Network Booting i386 Unix Variants

Philip Munts
21 January 2010

Contents

Change History

15 January 2001	Updated to reference FreeBSD 4.2.
16 January 2001	Revised section about `freebsd.pxeboot` to reflect FreeBSD 4.2.
1 February 2001	Removed paragraph about `/etc/profile` in FreeBSD section. Fixed a minor typographical error in the Linux section.
13 February 2001	Added customized `/sbin/init` for FreeBSD. Added NetBSD section. Added note to introduction about NFS. Also many minor cleanups.
14 February 2001	Added note about demo server OS. Clarified the boot image creation procedures and added notes about their make files.
27 February 2001	Reverted back to PXE ROM 2.0. Rewrote NetBSD PXE section. Added hyperlinks for BpBatch script files.
7 March 2001	Rewrote Linux section to remove reference to the staging directory. Added note regarding Linux `halt` and `reboot`. Updated to reference Linux kernel 2.2.18.
29 March 2001	Updated to reference Linux kernel 2.2.19.
30 April 2001	Updated to reference FreeBSD 4.3.
10 May 2001	Updated to reference Red Hat Linux 7.1 and kernel 2.4.4.
7 June 2001	Updated to reference Linux kernel 2.4.5.
23 July 2001	Updated to reference Linux kernel 2.4.7.
29 August 2001	Updated to reference Linux kernel 2.4.9.
28 November 2001	Updated to reference Red Hat Linux 7.2 and kernel 2.4.16, and FreeBSD 4.4.
18 January 2002	Removed references to `pxefloppy`.
21 January 2002	Added section for OpenBSD. Fixed dead hyperlinks and added some new ones. Cleaned up some odds and ends.
6 November 2002	Updated to reference Red Hat Linux 8.0 and kernel 2.4.19, FreeBSD 4.7 and OpenBSD 3.1.

Change History (continued)

10 November 2002	Updated URL for the Linux Documentation Project: their old domain seems to have been stolen.
20 December 2002	Use normal `/sbin/init` for Linux. Updated to reference OpenBSD 3.2. Updated to reference Linux kernel 2.4.20. Corrected filenames in OpenBSD `/tftpboot` directory example.
20 January 2004	Updated to reference Red Hat Linux 9 and Linux kernel 2.4.24. Updated to reference OpenBSD 3.4. Also fixed a few stale hyperlinks.
5 April 2005	Updated to reference Etherboot 5.2.5, Fedora Linux Core 3, FreeBSD 4.11, NetBSD 2.0, and OpenBSD 3.6. Updated or deleted stale URL's. Adopted GRUB wholesale as the preferred netboot OS loader. Note: FreeBSD 5.x support is still undergoing investigation.
16 December 2005	Updated to reference Centos 4.1, OpenBSD 3.8, Linux kernel 2.6.14.4. Also corrected a few minor typos and updated some stale URL's.
22 December 2005	Started editing with OpenOffice 2.0.1. Changed CentOS 4.1 to Centos 4. Added note about FreeBSD 6.0.
6 January 2006	Updated to reference Linux kernel 2.6.15, NetBSD 3.0. Removed notes about /lib from FreeBSD, NetBSD, and OpenBSD sections.
19 January 2006	Updated to reference FreeBSD 6.0. Changed "`initrd`" to "`mfsroot`" in FreeBSD section. Deprecated FreeBSD `pxeboot`. Fixed some minor typos and cut-and-paste errors in the FreeBSD NetBSD, and OpenBSD sections.
31 January 2006	Removed some more Linuxisms from the BSD sections. Fixed another cut and paste error in the NetBSD section. Changed references to Windows NT and 2000 to just Microsoft Windows. Updated the paragraph about `/sbin/init` in the FreeBSD section to match that in NetBSD and OpenBSD.
2 February 2006	Made minor cosmetic wording and formatting changes to FreeBSD and OpenBSD sections. Updated some stale URL's in the Linux and FreeBSD sections. Added a link for the Intel PXE spec to the list at the end of the background section.
7 November 2006	Updated to reference OpenBSD 4.0 and Linux kernel 2.6.18.2. Also updated stale hyperlinks.
6 March 2007	Updated to reference FreeBSD 6.2 and Linux kernel 2.6.19.6.

Change History (continued)

16 July 2007	Updated to reference CentOS 5, Linux kernel 2.6.21.6, and OpenBSD 4.1. Added notes about booting in VMware.
21 January 2008	Updated to reference Linux kernel 2.6.23.14 and OpenBSD 4.2.
9 May 2008	Removed mention of NetBSD boot ROM images. Updated to reference Linux kernel 2.6.25.2, NetBSD 4.0, and OpenBSD 4.3. Added notes about GRUB NIC support. Added section about `pxelinux`.
14 May 2008	Updated to reference Linux kernel 2.6.25.3 and FreeBSD 7.0
21 January 2010	Updated to reference Linux kernel 2.6.32.4 and Fedora 11. Switched to BusyBox for nearly all of Linux userland. Checked for stale URL's.

Background Information

Introduction

This document describes how to create network boot images for various popular i386 Unix variants. Each network boot image must satisfy the following requirements:

- Must work with an EtherBoot or PXE 2.0+ boot ROM.
- Must configure a network interface.
- Must obtain its IP address via BOOTP or DHCP.
- Must not require any local file store.
- Must not require an NFS or SMB file server.
- Must mount the root directory `/` from a RAM disk incorporated into the boot image.
- Must be able to mount an NFS file system for any additional software.

The goal of this effort is to create a self-contained system environment that can be downloaded on demand to a unit under test. The system environment provides a command line shell, fundamental Unix programs such as `cat` and `ls`, and basic network connectivity.

Applications for such a self-contained Unix environment might include:

- Manufacturing or diagnostic test of PC-based hardware systems.
- System disk initialization or recovery for PC-based hardware systems.
- Diskless workstation startup.
- Diskless network appliance operating firmware.

The remaining sections of this document describe how to use a self-contained network boot image, and how to create them for several popular Unix variants. The latest revision of this document is available on line at:

http://www.munts.com/diskless/Docs

Theory of Operation

When the unit under test powers up, the ROM BIOS scans adapter cards for ROM extensions. When the ROM BIOS discovers the boot ROM installed in the NIC (Network Interface Card), it makes a procedure call to the code within the boot ROM. The boot ROM initializes the NIC and begins broadcasting DHCP and/or BOOTP requests. Most current boot ROMS are able to generate both DHCP and BOOTP requests, and most current servers are able to generate both responses.

The BOOTP/DHCP server receives the request from the boot ROM and transmits a response containing the IP address of the client, the IP address of the default router, IP addresses of the default name servers, the net mask, and the name of a boot file.

The boot ROM receives the response from the BOOTP/DHCP server and extracts the address information, configuring the IP address and net mask for the NIC. The boot ROM then transfers the specified boot file from the TFTP server to local RAM. After the boot file has been successfully downloaded from the TFTP server, the boot ROM transfers control to the operating system kernel or second stage boot loader within the boot file.

When the operating system kernel begins execution, it configures the system hardware, mounts the root file system from a RAM disk, and runs `/sbin/init`. The `init` program performs final system configuration and finally runs the command line shell program `/bin/sh`. At this point the operator has command line control of a small Unix system and can initiate dedicated test software.

EtherBoot Client Setup

Free (GPL) boot ROMS for a wide variety of NIC's are available from the EtherBoot Project. The EtherBoot boot ROM used for the demonstration systems described below is version 5.2.5 obtained from http://rom-o-matic.net/.

PXE Client Setup

Most Intel and 3com NIC's now come with a PXE boot ROM installed. Most motherboards with an integrated network controller include PXE in the BIOS. PXE boot ROMS for a wide variety of legacy NIC's can be purchased from bootix Technology GMBH.

GRUB Setup

While there are various network boot loaders, GRUB offers a flexible and consistent solution for booting all of Linux, FreeBSD, NetBSD, and OpenBSD with either EtherBoot or PXE boot ROM's. For network booting of diskless systems, we use the GRUB second stage loaders for EtherBoot and PXE, `nbgrub` and `pxegrub` respectively.

The demonstration systems described below use GRUB 0.93 (with a patch for OpenBSD) for EtherBoot (`nbgrub`) or GRUB for PXE (`pxegrub`) built on a CentOS 5 Linux system with the following configuration options:

```
--enable-diskless
--enable-eepro100
--enable-lance
--enable-ns8390
--enable-via-rhine
--enable-preset-menu=../<OS>/grub.conf
```

Other NIC (Network Interface Cards) are supported by grub. Each operating system's `grub.conf` is configured to load `<os>.kernel` (as well as `linux.initrd` for Linux) by default.

Server Setup

The first step is to install a BOOTP (Boot Protocol) or DHCP (Dynamic Host Configuration Protocol) server on the same network as the unit under test. (It possible to configure special bridges such that the BOOTP/DHCP server can reside on another network, but that is beyond the scope of this document.) BOOTP/DHCP servers are available for almost all Unix variants as well as Microsoft Windows.

The second step is to install a TFTP (Trivial File Transfer Protocol) server, usually on the same machine as the BOOTP/DHCP server. TFTP is a lightweight file transfer protocol that runs over UDP (User Datagram Protocol) instead of TCP. It is simple enough to implement in the very small code space of a boot ROM. TFTP servers are available for almost all Unix variants as well as Microsoft Windows .

The demonstration systems described below were tested against an OpenBSD boot server, using the DHCP and TFTP servers included with that operating system.

Links

EtherBoot Project:
http://www.etherboot.org/

EtherBoot ROM image generator:
http://rom-o-matic.net/

GRUB (Grand Unified Bootloader):
http://www.gnu.org/software/grub

Intel PXE Specification
http://www.munts.com/diskless/Docs/pxespec.pdf

PXE utilities from 3COM:
http://support.3com.com/infodeli/tools/nic/mba.htm

bootix Technology GmbH:
http://www.bootix.com

Network Boot Images

Linux

Introduction

The demonstration diskless Linux system is contains userland programs from BusyBox, with a few other programs and libraries from Fedora 11. The Linux kernel was built from the latest stable kernel source release (2.6.32.4) published on www.kernel.org.

The demonstration Linux installation consists of two components: a compressed kernel image file and a compressed RAM disk image file. These components are most easily built on a running Fedora system with loopback device support enabled.

Procedure

1. Build a custom kernel `vmlinuz` for the unit under test.
2. Create an empty file `initrd`, the size of the initial RAM disk.
3. Make a Linux file system on `initrd`.
4. Mount `initrd` on a temporary directory `initrd.d` using the loopback device option `-o loop`.
5. Populate the file system mounted on `initrd.d`.
6. Unmount `initrd`, leaving behind the initial RAM disk file `initrd`.
7. Compress `initrd` with `gzip`.
8. Merge the kernel and initial RAM disk into a tagged image file.

These steps are automatically performed by the `Makefile` in the diskless Linux working directory.

EtherBoot

The EtherBoot ROM can boot a Linux kernel with an initial RAM disk if they are combined into a tagged image file using `mknbi-linux`, a utility program available from the EtherBoot project. The demonstration system `Makefile` creates the tagged image file `linux.nbi` as a final product. It should be copied to the TFTP server home directory and the BOOTP/DHCP server configured to name it as the boot file.

Soft links with the client name as prefix are preferred. For example, for a client computer named `munts1` booting diskless Linux via EtherBoot, install the following files and soft links in the TFTP server home directory:

```
-r--r--r--  1 root  wheel  4856832 Mar 30 16:31 linux.nbi
lrwxr-xr-x  1 root  wheel        9 Mar 30 16:31 munts1 -> linux.nbi
```

- `linux.nbi` is the tagged image file containing the compressed Linux kernel and initial RAM disk images.
- `munts1` is a soft link to `linux.nbi`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

GRUB provides a more flexible boot environment. The `nbgrub` stage 2 loader can be loaded by EtherBoot and in turn load Linux kernel and initial RAM disk images and boot the system:

```
-r--r--r--  1 root  wheel  3897867 Mar 30 16:31 linux.initrd
-r--r--r--  1 root  wheel   951591 Mar 30 16:31 linux.kernel
-r--r--r--  1 root  wheel   121508 Mar 30 16:31 linux.nbgrub
lrwxr-xr-x  1 root  wheel       14 Mar 30 16:31 munts1 -> linux.nbgrub
```

- `linux.initrd` is the compressed Linux initial RAM disk image.
- `linux.kernel` is the compressed Linux kernel image.
- `linux.nbgrub` is the GRUB stage 2 loader for EtherBoot and Linux, compiled to load `linux.kernel` and `linux.initrd` by default.
- `munts1` is a soft link to `linux.nbgrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

PXE

A PXE boot ROM cannot load a Linux kernel directly. GRUB provides a flexible boot loader for PXE that can netboot Linux, provided the particular NIC is supported. The `pxegrub` stage 2 loader can be loaded by the PXE boot ROM and in turn load Linux kernel and initial RAM disk images and boot the system:

```
-r--r--r--  1 root  wheel  3897867 Mar 30 16:31 linux.initrd
-r--r--r--  1 root  wheel   951591 Mar 30 16:31 linux.kernel
-r--r--r--  1 root  wheel   122532 Mar 30 16:31 linux.pxegrub
lrwxr-xr-x  1 root  wheel       14 Mar 30 16:31 munts1 -> linux.pxegrub
```

- `linux.initrd` is the compressed Linux initial RAM disk image.
- `linux.kernel` is the compressed Linux kernel image.
- `linux.pxegrub` is the GRUB stage 2 loader for PXE and Linux, compiled to load `linux.kernel` and `linux.initrd` by default.
- `munts1` is a soft link to `linux.pxegrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

If GRUB does not support a particular NIC, `pxelinux` may be used instead. For demonstration purposes, `pxelinux` has been patched to attempt to fetch its configuration file from `linux.config` before trying other alternatives. This allows for the following arrangement:

```
-r--r--r--  1 root  wheel      226 May  5 22:07 linux.config
-r--r--r--  1 root  wheel  8263009 May  5 22:07 linux.initrd
-r--r--r--  1 root  wheel  1506684 May  5 22:07 linux.kernel
-r--r--r--  1 root  wheel    14674 May  5 22:07 linux.pxelinux
lrwxr-xr-x  1 root  wheel       14 May  9 10:49 munts1 -> linux.pxelinux
```

- `linux.config` is the `pxelinux` configuration file.
- `linux.initrd` is the compressed Linux initial RAM disk image.
- `linux.kernel` is the compressed Linux kernel image.
- `linux.pxelinux` is the patched `pxelinux` binary.
- `munts1` is a soft link to `linux.pxegrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

Notes

The Linux netboot image will boot in a VMware virtual machine or in a VirtualBox virtual machine.

Links

BusyBox Linux Userland:
http://busybox.net

Diskless Linux working directory snapshot:
http://www.munts.com/diskless/Linux

Diskless Linux HOWTO:
http://www.tldp.org/HOWTO/Diskless-HOWTO.html

GRUB (Grand Unified Bootloader):
http://www.gnu.org/software/grub

Loopback Root File system mini-HOWTO:
http://www.faqs.org/docs/Linux-mini/Loopback-Root-FS.html

PXELINUX Home Page:
http://syslinux.zytor.com/pxe.php

FreeBSD

Introduction

The demonstration diskless FreeBSD system is based on a stripped down installation of FreeBSD 7.0, the current version at time of writing. It must be built on a running FreeBSD system with memory file system support enabled.

Procedure

1. Build a custom kernel, with space reserved for a RAM disk, for the unit under test.

2. Create an empty file `mfsroot`, the size of the MFS root file system.

3. Configure `mfsroot` as a vnode device.

4. Make a BSD file system on the vnode device.

5. Mount the vnode device on a temporary directory `mfsroot.d`.

6. Populate the file system mounted on `mfsroot.d`.

7. Unmount `mfsroot.d` and unconfigure the vnode device, leaving behind the MFS root file system image `mfsroot`.

8. Merge `mfsroot` into the custom kernel using a special program: `write_mfs_in_kernel`. This program was formerly supplied with FreeBSD but as of release 5.0 it was removed. It has been reverse engineered and source code is available in the diskless FreeBSD working directory.

These steps are automatically performed by the `Makefile` in the diskless FreeBSD working directory.

EtherBoot

The EtherBoot ROM can boot an uncompressed FreeBSD kernel image directly. Simply copy `freebsd.kernel` to the TFTP server home directory and configure the BOOTP/DHCP server to name it as the boot file.

Soft links with the client name as prefix are preferred. For example, for a client computer named `munts1` booting diskless FreeBSD via EtherBoot, install the following files and soft links in the TFTP server home directory:

```
-r--r--r--  1 root  wheel  10152510 Mar 30 16:31 freebsd.kernel
lrwxr-xr-x  1 root  wheel        14 Mar 30 16:31 munts1 -> freebsd.kernel
```

- `freebsd.kernel` is the uncompressed FreeBSD kernel and RAM disk image.
- `munts1` is a soft link to `freebsd.kernel`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

GRUB provides a more flexible boot environment, provided the particular NIC is supported. The `nbgrub` stage 2 loader can be loaded by EtherBoot and in turn load a compressed FreeBSD kernel image and boot the system:

```
-r--r--r--  1 root  wheel   3805051 Mar 30 16:31 freebsd.kernel
-r--r--r--  1 root  wheel    121476 Mar 30 16:31 freebsd.nbgrub
lrwxr-xr-x  1 root  wheel        14 Mar 30 16:31 munts1 -> freebsd.nbgrub
```

- `freebsd.kernel` is the compressed FreeBSD kernel and RAM disk image.
- `freebsd.nbgrub` is the GRUB stage 2 loader for EtherBoot and FreeBSD, compiled to load `freebsd.kernel` by default.
- `munts1` is a soft link to `freebsd.nbgrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

PXE

A PXE boot ROM cannot load a FreeBSD kernel directly. GRUB provides a flexible boot loader for PXE that can netboot FreeBSD, provided the particular NIC is supported. The `pxegrub` stage 2 loader can be loaded by the PXE boot ROM and in turn load a compressed FreeBSD kernel image and boot the system:

```
-r--r--r--  1 root  wheel   3805051 Mar 30 16:31 freebsd.kernel
-r--r--r--  1 root  wheel    122500 Mar 30 16:31 freebsd.pxegrub
lrwxr-xr-x  1 root  wheel        14 Mar 30 16:31 munts1 -> freebsd.pxegrub
```

- `freebsd.kernel` is the compressed FreeBSD kernel and RAM disk image.
- `freebsd.pxegrub` is the GRUB stage 2 loader for PXE and Linux, compiled to load `freebsd.kernel`.
- `munts1` is a soft link to `freebsd.pxegrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

Notes

The stock FreeBSD `/sbin/init` performs a lot of unnecessary processing, such as transitioning between single-user mode and multi-user mode. Therefore a customized `init` is installed, that merely opens `stdin`, `stdout`, and `stderr`, runs `/etc/rc` and starts a command line shell with `/bin/sh`.

The FreeBSD kernel doesn't recognize the CTRL-ALT-DELETE sequence to reboot the system. However, `halt` and `reboot` do work correctly.

The FreeBSD PXE boot loader `pxeboot` is now deprecated, as it is buggy and complicated to configure.

The FreeBSD netboot image will boot in a VMware virtual machine.

Links

Diskless FreeBSD working directory snapshot:
http://www.munts.com/diskless/FreeBSD

GRUB (Grand Unified Bootloader):
http://www.gnu.org/software/grub

NetBSD

Introduction

The demonstration diskless NetBSD system is based on a stripped down installation of NetBSD 4.0, the current version at time of writing. It must be built on a running NetBSD system with vnode disk support enabled.

Procedure

1. Build a custom kernel, with space reserved for a RAM disk, for the unit under test.
2. Create an empty file `memdisk`, the size of the root file system.
3. Configure `memdisk` as a vnode disk.
4. Make a BSD file system on the vnode disk.
5. Mount the vnode disk on a temporary directory `memdisk.d`.
6. Populate the file system mounted on `memdisk.d`.
7. Unmount `memdisk.d` and unconfigure the vnode disk, leaving behind the root file system image `memdisk`.
8. Merge `memdisk` into the custom kernel using a special program supplied with NetBSD: `mdsetimage`.

These steps are automatically performed by the `Makefile` in the diskless NetBSD working directory.

EtherBoot

An EtherBoot boot ROM cannot load a NetBSD kernel directly. GRUB provides a flexible boot loader for EtherBoot that can netboot NetBSD, provided the particular NIC is supported. The `nbgrub` stage 2 loader can be loaded by the EtherBoot boot ROM and in turn load the NetBSD kernel and boot the system:

```
-r--r--r--  1 root  wheel   2294180 Mar 30 16:31 netbsd.kernel
-r--r--r--  1 root  wheel    121476 Mar 30 16:31 netbsd.nbgrub
lrwxr-xr-x  1 root  wheel        14 Mar 30 16:31 munts1 -> netbsd.nbgrub
```

- `netbsd.kernel` is the compressed NetBSD kernel and RAM disk image.
- `netbsd.nbgrub` is the GRUB stage 2 loader for EtherBoot and NetBSD, compiled to load `netbsd.kernel` by default.
- `munts1` is a soft link to `netbsd.nbgrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

PXE

A PXE boot ROM cannot load a NetBSD kernel directly. GRUB provides a flexible boot environment for PXE that can netboot NetBSD, provided the particular NIC is supported. The `pxegrub` stage 2 loader can be loaded by the PXE boot ROM and in turn load the NetBSD kernel and boot the system:

```
lrwxr-xr-x  1 root  wheel        14 Mar 30 16:31 munts1 -> netbsd.pxegrub
-r--r--r--  1 root  wheel   2294180 Mar 30 16:31 netbsd.kernel
-r--r--r--  1 root  wheel    122500 Mar 30 16:31 netbsd.pxegrub
```

- `munts1` is a soft link to `netbsd.pxegrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.
- `netbsd.kernel` is the compressed NetBSD kernel and RAM disk image.
- `netbsd.pxegrub` is the GRUB stage 2 loader for PXE and NetBSD, compiled to load `netbsd.kernel` by default.

If GRUB does not support a particular NIC, NetBSD also includes a dedicated PXE boot loader, `/usr/mdec/pxeboot`. The `pxeboot` program for NetBSD has an interesting idiosyncrasy compared to those for FreeBSD and OpenBSD: it ignores the filename parameter from the DHCP server unless it is prefixed with "`tftp:`". If the filename parameter does begin with "`tftp:`", then `pxeboot` will strip it off and use the remainder as the name of a compressed NetBSD kernel image to load and boot. This allows the following setup:

```
lrwxr-xr-x  1 root  wheel       13 Mar 30 16:31 munts1 -> netbsd.kernel
-r--r--r--  1 root  wheel  2294180 Mar 30 16:31 netbsd.kernel
-r--r--r--  1 root  wheel    41560 Mar 30 16:31 netbsd.pxeboot
lrwxr-xr-x  1 root  wheel       14 Mar 30 16:31 tftp:munts1 -> netbsd.pxeboot
```

- `munts1` is a soft link to `netbsd.kernel`.
- `netbsd.kernel` is the compressed NetBSD kernel and RAM disk image.
- `netbsd.pxeboot` is the PXE loader for NetBSD, copied from `/usr/mdec/pxeboot`.
- `tftp:munts1` is a soft link to `netbsd.pxeboot`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

Notes

The stock NetBSD `/sbin/init` performs a lot of unnecessary processing, such as transitioning between single-user mode and multi-user mode. Therefore a customized `init` is installed, that merely opens `stdin`, `stdout`, and `stderr`, runs `/etc/rc` and starts a command line shell with `/bin/sh`.

The NetBSD boot ROM's can only load an uncompressed NetBSD kernel image.

The NetBSD kernel doesn't recognize the CTRL-ALT-DELETE sequence to reboot the system. However, `halt` and `reboot` do work correctly.

Links

Diskless NetBSD working directory snapshot:
http://www.munts.com/diskless/NetBSD

GRUB (Grand Unified Bootloader):
http://www.gnu.org/software/grub

Diskless NetBSD HOW-TO:
http://www.netbsd.org/Documentation/network/netboot

OpenBSD

Introduction

The demonstration diskless OpenBSD system is based on a stripped down installation of OpenBSD 4.3, the current version at time of writing. It must be built on a running OpenBSD system with vnode disk support enabled.

Procedure

1. Build a custom kernel, with space reserved for a RAM disk, for the unit under test.
2. Create an empty file `miniroot`, the size of the root file system.
3. Configure `miniroot` as a vnode disk.
4. Make a BSD file system on the vnode disk.
5. Mount the vnode disk on a temporary directory `miniroot.d`.
6. Populate the file system mounted on `miniroot.d`.
7. Unmount `miniroot` and unconfigure the vnode disk, leaving behind the root file system image `miniroot`.
8. Merge `miniroot` into the custom kernel using a special program supplied with OpenBSD: `elfrdsetroot.`

These steps are automatically performed by the `Makefile` in the diskless OpenBSD working directory.

EtherBoot

An EtherBoot boot ROM cannot load an OpenBSD kernel directly. GRUB provides a flexible boot loader for EtherBoot that can netboot OpenBSD, provided the particular NIC is supported. The `nbgrub` stage 2 loader can be loaded by the EtherBoot boot ROM and in turn load the OpenBSD kernel and boot the system:

```
lrwxr-xr-x  1 root  wheel       14 Mar 30 16:31 munts1 -> openbsd.nbgrub
-r--r--r--  1 root  wheel  5239868 Mar 30 16:31 openbsd.kernel
-r--r--r--  1 root  wheel   121476 Mar 30 16:31 openbsd.nbgrub
```

- `openbsd.kernel` is the compressed OpenBSD kernel and RAM disk image.
- `openbsd.nbgrub` is the GRUB stage 2 loader for EtherBoot and OpenBSD, compiled to load `openbsd.kernel` by default.
- `munts1` is a soft link to `openbsd.nbgrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

PXE

A PXE boot ROM cannot load a OpenBSD kernel directly. GRUB provides a flexible boot environment for PXE that can netboot OpenBSD, provided the particular NIC is supported. The `pxegrub` stage 2 loader can be loaded by the PXE boot ROM and in turn load the OpenBSD kernel and boot the system:

```
lrwxr-xr-x  1 root  wheel       15 Mar 30 16:31 munts1 -> openbsd.pxegrub
-r--r--r--  1 root  wheel  5239868 Mar 30 16:31 openbsd.kernel
-r--r--r--  1 root  wheel   122500 Mar 30 16:31 openbsd.pxegrub
```

- `openbsd.kernel` is the compressed OpenBSD kernel and RAM disk image.
- `openbsd.pxegrub` is the GRUB stage 2 loader for PXE and OpenBSD, compiled to load `openbsd.kernel` by default.
- `munts1` is a soft link to `openbsd.pxegrub`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.

If GRUB does not support a particular NIC, OpenBSD also includes a dedicated PXE boot loader, `/usr/mdec/pxeboot`. It loads the compressed kernel `bsd` from the TFTP server.

Soft links with the client name as prefix are preferred. For example, for a client computer named `munts1` booting diskless OpenBSD via PXE, install the following files and soft links in the TFTP server home directory:

```
lrwxr-xr-x  1 root  wheel       14 Mar 30 16:31 bsd -> openbsd.kernel
lrwxr-xr-x  1 root  wheel       15 Mar 30 16:31 munts1 -> openbsd.pxeboot
-r--r--r--  1 root  wheel  5239868 Mar 30 16:31 openbsd.kernel
-r--r--r--  1 root  wheel    53848 Mar 30 16:31 openbsd.pxeboot
```

- `bsd` is a soft link to the compressed OpenBSD kernel.
- `munts1` is a soft link to `pxeboot`, and must be named by the BOOTP/DHCP server as the boot file for client computer `munts1`.
- `openbsd.kernel` is the compressed OpenBSD kernel and RAM disk image.
- `openbsd.pxeboot` is the PXE boot loader loader for OpenBSD, copied from `/usr/mdec/pxeboot`.

Notes

The stock OpenBSD `/sbin/init` performs a lot of unnecessary processing, such as transitioning between single-user mode and multi-user mode. Therefore a customized `init` is installed, that merely opens `stdin`, `stdout`, and `stderr`, runs `/etc/rc` and starts a command line shell with `/bin/sh`..

The OpenBSD kernel doesn't recognize the CTRL-ALT-DELETE sequence to reboot the system. However, `halt` and `reboot` do work correctly.

GRUB requires a patch (available for GRUB 0.93 in the working directory snapshot below) to support current OpenBSD kernels properly.

OpenBSD does not reliably support a miniroot file system larger than 10 MB. Staying within this limitation requires special measures, such as using the crunched binaries from `bsd.rd` whenever possible. See the OpenBSD `Makefile` for details.

The OpenBSD netboot image will boot in a VMware virtual machine.

Links

Diskless OpenBSD working directory snapshot:
http://www.munts.com/diskless/OpenBSD

GRUB (GRand Unified Bootloader):
http://www.gnu.org/software/grub