PXELINUX

From Syslinux Wiki

Contents

What is PXELINUX?

PXELINUX is a SYSLINUX derivative, for booting Linux 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. If you want that, 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/).

How do I Configure PXELINUX?

PXELINUX operates in many ways like SYSLINUX. If you are not familiar with SYSLINUX, read the SYSLINUX FAQ first, since this documentation only explains the differences.

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.

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. PXELINUX will search for its config file on the boot server in the following way:

- First, it will search for the config file using the hardware type (using its ARP type code) and address, all in lower case 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.

- Next, it will search for the config file using its own IP address in upper case hexadecimal, e.g. 192.0.2.91 -> C000025B (you can use the included progam gethostip to compute the hexadecimal IP address for any host). If that file is not found, it will remove one hex digit and try again. Ultimately, it will try looking for a file named default (in lower case). As an example, if the boot file name is /mybootdir/pxelinux.0, the Ethernet MAC address is `88:99:AA:BB:CC:DD` and the IP address 192.0.2.91, it will try following files (in that order):

```
/mybootdir/pxelinux.cfg/01-88-99-aa-bb-cc-dd
/mybootdir/pxelinux.cfg/C000025B
/mybootdir/pxelinux.cfg/C000025
/mybootdir/pxelinux.cfg/C00002
/mybootdir/pxelinux.cfg/C0000
/mybootdir/pxelinux.cfg/C000
/mybootdir/pxelinux.cfg/C00
/mybootdir/pxelinux.cfg/C0
/mybootdir/pxelinux.cfg/C
/mybootdir/pxelinux.cfg/default
```

Note that all filename references are relative to the directory pxelinux.0 lives in. PXELINUX generally requires that filenames (including any relative path) are 127 characters or shorter in length.

PXELINUX does not support MTFTP, and I have no immediate 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.

How Should I Setup my TFTP server?

PXELINUX currently requires that the boot server has a TFTP server which supports the "tsize" TFTP option (RFC 1784/RFC 2349).

Also, please do check out the problematic hardware reference page to see if your PXE stacks need any special workarounds.

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

- The "tftp-hpa" TFTP server, a highly portable update and port of the BSD TFTP server code is available at:

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

- Another TFTP server which supports this 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 runs Windows (and you can't fix that), try tftpd32 by Philippe Jounin: 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 this link.

How Should I Setup My DHCP server?

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.

If this does not work for your configuration, 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.

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:07:0
        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.

Can I send information to PXELINUX via special options in the DHCP response?

PXELINUX (starting with version 1.62) supports the following nonstandard DHCP options, which depending on your DHCP server you may be able to use to customize the specific behaviour of PXELINUX:

- Option 208: **pxelinux.magic** must be set to F1:00:74:7E (241.0.116.126) for PXELINUX to recognize any special DHCP options whatsoever.
- Option 209: **pxelinux.configfile** specifies the PXELINUX configuration file name.
- Option 210: **pxelinux.pathprefix** specifies 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** specifies, in seconds, the time to wait before reboot in the event of TFTP failure. 0 means wait "forever" (in reality, it waits approximately 136 years.)

ISC dhcp 3.0 supports a rather nice syntax for specifying custom options; you can use the following syntax 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;
```

In current versions of PXELINUX, this is supported both as a site-option-space, and as a vendor-option-space.

Inside your PXELINUX-booting group or class (whereever 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.

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".

using 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

using 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";
      }
```

What Happens When a Boot Fails?

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.

How do I keep the PXE stack loaded after boot?

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, PXELINUX will keep the PXE and UNDI stacks in memory. (If you don't know what this means, you probably don't need it.)

What Problems Are There Currently With PXELINUX?

Requires a TFTP server which supports the "tsize" option.

The error recovery routine doesn't work quite right. For right 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 don't work yet... they probably need auxilliary information (such as device) to work at all.

If you have additional problems, please contact the SYSLINUX mailing list. See the SYSLINUX FAQ for details. Before you post something, please make sure you have checked that your kernel files aren't named using the extensions which have special meaning:

```
.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]
```

pxechain.com, as of PXELINUX 4.00, is broken. See also

Common_Problems#pxechain.com.2FPXELINUX-4.00.2B.

PXE stack on a floppy

If your network card doesn't have a PXE boot ROM, there is 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 Etherbot home page is at: http://www.etherboot.org/

```
... and you can use ROM-o-matic to automatically create customized boot ROMs for your needs.
```

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 is 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 I will use the Simple menu system only but it is easy to modify the following to use the vesamenu system or no menu.

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

On RIS Server Create 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
```

eg

```
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 file downloaded extract the file "pxelinux.0" 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 Fixes Menu
  MENU LABEL ^Fixes Menu
  KERNEL menu.c32
  APPEND pxelinux.cfg/graphics.conf pxelinux.cfg/fixes.menu

LABEL Setup Menu
  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
```

Set ALLOWOPTIONS to 1 if you want to be able to edit any of the entries while booted with PXE on the menu system. If you might want that for testing but once its final I like it set to 0. Also set NOESCAPE to 0 for the same reasons.

/tftpboot/pxelinux.cfg/fixes.menu

```
    MENU TITLE Fixes Menu

    LABEL Main Menu
      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 s
```

/tftpboot/pxelinux.cfg/setup.menu

```
    MENU TITLE Setup Menu

    LABEL Main Menu
      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/ir
autoyast=http://IPADDR/9.2/scripts/ay92.xml

    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 splash=verbose
```

```
    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/ir
autoyast=http://IPADDR/9.2/scripts/develdesktop.xml
```

Retrieved from "http://www.syslinux.org/wiki/index.php?title=PXELINUX&oldid=3438"
Categories: PXELINUX | TFTP | Menu

- This page was last modified on 7 June 2012, at 04:10.