxelinux";
option pxelinux.magic f1:00:74:7e;
if exists dhcp-parameter-request-list {
  # Always send the PXELINUX options (specified in hexadecimal)
  option dhcp-parameter-request-list = concat(option dhcp-parameter-request-list,d0,d1,d2,d3);
}
# These lines should be customized to your setup
option pxelinux.configfile "configs/common";
option pxelinux.pathprefix "/tftpboot/pxelinux/files/";
option pxelinux.reboottime 30;
filename "/tftpboot/pxelinux/pxelinux.bin";
```

Note that the configfile is relative to the pathprefix: this will look for a config file called `/tftpboot/pxelinux/files/configs/common` on the TFTP server.

The "`option dhcp-parameter-request-list`" statement forces the DHCP server to send the PXELINUX-specific options, even though they are not explicitly requested. Since the DHCP request is done before PXELINUX is loaded, the PXE client won't know to request them.

In ISC dhcp versions greater than 3.0, site-local option spaces start at 224, not 128 (to be compliant with RFC 3942), so you should define the PXELINUX options 208-211 as regular DHCP options, rather than site local ones. For example:

```
       option magic      code 208 = string;
       option configfile code 209 = text;
       option pathprefix code 210 = text;
       option reboottime code 211 = unsigned integer 32;
```

Inside your PXELINUX-booting group or class (wherever you have the PXELINUX-related options, such as the "filename" option), you would add, for example:

```
# Always include the following lines for all PXELINUX clients
option magic f1:00:74:7e;
if exists dhcp-parameter-request-list {
  # Always send the PXELINUX options (specified in hexadecimal, ie: 208 = 0xd0, 209 = 0xd1, etc.)
  option dhcp-parameter-request-list = concat(option dhcp-parameter-request-list,d0,d1,d2,d3);
}
# These lines should be customized to your setup
option configfile "configs/common";
option pathprefix "/tftpboot/pxelinux/files/";
option reboottime 30;
filename "/tftpboot/pxelinux/pxelinux.bin";
```

Note that the configfile is relative to the pathprefix: this will look for a config file called `/tftpboot/pxelinux/files/configs/common` on the TFTP server.

The "`option dhcp-parameter-request-list`" statement forces the DHCP server to send the PXELINUX-specific options, even though they are not explicitly requested. Since the DHCP request is done before PXELINUX is loaded, the PXE client won't know to request them.

Using ISC dhcp 3.0 you can create a lot of these strings on the fly. For example, to use the hexadecimal form of the hardware address as the configuration file name, you could do something like:

```
site-option-space "pxelinux";
option pxelinux.magic f1:00:74:7e;
if exists dhcp-parameter-request-list {
  # Always send the PXELINUX options (specified in hexadecimal)
  option dhcp-parameter-request-list = concat(option dhcp-parameter-request-list,d0,d1,d2,d3);
}
option pxelinux.configfile =
  concat("pxelinux.cfg/", binary-to-ascii(16, 8, ":", hardware));
filename "/tftpboot/pxelinux.bin";
```

If you used this from a client whose Ethernet address was `58:FA:84:CF:55:0E`, this would look for a configuration file named "`/tftpboot/pxelinux.cfg/1:58:fa:84:cf:55:e`".

vendor options

```
host trantor-sky2 {
  hardware ethernet 00:00:5a:70:c2:71;
  vendor-option-space pxelinux;
  option pxelinux.magic f1:00:74:7e;
  option pxelinux.pathprefix "http://raidtest.hos.anvin.org/tftpboot/";
  option pxelinux.reboottime 30;
  filename "/pxelinux.0";
}
```

That removes the need to muck with the dhcp-parameter-request-list.

vendor options - handcrafted

```
host trantor-sky2 {
  hardware ethernet 00:00:5a:70:c2:71;
  option vendor-encapsulated-options
    d0:04:f1:00:74:73:
    d2:23:68:74:74:70:3a:2f:2f:72:61:69:64:74:65:73:74:2e:
      61:6e:76:69:6e:2e:6f:72:67:2f:74:66:74:70:62:6f:
      6f:74:2f:
    d3:04:00:00:00:1e;
  filename "/pxelinux.0";
}
```

Known issues

- ~~[-3.63] Requires a TFTP server with support for the "tsize" option.~~
- The error recovery routine doesn't work quite right. For now, it just does a hard reset - seems good enough.
- We should probably call the UDP receive function in the keyboard entry loop, so that we answer ARP requests.
- Boot sectors/disk images are not supported yet... They probably need auxilliary information (such as device) to work at all.
- pxechain.com, as of PXELINUX 4.00, was broken. See Common Problems. See also pxechn.c32.

If you have additional problems, please contact the Syslinux mailing list. Before you post something, please make sure you have checked that your kernel files are not named using extensions that have special meanings:

```
.0    PXE bootstrap program (NBP) [PXELINUX only]
.bin  "CD boot sector" [ISOLINUX only]
.bs   Boot sector [SYSLINUX only]
.bss  Boot sector, DOS superblock will be patched in [SYSLINUX only]
.c32  COM32 image (32-bit COMBOOT)
.cbt  COMBOOT image (not runnable from DOS)
.com  COMBOOT image (runnable from DOS)
.img  Disk image [ISOLINUX only]
```

Broken PXE stacks

Lots of PXE stacks, especially old ones, have various problems of varying degrees of severity. Please check out the Hardware Compatibility reference page for possible workarounds.

PXE stack on a floppy

If your network card doesn't have a PXE boot ROM, there are a couple of PXE stacks available.

Etherboot is a ROM kit that allows you to create your own PXE boot ROM (version 5.3.7 or later required), as well as make one that can be run from a boot floppy. The Etherboot home page is at: http://www.etherboot.org/ ... and you can use ROM-o-matic to automatically create customized boot ROMs for your needs. See: http://rom-o-matic.net/

NetBoot is a ROM kit that may allow you to create your own PXE boot ROM, and possibly also run one from a floppy. It is available at: http://netboot.sourceforge.net/

A multi-hardware boot floppy is included with Windows Server 2000 and 2003. A company called Argon Technology used to offer a free-as-in-beer updated version, but it seems to have gone fully commercial. This floppy (which can also be burned to a CD using El Torito in floppy-emulation mode), is known to work with PXELINUX 2.03 or later only.

Deploy Linux from Windows WDS/RIS server using PXELinux

Note 1: For this example we will use the Simple Menu System only, but it is easy to modify the following procedure so to use the vesamenu system or no menu.

Note 2: For WDS, it is best to run it in Mixed Mode (makes life easier). Alternatively, see WDSLINUX for setting it up with WDS only.

On RIS Server, create the following folder structure:

```
      Setup\English\Images\PXELinux\i386\templates\pxelinux.cfg\
      Setup\English\Images\PXELinux\i386\templates\conf
      Setup\English\Images\PXELinux\i386\templates\knl
      Setup\English\Images\PXELinux\i386\templates\img
```

NOTE: `Setup\English\Images` is the location of the other RIS images. You can also change the name PXELinux to anything you want if for example you wish to have a seperate option in RIS for each distro you deploy.

Download the latest version of Syslinux from: http://www.kernel.org/pub/linux/utils/boot/syslinux/

From Redhat AS4u3 CD1 (or cd of the distro you wish to deploy), in the directory "`images\pxeboot`" copy the following files into "`Setup\English\Images\PXELinux\i386\templates`" on the RIS server.

```
      vmlinuz
      initrd.img
```

Rename these files to:

```
      vmlinuz-<distro>-<arch>
      initrd-<distro>-<arch>.img
```

e.g.:

```
      vmlinuz-rhas43-x86
      initrd-rhas43-x86.img
```

Place the renamed "vmlinuz" file in the folder "\knl". Place the renamed "initrd.img" file in the folder "\img".

NOTE: You must use the files vmlinuz and initrd.img from the distro version you intend to deploy (although sometimes you can get away with using older or newer ones for older / newer versions).

From the Syslinux distribution file downloaded, extract the file "`pxelinux.0`" (and, since version 5.00, also extract the corresponding "`ldlinux.c32`" file) to "`Setup\English\Images\PXELinux\i386\templates`" on the RIS server.

In "`Setup\English\Images\PXELinux\i386\templates`" create a file "pxelinux.sif" and give it the following Contents:

```
[OSChooser]
Description = "Linux"
Help = "This option runs a Linux installer."
LaunchFile = "Setup\English\Images\PXELinux\i386\templates\pxelinux.0"
ImageType = Flat
Version="1.01"
```

In "`Setup\English\Images\PXELinux\i386\templates\pxelinux.cfg\`" create a file called "default" and give it the following contents:

```
 # Default boot option to use
 DEFAULT menu.c32
 # Prompt user for selection
 PROMPT 0
 # Menu Configuration
 MENU WIDTH 80
 MENU MARGIN 10
 MENU PASSWORDMARGIN 3
 MENU ROWS 12
 MENU TABMSGROW 18
 MENU CMDLINEROW 18
 MENU ENDROW 24
 MENU PASSWORDROW 11
 MENU TIMEOUTROW 20
 MENU TITLE Main Menu
 # Menus
 # x86
 LABEL x86
   MENU LABEL 32Bit (x86)
   KERNEL menu.c32
   APPEND conf/x86.conf
 # x64
 LABEL x64
   MENU LABEL 64Bit (x64)
   KERNEL menu.c32
   APPEND conf/x64.conf
```

In "`Setup\English\Images\PXELinux\i386\templates\conf\`" create a file called "x86.conf" (this will list all of our 32bit OS installs) and give it the following contents:

```
 # Default boot option to use
 DEFAULT menu.c32
 # Prompt user for selection
 PROMPT 0
 # Menu Configuration
 MENU WIDTH 80
 MENU MARGIN 10
 MENU PASSWORDMARGIN 3
 MENU ROWS 12
 MENU TABMSGROW 18
 MENU CMDLINEROW 18
 MENU ENDROW 24
 MENU PASSWORDROW 11
 MENU TIMEOUTROW 20
 MENU TITLE 32Bit (x86) OS Choice
 # Return to Main Menu
 LABEL MainMenu
```

```
  MENU DEFAULT
  MENU LABEL ^Main Menu
  KERNEL menu.c32
#
# Blank boots
#
LABEL linux-43
  MENU LABEL ^Blank Boot 4.3
  KERNEL knl/vmlinuz-rhas43-x86
  APPEND initrd=initrd=img/initrd-rhes43-x86.img
```

In "`Setup\English\Images\PXELinux\i386\templates\conf\`" create a file called "x64.conf" (this will list all of our 64bit OS installs) and give it the following contents:

```
# Default boot option to use
DEFAULT menu.c32
# Prompt user for selection
PROMPT 0
# Menu Configuration
MENU WIDTH 80
MENU MARGIN 10
MENU PASSWORDMARGIN 3
MENU ROWS 12
MENU TABMSGROW 18
MENU CMDLINEROW 18
MENU ENDROW 24
MENU PASSWORDROW 11
MENU TIMEOUTROW 20
MENU TITLE 64Bit (x64) OS Choice
# Return to Main Menu
LABEL MainMenu
  MENU DEFAULT
  MENU LABEL ^Main Menu
  KERNEL menu.c32
#
# Blank boots
#
LABEL linux-43
  MENU LABEL ^Blank Boot 4.3
  KERNEL knl/vmlinuz-rhas43-x64
  APPEND initrd=img/initrd-rhes43-x64.img
```

Now if you Boot to your RIS server, on the OS list screen you should see one called Linux. Choosing this will boot PXELinux and take you to the main menu to choose your arch type and then the distro you would like to install.

Using the new Syslinux features for vesamenu can make for a very easy to use and pleasant interface.

Custom Menu Example with sub-menus

Many advanced options here. Read full documentation on Syslinux to understand it all.

Its password protected from modification during PXE boot, very useful to prevent tampering.

Note: this example uses the legacy way to generate submenus, which is compatible with older Syslinux versions. Syslinux 3.62 supports a slightly different syntax, which is faster and somewhat more flexible.

Directory Structure:

```
  /tftpboot/
  /tftpboot/memdisk
  /tftpboot/pxelinux.0
  /tftpboot/menu.c32
```

```
    /tftpboot/pxelinux.cfg/
    /tftpboot/pxelinux.cfg/default
    /tftpboot/pxelinux.cfg/graphics.conf
    /tftpboot/pxelinux.cfg/fixes.menu
    /tftpboot/pxelinux.cfg/setup.menu

    /tftpboot/TRK/
    /tftpboot/TRK/chkdsk.trk
    /tftpboot/TRK/initrd.trk
    /tftpboot/TRK/kernel.trk

    /tftpboot/Memtest/memtest.x86

    /tftpboot/Suse/
    /tftpboot/Suse/initrd92
    /tftpboot/Suse/linux92

    /tftpboot/Floppy/
    /tftpboot/Floppy/kbfloppy.img
```

`/tftpboot/pxelinux.cfg/default`:

```
DEFAULT menu.c32
PROMPT 0

MENU TITLE PXE Special Boot Menu
MENU INCLUDE pxelinux.cfg/graphics.conf
MENU AUTOBOOT Starting Local System in # seconds

LABEL bootlocal
  MENU LABEL ^Boot Point of Sale
  MENU DEFAULT
  LOCALBOOT 0
TIMEOUT 80
TOTALTIMEOUT 9000

LABEL FixesMenu
  MENU LABEL ^Fixes Menu
  KERNEL menu.c32
  APPEND pxelinux.cfg/graphics.conf pxelinux.cfg/fixes.menu

LABEL SetupMenu
  MENU LABEL ^Setup Menu
  KERNEL menu.c32
  APPEND pxelinux.cfg/graphics.conf pxelinux.cfg/setup.menu
```

`/tftpboot/pxelinux.cfg/graphics.conf`:

```
MENU COLOR TABMSG    37;40     #80ffffff #00000000
MENU COLOR HOTSEL    30;47     #40000000 #20ffffff
MENU COLOR SEL       30;47     #40000000 #20ffffff
MENU COLOR SCROLLBAR 30;47     #40000000 #20ffffff
MENU MASTER PASSWD yourpassword
MENU WIDTH 80
MENU MARGIN 22
MENU PASSWORDMARGIN 26
MENU ROWS 6
MENU TABMSGROW 15
MENU CMDLINEROW 15
MENU ENDROW 24
MENU PASSWORDROW 12
MENU TIMEOUTROW 13
MENU VSHIFT 6
MENU PASSPROMPT Enter Password:
NOESCAPE 1
ALLOWOPTIONS 0
```

Change `ALLOWOPTIONS` to `1` (one) so to be able to edit any of the entries while booted with PXE on the menu system for testing purposes. Also change `NOESCAPE` to `0` (zero) for the same reasons.

`/tftpboot/pxelinux.cfg/fixes.menu`:

```
MENU TITLE Fixes Menu

LABEL MainMenu
  MENU LABEL ^Return to Main Menu
  KERNEL menu.c32
  APPEND pxelinux.cfg/default

LABEL fsck
  MENU LABEL ^File system check
  KERNEL TRK/kernel.trk
  APPEND initrd=TRK/chkdsk.trk ramdisk_size=32768 root=/dev/ram0 vga=0

LABEL memtest
  MENU LABEL ^Memory Test: Memtest86+ v1.65
  KERNEL Memtest/memtest.x86

LABEL trk3
  MENU LABEL ^Trinity Rescue Kit
  KERNEL TRK/kernel.trk
  APPEND initrd=TRK/initrd.trk ramdisk_size=32768 root=/dev/ram0 vga=0 trknfs=IPADDR:/trk ip=:::::dhcp splash=ver
```

`/tftpboot/pxelinux.cfg/setup.menu`:

```
MENU TITLE Setup Menu

LABEL MainMenu
  MENU LABEL ^Return to Main Menu
  KERNEL menu.c32
  APPEND pxelinux.cfg/default

LABEL setupkb
  MENU LABEL ^Any floppy disk image
  KERNEL memdisk
  APPEND initrd=Floppy/kbfloppy.img

LABEL linux
  MENU PASSWD yourpassword
  MENU LABEL Install - ^Classic
  KERNEL Suse/linux92
  APPEND initrd=Suse/initrd92 ramdisk_size=65536 vga=0 textmode=1 install=http://IPADDR serverdir=/9.2/install aut

LABEL trkclone
  MENU PASSWD yourpassword
  MENU LABEL Install - ^Faster
  KERNEL TRK/kernel.trk
  APPEND initrd=TRK/initrd.trk ramdisk_size=65536 root=/dev/ram0 vga=0 install=Y trknfs=IPADDR:/trk ip=:::::dhcp

LABEL linuxfull
  MENU PASSWD yourpassword
  MENU LABEL Install - ^Developer
  KERNEL Suse/linux92
  APPEND initrd=Suse/initrd92 ramdisk_size=65536 vga=0 textmode=1 install=http://IPADDR serverdir=/9.2/install aut
```

Notes

Error recovery

If the boot fails, PXELINUX (unlike SYSLINUX) will not wait forever; rather, if it has not received any input for approximately five minutes after displaying an error message, it will reset the machine. This allows an unattended machine to recover in case it had bad enough luck of trying to boot at the same time the TFTP server goes down.

Please check out the Hardware Compatibility reference page to see if your PXE stacks need any special workarounds.

MTFTP

PXELINUX does not support MTFTP, and there are no plans of doing so. It is of course possible to use MTFTP for the initial boot, if you have such a setup. MTFTP server setup is beyond the scope of this document.

UEFI

The "`(l)pxelinux.0`" bootloaders are capable of netbooting BIOS-based clients. Hardware using UEFI has to use the adequate "`syslinux.efi`" (for EFI IA32 or for EFI X64, respectively) instead of using "`(l)pxelinux.0`".

For example, in the dhcp configuration file, something similar to the following conditions could be used:

```
; This one line must be outside any bracketed scope
option architecture-type code 93 = unsigned integer 16;

class "pxeclients" {
     match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";

     if option architecture-type = 0 {
         filename "path/to/BIOS/pxelinux.0";
     } elsif option architecture-type = 9 {
         filename "path/to/EFIx64/syslinux.efi";
     } elsif option architecture-type = 7 {
         filename "path/to/EFIx64/syslinux.efi";
     } elsif option architecture-type = 6 {
         filename "path/to/EFIia32/syslinux.efi";
     }
}
```

About "architecture-type":

- 06 (EFI IA32) is sometimes (mis)used for legacy (CSM) boot of x64 machines by some vendors.
- 07 (EFI BC) is sometimes (mis)used for EFI x64 boot by some vendors.

- Each bootloader needs its respective "`ldlinux.*`" module too:

```
path/to/BIOS/ldlinux.c32
path/to/EFIia32/ldlinux.e32
path/to/EFIx64/ldlinux.e64
```

Alternatively, the PXE directory can be organized by architecture number. This allows the namespace to be managed on the TFTP server (usually with symlinks), rather than requiring changes to DHCP:

```
class "pxeclients" {
    match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
    filename concat("path/to/PXE-",  binary-to-ascii(16, 16, "", option architecture-type), "/pxelinux.0");
}
```

- The respective library modules for each firmware (if they are needed), cannot share the same directory between each other, since they have the same file name. Use individual configuration files for each architecture/firmware, and if necessary also add a relevant PATH directive in each of them.

- The parent directory for the network bootloaders could be the same for all of them, if each bootloader is named differently. In such case, relevant PATH directives might be needed.

- Optionally, use additional directives, such as INCLUDE and/or CONFIG, so to share in-common Syslinux configuration files.

- See also PXELINUX-Multi-Arch.

Notes:

- Instead of "`pxelinux.0`", the alternative "`lpxelinux.0`" (with initial lowercase letter "L") can be used for BIOS clients.

- The "`syslinux.efi`" file for EFI IA32 is different from the one for EFI X64; each architecture/firmware has its own "`syslinux.efi`".

- The "`syslinux.efi`" file for EFI X64 is the same binary for disk booting EFI X64 and for network booting EFI X64.

- The "`syslinux.efi`" file for EFI IA32 is the same binary for disk booting EFI IA32 and for network booting EFI IA32.

- Each "`syslinux.efi`" file can be renamed (e.g. to "`bootx64.efi`"); just beware to use the adequate path(s) and name(s) in the dhcp configuration file.

Resources

- RFC 2132 - March 1997 - DHCP Options and BOOTP Vendor Extensions
- RFC 4578 - November 2006 - DHCP Options for PXE
- RFC 5071 - December 2007 - DHCP Options used by PXELINUX
- RFC 5494 - April 2009 - IANA Guidelines for ARP
- RFC 5970 - September 2010 - DHCPv6 Options for Network Boot

Retrieved from "http://www.syslinux.org/wiki/index.php?title=PXELINUX&oldid=4494"

Categories: PXELINUX | TFTP | Menu

- This page was last modified on 16 July 2016, at 06:01.
- This page has been accessed 1,071,203 times.