NAME

INSTALL — Installation procedure for NetBSD/macppc.

CONTENTS

DESCRIPTION

About this Document

This document describes the installation procedure for NetBSD 10.2_STABLE on the *macppc* platform. It is available in four different formats titled `INSTALL.`*`ext`*, where *`.ext`* is one of `.ps`, `.html`, `.more`, or `.txt`:

`.ps`	PostScript.
`.html`	Standard Internet HTML.
`.more`	The enhanced text format used on UNIX-like systems by the `more`(1) and `less`(1) pager utility programs. This is the format in which the on-line *man* pages are generally presented.
`.txt`	Plain old ASCII.

You are reading the *PostScript* version.

Quick install notes for the impatient

This section contains some brief notes describing what you need to install NetBSD 10.2_STABLE on a machine of the macppc architecture.

- Fetch files needed to install NetBSD. The files depend on which model you are using and how you plan to boot your machine. For systems with built-in floppy drives (Open Firmware 1 or 2), fetch the pair of boot floppy images `macppc/installation/floppy/boot1.fs` and `macppc/installation/floppy/boot2.fs`, which include the bootloader and installation kernel. For systems without floppy drives (most are Open Firmware 3), fetch the bootloader `macppc/installation/ofwboot.xcf` and the installation kernel `macppc/binary/kernel/netbsd-GENERIC_MD.gz`. If you have a CD-R, you can fetch the CD image, `NetBSD-10.2_STABLE-macppc.iso`. Alternatively, you may netboot the installation kernel. This process is covered below, in detail.
- The actual binary distribution is in the `macppc/binary/sets/` directory. When you boot the install kernel from floppies, hard drive, or CD-ROM, the installation program can fetch these files for you (using, e.g., ftp) if you have a network connection. There are several other methods to get the binary sets onto your machine.

  You will at a minimum need the following sets: `kern-GENERIC.tgz`, `base.tgz` and `etc.tgz`. In a typical workstation installation you will probably want all the installation sets.
- If your macppc has a floppy drive, create the pair of boot floppies using **`suntar`** (MacOS 9), **`rawrite`** (Windows), or **`dd`** (any UNIX-like system with floppy support). If your system has Open Firmware 3, drag `ofwboot.xcf` and `netbsd-GENERIC_MD.gz` to your hard drive icon (the top level of the drive, not the desktop). If you are using the CD image, burn it now.

The media you just prepared will be used to boot the installation kernel, which contains all the tools required to install NetBSD.

- Determine your machine's model, quirks, and Open Firmware version from the NetBSD/macppc *Model Support webpage:*

  https://www.NetBSD.org/ports/macppc/models.html.

  At present, NetBSD/macppc cannot exist on the same hard drive as Mac OS unless you partition your disk before running the installer. Open Firmware versions prior to 3 require a dedicated NetBSD drive — you must use the entire disk, partitioned with the installation tools. Open Firmware version 3 cannot boot into NetBSD on a drive partitioned with the installation tools, you must partition your disk before running the installer, then select the "**Re-install sets or install additional sets**" option in the installer (selecting the "**Install NetBSD to hard disk**" or "**Upgrade NetBSD on a hard disk**" options will render your drive unbootable). If you are unsure, you may want to read the section below on **Partitioning your hard drive for NetBSD**

- For systems with Open Firmware versions prior to 3, you may need to use Apple's System Disk utility to enter Open Firmware and use your screen and keyboard. To enter Open Firmware, hold down the `COMMAND-OPTION-O-F` keys after the boot chime starts, but before the chime ends. Entering Open Firmware versions prior to 3 is usually the most frustrating part of installation — you may want to read the section below on **Older Open Firmware System Preparation**

  You should have the Open Firmware "`0 >`" prompt on your screen before attempting to boot NetBSD/macppc.

- At the Open Firmware prompt, type the command to boot. To boot from the installation floppies, the command is "**`boot fd:0`**". For the install kernel and bootloader on your hard drive (Open Firmware 3), the command is "**`boot hd:,\ofwboot.xcf netbsd-GENERIC_MD.gz`**".

  For boot CDs, the command is something like "**`boot cd:,\ofwboot.xcf netbsd.macppc`**" (for Open Firmware 3) or "**`boot scsi-int/sd@3:0 NETBSD.MACPPC`**" (for earlier Open Firmware versions). You will need to use the correct case for **`OFWBOOT.XCF`** and **`NETBSD.MACPPC`** depending on how your version of Open Firmware interprets the ISO file system. You may need to replace **`cd`** with **`scsi/sd@3 , scsi-int/sd@3 , ata/atapi-disk ,`** or some other device alias. You should also use the Open Firmware **`dir`** command to confirm that the NetBSD/macppc kernel is called `NETBSD.MACPPC`. You may want to read the section below on **Open Firmware boot syntax**

-

  PowerPC 601 machines need to use separate boot floppies `macppc/installation/floppy/boot601_1.fs` and `macppc/installation/floppy/boot602_2.fs`, a different kernel set `kern-GENERIC_601.tgz`, and a different install kernel `netbsd-INSTALL_601.gz`. The same boot CD can be used but at the boot prompt you must specify the 601 kernel, i.e., replace **`netbsd.macppc`** with **`netbsd.601`**

- For third-party programs which are not part of the base NetBSD distribution, you will want to explore the **`pkgsrc`** package management system, which contains thousands of third-party software applications.

What is NetBSD?

The NetBSD Operating System is a fully functional open-source operating system derived from the University of California, Berkeley Networking Release 2 (Net/2), 4.4BSD-Lite, and 4.4BSD-Lite2 sources. NetBSD runs on many different different system architectures (ports) across a variety of distinct CPU families, and is being ported to more. The NetBSD 10.2_STABLE release contains complete binary releases for most of these system architectures, with preliminary support for the others included in source form. For more information please visit **https://www.NetBSD.org/**.

NetBSD is a completely integrated system. In addition to its highly portable, high performance kernel, NetBSD features a complete set of user utilities, compilers for several languages, the X Window System, firewall software and numerous other tools, all accompanied by full source code.

NetBSD is a creation of the members of the Internet community. Without the unique cooperation and coordination the net makes possible, NetBSD would not exist.

Changes Between The NetBSD 9 and 10 Releases

The NetBSD 10.2_STABLE release provides many significant changes, including support for many new devices, hundreds of bug fixes, new and updated kernel subsystems, and numerous userland enhancements. The result of these improvements is a stable operating system fit for production use that rivals most commercially available systems.

One important new feature in this release is the support for extended attributes and access control lists on FFS file systems.

For new installations the installer will by default disable these features, so the file system is compatible with older NetBSD releases (before 10), and allow other operating systems to mount this file systems at least in read-only mode.

If you want a new installed file system to support extended attributes, change the file system type from "FFSv2" to "FFSv2ea" in the partitioning menu. You can also convert file systems later, using the `fsck_ffs`(8) utility. More details are available in *this guide:*

https://wiki.netbsd.org/tutorials/acls_and_extended_attributes_on_ffs.

If you are upgrading from a version of NetBSD -current please also check the **Compatibility Issues With Previous NetBSD Releases**. It is impossible to completely summarize the massive development that went into the NetBSD 10.2_STABLE release. The complete list of changes can be found in

CHANGES:

https://nycdn.NetBSD.org/pub/NetBSD-daily/netbsd-10/latest/CHANGES

CHANGES-10.1:

https://nycdn.NetBSD.org/pub/NetBSD-daily/netbsd-10/latest/CHANGES-10.1

CHANGES-10.2:

https://nycdn.NetBSD.org/pub/NetBSD-daily/netbsd-10/latest/CHANGES-10.2

CHANGES-10.3:

https://nycdn.NetBSD.org/pub/NetBSD-daily/netbsd-10/latest/CHANGES-10.3

which are also present in the top level directory of the NetBSD 10.2 release tree.

Features to be removed in a later release

The following features are to be removed from NetBSD in the future:

- `groff`(1). Man pages are now handled with `mandoc`(1), and `groff`(1) can still be found in pkgsrc as `textproc/groff`.
- `pf`(4). This packet filter is obsolete and unmaintained in NetBSD. It will be eventually removed due to possible long-standing security issues and lack of multiprocessor support. New installations should use `npf`(7).

The NetBSD Foundation

The NetBSD Foundation is a tax exempt, not-for-profit 501(c)(3) corporation that devotes itself to the traditional goals and spirit of the NetBSD Project and owns the trademark of the word "NetBSD". It supports the design, development, and adoption of NetBSD worldwide. More information on the NetBSD Foundation, its composition, aims, and work can be found at: **https://www.NetBSD.org/foundation/**

Sources of NetBSD

Refer to *mirrors:* **https://www.NetBSD.org/mirrors/**

NetBSD 10.2_STABLE Release Contents

The root directory of the NetBSD 10.2_STABLE release is organized as follows:

`.../NetBSD-10.2_STABLE/`

`CHANGES`	Changes between the 9.0 and 10.0 releases.
`CHANGES-10.0`	Changes between the initial 10.0 branch and final release of 10.0.
`CHANGES-10.1`	Changes between the 10.0 and the 10.1 release.
`CHANGES-10.2`	Changes between the 10.1 and the 10.2 release.
`CHANGES-10.3`	Changes after the release of 10.2.
`CHANGES.prev`	Changes in previous NetBSD releases.
`README.files`	README describing the distribution's contents.
`images/`	Images (ISO 9660 or USB) for installing NetBSD. Depending on your system, these may be bootable.
`source/`	Source distribution sets; see below.

In addition to the files and directories listed above, there is one directory per architecture, for each of the architectures for which NetBSD 10.2_STABLE has a binary distribution.

The source distribution sets can be found in subdirectories of the `source` subdirectory of the distribution tree. They contain the complete sources to the system. The source distribution sets are as follows:

gnusrc	This set contains the "gnu" sources, including the source for the compiler, assembler, groff, and the other GNU utilities in the binary distribution sets.
sharesrc	This set contains the "share" sources, which include the sources for the man pages not associated with any particular program; the sources for the typesettable document set; the dictionaries; and more.
src	This set contains all of the base NetBSD 10.2_STABLE sources which are not in **gnusrc**, **sharesrc**, or **syssrc**.
syssrc	This set contains the sources to the NetBSD 10.2_STABLE kernel for all architectures as well as the `config`(1) utility.
xsrc	This set contains the sources to the X Window System.

All the above source sets are located in the `source/sets` subdirectory of the distribution tree.

The source sets are distributed as compressed tar files. Except for the **pkgsrc** set, which is traditionally unpacked into `/usr/pkgsrc`, all sets may be unpacked into `/usr/src` with the command:

```
# cd / ; tar -zxpf set_name.tgz
```

In each of the source distribution set directories, there are files which contain the checksums of the files in the directory:

`MD5`	MD5 digests in the format produced by the command: **`cksum -a MD5`** *`file`*.
`SHA512`	SHA512 digests in the format produced by the command: **`cksum -a SHA512`** *`file`*.

The SHA512 digest is safer, but MD5 checksums are provided so that a wider range of operating systems can check the integrity of the release files.

NetBSD/macppc subdirectory structure

The macppc-specific portion of the NetBSD 10.2_STABLE release is found in the `macppc` subdirectory of the distribution: `.../NetBSD-10.2_STABLE/macppc/`. It contains the following files and directories:

- `INSTALL.html`
- `INSTALL.ps`
- `INSTALL.txt`
- `INSTALL.more` Installation notes in various file formats, including this file. The `.more` file contains underlined text using the `more`(1) conventions for indicating italic and bold display.
- `binary/`
  - `kernel/`
    - `netbsd-GENERIC.gz` A gzipped NetBSD kernel containing code for everything supported in this release.
    - `netbsd-GENERIC_601.gz` A gzipped NetBSD kernel containing code for PowerPC 601 machines.
    - `netbsd-GENERIC.MP.gz` A gzipped NetBSD kernel containing code for everything supported in this release. This kernel supports SMP on systems with more than one CPU.
    - `netbsd-GENERIC_MD.gz` A gzipped NetBSD kernel containing code for everything supported in this release. It also has a RAM-disk installer. This is the kernel you should use when you want to install via netboot, from a CD-R, or an HFS partition on Open Firmware 3 systems.
    - `netbsd-INSTALL.gz` A gzipped NetBSD kernel containing code only for Open Firmware 1.0.5, 1.1.22, 2.0.x, and 2.4 systems (i.e. those that have floppy drives). It has a RAM-disk installer.
    - `netbsd-INSTALL_601.gz` A gzipped NetBSD kernel containing code only for PowerPC 601 machines, Open Firmware 1.0.5, 1.1.22, 2.0.x, and 2.4 systems (i.e. those that have floppy drives). It has a RAM-disk installer.
    - `netbsd-POWERMAC_G5.gz` A gzipped NetBSD kernel containing code only for uniprocessor PowerPC 970FX and 970MP machines. It is intended for netbooting, uncompressed, via TFTP.
    - `netbsd-POWERMAC_G5.MP.gz` A gzipped NetBSD kernel containing code only for multi-processor PowerPC 970FX and 970MP machines. It is intended for netbooting, uncompressed, via TFTP.
  - `sets/` macppc binary distribution sets; see below.
- `installation/`
  - `floppy/boot1.fs`

`floppy/boot2.fs`
macppc boot and installation floppy images. The first is the bootloader and kernel. The second image has the installation tools. Only usable on models that ship with a floppy drive (pre-Open Firmware 3); see below.

`floppy/boot601_1.fs`
`floppy/boot601_2.fs`
Same as boot1.fs, boot2.fs, except for PowerPC 601 machines only.

`ofwboot.xcf`
macppc bootloader; see below.

Binary distribution sets

The NetBSD macppc binary distribution sets contain the binaries which comprise the NetBSD 10.2_STABLE release for macppc. The binary distribution sets can be found in the `macppc/binary/sets` subdirectory of the NetBSD 10.2_STABLE distribution tree, and are as follows:

base The NetBSD 10.2_STABLE macppc **base** binary distribution. You *must* install this distribution set. It contains the base NetBSD utilities that are necessary for the system to run and be minimally functional.

comp Things needed for compiling programs. This set includes the system include files (`/usr/include`) and the various system libraries (except the shared libraries, which are included as part of the **base** set). This set also includes the manual pages for all of the utilities it contains, as well as the system call and library manual pages.

debug This distribution set contains debug information for all base system utilities. It is useful when reporting issues with binaries or during development. This set is huge, if the target disk is small, do not install it.

etc This distribution set contains the system configuration files that reside in `/etc` and in several other places. This set *must* be installed if you are installing the system from scratch, but should *not* be used if you are upgrading.

games This set includes the games and their manual pages.

kern-GENERIC
This set contains a NetBSD/macppc 10.2_STABLE `GENERIC` kernel named `/netbsd`. You *must* install this distribution set unless you have a PowerPC 601 machine.

kern-GENERIC_601
This set contains a NetBSD/macppc 10.2_STABLE `GENERIC_601` kernel named `/netbsd`. You *must* install this distribution set if you have a PowerPC 601 machine.

kern-GENERIC.MP
This set contains a NetBSD/macppc 10.2_STABLE `GENERIC.MP` kernel, which will use multiple processors (if present), named `/netbsd`.

man This set includes all of the manual pages for the binaries and other software contained in the **base** set. Note that it does not include any of the manual pages that are included in the other sets.

misc This set includes the system dictionaries, the typesettable document set, and other files from `/usr/share`.

modules This set includes kernel modules to add functionality to a running system.

rescue This set includes the statically linked emergency recovery binaries installed in `/rescue`.

text	This set includes NetBSD's text processing tools, including `groff`(1), all related programs, and their manual pages.

NetBSD maintains its own set of sources for the X Window System in order to assure tight integration and compatibility. These sources are based on X.Org. Binary sets for the X Window System are distributed with NetBSD. The sets are:

xbase	The basic files needed for a complete X client environment. This does not include the X servers.
xcomp	The extra libraries and include files needed to compile X source code.
xdebug	This distribution set contains debug information for all X11 binaries. It is useful when reporting issues with these binaries or during development. This set is huge, if the target disk is small, do not install it.
xfont	Fonts needed by the X server and by X clients.
xetc	Configuration files for X which could be locally modified.
xserver	The X server. This includes the modular Xorg server.

The macppc binary distribution sets are distributed as gzipped tar files named with the extension **.tgz**, e.g. `base.tgz`.

The instructions given for extracting the source sets work equally well for the binary sets, but it is worth noting that if you use that method, the filenames stored in the sets are relative and therefore the files are extracted *below the current directory*. Therefore, if you want to extract the binaries into your system, i.e. replace the system binaries with them, you have to run the **`tar -xzpf`** command from the root directory (`/`) of your system.

Note: Each directory in the macppc binary distribution also has its own checksum files, just as the source distribution does.

NetBSD/macppc System Requirements and Supported Devices

Currently, NetBSD/macppc requires the use of Open Firmware to boot. Open Firmware is a command environment using the FORTH language. The NetBSD kernel uses Open Firmware to gather information about your system and to control some of your devices. It is part of the boot ROMs in most PowerPC-based Macintosh systems. Until late 1996, Apple never intended to use Open Firmware for anything other than internal debugging and hardware support. It was not intended to be used to boot an operating system. This is why earlier machines have so much trouble with Open Firmware. This also means that PowerMacs and clones that lack Open Firmware cannot boot NetBSD on the macppc platform. Most machines introduced by Apple and the clone-makers after August 17, 1995 have Open Firmware and are supported.

Apple made several revisions of this Open Firmware environment, and each has various quirks and problems that we must work around. The single hardest step of installing NetBSD/macppc is to set up Open Firmware properly. Open Firmware versions 1.0.5 and 2.0.x act similarly and the same set of instructions applies to them. Open Firmware version 2.4 is slightly different with regards to booting. Open Firmware version 3 is altogether different, but easier to set up for NetBSD.

The minimal configuration requires 16 MB of RAM and 160 MB of disk space. To install the entire system requires considerably more, plus space for the swap partition (usually the RAM size, unless you've got a lot of RAM). To use X, 32MB of RAM is required. NetBSD with 16 MB of RAM is very slow. Until you have around 32 MB of RAM, getting more RAM is more important than getting a faster CPU.

Supported models

Find your model from the list below and take note of its Open Firmware version:

Open Firmware 1.0.5

- Apple PowerMacintosh (7200, 7300, 7500, 7600, 8500, 8600, 9500, and 9600)
- Apple Workgroup Server 8550
- Power Computing (PowerCenter, PowerCenter Pro, PowerCurve, PowerTower, PowerTower Pro, and PowerWave)
- UMAX (J700, S900)

Open Firmware 1.1.22

- Apple Network Server (500 and 700)

  Note: Users have reported that the NetBSD kernel does not work well with the on-board MACE-based ethernet (the one that requires an Apple AAUI dongle). If your system does not have the Apple PCI Ethernet Card, you may need to purchase and install another ethernet card in your Apple Network Server.

Open Firmware 2.0.x

- Apple PowerBook (2400, 3400, G3, and G3 Series)
- Apple PowerMacintosh/Performa (4400, 54xx, 5500, 6300/160, 6360, 6400, and 6500)
- Apple PowerMacintosh (G3 "Beige" models with ATI RageII+: Desktop, Mini Tower, and All-in-One)
- APS Tech (M∗Power 604e/200)
- Motorola StarMax (3000, 4000, 5000, and 5500)
- Power Computing (PowerBase)
- UMAX (Apus 2000, Apus 3000, C500, and C600)

Open Firmware 2.4

- Apple PowerMacintosh (G3 "Beige" models with ATI Rage Pro: Desktop, Mini Tower, and All-in-One)

Open Firmware 3

- Apple original iBook, iBook SE, iBook (Dual USB), iBook (Late 2001), iBook (16VRAM), iBook (Opaque 16 VRAM), iBook (32 VRAM), iBook (14.1 LCD 32 VRAM), and iBook G4
- Apple iMac; Bondi Blue (Rev A and Rev B, 233 MHz), 5 Flavors (Rev C and Rev D, 266 MHz and 333 MHz), iMac (Slot Loading), iMac (Summer 2000), iMac (Early 2001), iMac (Summer 2001), and iMac (USB 2.0)
- Apple original eMac and eMac (ATI Graphics)
- Apple PowerBook (G3 Series (bronze keyboard) and G3 (FireWire))
- Apple PowerBook (G4 (Titanium), G4 (Gigabit Ethernet), G4 (DVI), G4 (12-inch), G4 (17-inch), G4 (12-inch DVI), PowerBook G4 (15-inch FW 800), PowerBook G4 (17-inch 1.33GHz/1.5GHz))
- Apple PowerMacintosh G3 (Blue and White)
- Apple PowerMacintosh (G4 (PCI), G4 (AGP), G4 (Gigabit Ethernet), G4 (Digital Audio), G4 (Quicksilver), G4 (Quicksilver 2002), G4 (Mirrored Drive Doors), G4 (FW 800), and G4 Cube)
- Apple Mac Mini G4 (1.25GHz/1.42GHz/1.5GHz)
- Apple original Xserve and Xserve (Slot Load)

  Note: Some newer G4 models may not run at their full CPU speed, due to the fact Open Firmware starts the CPU at a reduced frequency.

- Apple iMac G5, PowerMacintosh G5, Xserve G5

  Note: G5 models are supported as a 32-bit port in bridge mode only.

Unsupported models

- Systems with a PowerPC microprocessor, but lacking Open Firmware
  - Apple PowerBook (1400, 2300, and 5300)
  - Apple PowerMacintosh/Performa (52xx, 53xx, 62xx, and 63xx (except 6300/160 and 6360 which are supported))
  - Apple PowerMacintosh/Performa (61xx, 71xx, and 81xx)
  - PowerComputing (Power 100 and Power 120)
- Systems with broken Open Firmware (Twentieth Anniversary Macintosh)
- Systems with a Motorola 68k microprocessor (these systems are supported by NetBSD/mac68k)
- Systems upgraded from any of the above (unless the motherboard is replaced as part of the upgrade)

Supported devices

- CPU upgrades
  - Only one CPU in a multiprocessor system is supported with the GENERIC kernel. For the 601, only one CPU is supported with the GENERIC_601 kernel. All CPUs (except the 601) are supported with the GENERIC.MP kernel.
  - Most CPU upgrades are supported. Some older models will not automatically enable the L2 and/or L3 caches on a G3 or G4 processor. See the FAQ entry

    http://www.NetBSD.org/ports/macppc/faq.html#cpu-upgrade

- Ethernet
  - On-board 79C950-based MACE Ethernet interface (*mc*)
  - On-board bmac Ethernet interface (*bm*)
  - On-board gmac Ethernet interface (*gem*)
  - Apple PCI Ethernet Card (option for Apple Network Server) (*tlp*)
  - Asante Mac 10/100 PCI Rev A, part number 09-00169-01 (*tlp*)
  - Farallon Fast EtherTX 10/100, part number PN996L-TX (*tlp*)
  - SMC Etherpower II (9432TX) (*epic*)
  - SMC 83c170 (*epic*)
  - 3Com 3c905 (*ex*)
  - Intel EtherExpress PRO/10+ PCI LAN Adapter (*fxp*)
  - Realtek 8029 Ethernet (*ne*)
  - VIA Technologies VT86C926 (*ne*)
  - D-Link DFE-530TX+ (*rtk*)
  - Realtek 8139 (*rtk*)
  - Netgear FA-311 (*sip*)
  - Lite-On PNIC (*tlp*)
  - D-Link DFE-530TX (*vr*)
  - Many other PCI and Cardbus Ethernet interfaces, such as Tulip-compatible (*tlp*), 3Com (*ep*), SMC (*epic*), Intel (*fxp*), NE2000-compatible (*ne*), and Realtek (*rtk*)
  - Many USB Ethernet interfaces (*aue*, *cue* and *kue*)

- Wireless
  - On-board AirPort or AirMac 802.11b (*wi*)

  - AT&T/Lucent WaveLan IEEE PCMCIA card (*wi*)
- SCSI
  - On-board NCR 53c94 SCSI controller (*esp*)
  - On-board MESH SCSI controller (*mesh*)
  - Adaptec PCI controllers 291x, 2920, 2930C, 294x, 295x, 39xx, 19160, 29160 and AIC-78xx (*ahc*)

    Note: The 294x models are not bootable in Open Firmware, even though they can boot MacOS 9 and earlier.

  - AdvanSys 1200[A,B], 9xx[U,UA] SCSI controller (*adv*)
  - AMD 53c974 (*pcscp*)
  - Apple 53c875 Ultra Wide SCSI (shipped in some Beige G3 models) (*ncr* or *siop*)

    Note: This card may need a firmware update to boot NetBSD. Use the Mac OS X SCSI Card Updater from

    http://docs.info.apple.com/article.html?artnum=25176
  - NCR/Symbios 53C8xx (*siop* or *esiop*)
  - Many other PCI SCSI controllers should work, but no one has tried them
  - Most SCSI disk/tape/CD-ROM devices should work
- IDE
  - On-board IDE controllers
  - Promise Ultra66 (*pciide*) (this, and other PC-based cards are not bootable on macppc)
  - Acard ATP-850/860 based IDE controllers (*pciide*) (including the Acard AEC-6210/6260 and the Sonnet Tempo ATA/66 cards)
  - Some other PCI IDE controllers should work, although no one has had much success
  - Most IDE disk/CD-ROM/ATAPI devices should work
- Input devices
  - Most ADB keyboards, mice, trackballs, and trackpads
  - Most USB keyboards, mice, trackballs, and trackpads
  - Most PS/2 keyboards, mice, and trackballs (middle button on 3-button mice may not work)
- Video
  - On-board video on most models (*ofb*)

    Note: Several models have been reported to not work with NetBSD if the on-board video is in use, such as the Performa 54xx, 6360, 6400, PowerComputing PowerCenter and PowerTower, and UMAX C600 and Apus 3000. Also, the video card that shipped with the PowerMacintosh 9500/150, 9500/180, and 9500/200 systems is reported to have the same problem.

  - PCI frame buffers which have Open Firmware support (ATI, IMS, Matrox, and NVIDIA have several models which work)
  - Valkyrie and Platinum framebuffer drivers (*valkyriefb and platinumfb*)
  - Full wscons capabilities, such as virtual terminals and color text are available. (Note however that switching between virtual terminals while running an X session may not be supported by all drivers.)
- Audio
  - On-board audio on most models (*awacs* or *snapper*)

  - USB audio devices (*uaudio*)
  - PCI audio cards, although only the Ensoniq AudioPCI (*eap*) has been thoroughly tested
- Serial ports
  NetBSD/macppc refers to the on-board serial ports as (*tty00*) and (*tty01*). Firmware uses the names listed below.
  - On-board serial ports (the modem and printer ports) (*ttya* and *ttyb*)
    ttya is the mini-DIN 8 serial port with the 'Modem' icon, and *ttyb* is has the 'Printer' icon. Some models with internal modems have the serial port with the 'Modem' icon covered over, and the modem is *ttya*.

    Note: The on-board serial ports can be used for console (38400 bps, 8 bits, no parity, 1 stop bit, no handshaking), although many users have reported problems trying to run ppp or other high speed serial applications
  - The Xserve serial port.

    Note: This serial port can be used for console (57600 bps, 8 bits, no parity, 1 stop bit, no handshaking). Use (*scca*) for the Open Firmware 'input-device' and 'output-device' variables. Alternatively, if you remove your video card, Open Firmware will default to using the serial port for the console.
  - The Gee Three Stealth Serial Port, possibly the Griffin Technology gPort

    Note: These serial ports can be used for console (57600 bps, 8 bits, no parity, 1 stop bit, no handshaking). Use (*scca*) for the Open Firmware 'input-device' and 'output-device' variables.
  - Some USB, PCI, and Cardbus serial ports should work, but no one has tried them
- USB devices
  - Most MI USB devices should work (such as disks, printers, input devices, SCSI adapters, and ethernet interfaces)
    http://www.NetBSD.org/support/hardware/usb.html
- PCMCIA and Cardbus cards
  - Most MI PCMCIA and Cardbus cards should work, although very few have been tested with NetBSD/macppc
    http://www.NetBSD.org/support/hardware/cardbus.html
    http://www.NetBSD.org/support/hardware/pcmcia.html
- PCI cards
  - Most MI PCI cards should work, although very few have been tested with NetBSD/macppc
    http://www.NetBSD.org/support/hardware/pci.html

  Note: While the NetBSD kernel may support the various Cardbus, PCI, PCMCIA, and USB devices you may have, Open Firmware does *not* unless it has a specific Open Firmware ROM. This means you cannot boot from these devices. Some Adaptec SCSI controllers have bootable Open Firmware ROMs.

Unsupported devices

- Floppy disk

  Note: Though NetBSD/macppc can boot from a floppy, there is no kernel support yet

- Advanced power management (cannot put system to 'sleep')
- On-board video capture
- AirPort Extreme
- On-board SATA controller on G5 systems (these machines must be netbooted)

Supported boot devices and media

Each version of Open Firmware supports different devices and media that you may boot from. We define the terms "bootable media" as the media (hard drive, floppy, CD-R, ethernet) that will be used to bootstrap your macppc system into NetBSD, and "distribution sets" or "distribution media" as the media (hard drive, CD-R, ethernet) that contains the files that will be installed to generate a working NetBSD system onto your destination media.

Go to the NetBSD/macppc Model Support webpage and look up your system. Take note of the comments about your model and keep these in mind during the rest of this installation procedure.

http://www.NetBSD.org/ports/macppc/models.html

Getting the NetBSD System on to Useful Media

You should wait to decide where to put the NetBSD distribution sets until you have figured out how you are going to boot your system. Refer back to this section after you have done so.

Note: Some Mac OS **`ftp`** clients default to downloading files in 'ASCII' mode. This will render the NetBSD files useless. Make sure to set your ftp program to download in 'binary' mode.

Installation is supported from several media types, including:

- CD-ROM / DVD / USB stick
- FTP
- Remote NFS partition
- Tape
- Existing NetBSD or MacOS X UFS partitions, if performing an upgrade

The steps necessary to prepare the distribution sets for installation depend upon which installation medium you choose. The steps for the various media are outlined below.

CD-ROM / DVD / USB stick

Find out where the distribution set files are on the CD-ROM, DVD or USB stick. Likely locations are `binary/sets` and `macppc/binary/sets`. (You only need to know this if you are mixing installer and installation media from different versions - the installer will know the proper default location for the sets it comes with).

Proceed to the instructions on installation.

FTP

The preparations for this installation/upgrade method are easy; all you need to do is make sure that there's an FTP site from which you can retrieve the NetBSD distribution when you're about to install or upgrade. If you don't have DHCP available on your network, you will need to know the numeric IP address of that site, and, if it's not on a network directly connected to the machine on which you're installing or upgrading NetBSD, you need to know the numeric IP address of the router closest to the NetBSD machine. Finally, you need to know the numeric IP address of the NetBSD machine itself.

Once you have this information, you can proceed to the next step in the installation or upgrade process. If you're installing NetBSD from scratch, go to the section on preparing your hard disk, below. If you're upgrading an existing installation, go directly to the section on upgrading.

NFS

Place the NetBSD distribution sets you wish to install into a directory on an NFS server, and make that directory mountable by the machine on which you are installing or upgrading NetBSD. This will probably require modifying the `/etc/exports` file on the NFS server and resetting its mount daemon (mountd). (Both of these actions will probably require superuser privileges on the server.)

You need to know the numeric IP address of the NFS server, and, if you don't have DHCP available on your network and the server is not on a network directly connected to the machine on which you're installing or upgrading NetBSD, you need to know the numeric IP address of the router closest to the NetBSD machine. Finally, you need to know the numeric IP address of the NetBSD machine itself.

Once the NFS server is set up properly and you have the information mentioned above, you can proceed to the next step in the installation or upgrade process. If you're installing NetBSD from scratch, go to the section on preparing your hard disk, below. If you're upgrading an existing installation, go directly to the section on upgrading.

Tape

To install NetBSD from a tape, you need to make a tape that contains the distribution set files, in 'tar' format.

If you're making the tape on a UNIX-like system, the easiest way to do so is probably something like:

```
# tar -cf tape_device dist_sets
```

where `tape_device` is the name of the tape device that represents the tape drive you're using. This might be `/dev/rst0`, or something similar, but it will vary from system to system. In the above example, `dist_sets` is a list of filenames corresponding to the distribution sets that you wish to place on the tape. For instance, to put the **kern-GENERIC, base,** and **etc** distributions on tape (the absolute minimum required for installation), you would do the following:

```
# cd .../NetBSD-10.2_STABLE
# cd macppc/binary
# tar -cf tape_device kern-GENERIC.tgz
base.tgz etc.tgz
```

Note: You still need to fill in `tape_device` in the example.

Once you have the files on the tape, you can proceed to the next step in the installation or upgrade process. If you're installing NetBSD from scratch, go to the section on preparing your hard disk, below. If you're upgrading an existing installation, go directly to the section on upgrading.

Preparing your System for NetBSD installation

Prepare yourself

Take a deep breath.

Good. Now, make sure you are reading the PostScript or HTML version of this document, as the `.txt` and `.more` versions lack important formatting information that will prevent you from following the twisted path documented below. OK, good. Now, print out this document.

While it's printing, get some coffee, relax a bit, and mentally prepare yourself for something that promises to be confusing, frustrating, and annoying. If you assume the worst, you'll be pleasantly surprised when everything works easier than you expected. Also, forget everything you've been told about installing NetBSD/macppc. That's right, flush your knowledge cache -- some of it is almost certainly dirty.

Some of this document assumes familiarity with MacOS, e.g. how to download BinHex files and extract things with StuffIt Expander. If you've never heard of those terms before, it is possible to install NetBSD/macppc without booting or knowing how to use MacOS, but depending on your model it may be almost as painful as learning a little MacOS. See the FAQ for more help:

http://www.NetBSD.org/ports/macppc/faq.html#macos-newbie

It's done printing? Fine, time to get started.

The recommended installation procedure is as follows:

1. Go to the NetBSD/macppc model support page and look up your model information and issues (I can't stress this enough times).
   http://www.NetBSD.org/ports/macppc/models.html
2. Spend about 15 minutes and read through this document from this point onward. There will be many pages of instructions that apply to Open Firmware versions that you do not have. Throw them out. Do not read them — at best you will be confused, at worst you may damage your system, requiring repairs.
3. Now create your bootable media and media for the distribution sets.
4. Prepare your machine, depending on the instructions for your model (for example: partition your drive, download and run System Disk, or hook up a serial console).
5. Boot into Open Firmware and verify that it has the version of Open Firmware you think it does. Also make sure that the other variables are set correctly.
6. Figure out the correct boot command and boot your machine from the bootable media you just created. If you're having trouble, be sure to read the section on **Common Problems and Error Messages**
7. Celebrate! The worst is over, but you've still got some work to do. Take a break, maybe more coffee, maybe a quick walk around the block, whatever turns your fancy.
8. Now use the installer to install the distribution sets onto your system and do some initial configuration.
9. Figure out how to boot from the installed partition. Boot into NetBSD for the first time.
10. You may then boot into Open Firmware and set it to always boot your favorite operating system.
11. Configure to your preferences, install your favorite packages, and have fun with your new NetBSD/macppc machine!

Note: You really actually truly do need to follow the procedure listed in this document in the order that we describe. These systems are rather tricky to boot for the novice and expert alike. Once you cross off the sections that don't apply to you, it will make more sense.

Note: If the instructions in a subsection below do not apply to *all* versions of Open Firmware. There will be a line listing which versions of Open Firmware they apply to, such as:
(Open Firmware 1.0.5, Open Firmware 1.1.22, Open Firmware 2.0.x, Open Firmware 2.4, Open Firmware 3)

If you do not have an Open Firmware 3 system, skip down to **Preparing your Open Firmware 1.x or 2.x System for NetBSD**

Preparing your Open Firmware 3 System for NetBSD

Updating your BootROM

Open Firmware 3 systems have a rewritable "firmware", also called the BootROM. When you use an Apple firmware updater, it updates the BootROM. This will *not* change the version of Open Firmware in your machine — it will still be Open Firmware 3. The BootROM is what is first executed when you power on or reset your system. The BootROM then loads Open Firmware, which boots your operating system.

Go to the 'Apple Support Downloads' website at **http://support.apple.com/downloads/** and search for 'firmware' and install the most recent version for your model. For most G3 and G4 models, you will need to run the FirmWare updater from MacOS 9.

Note: If you accidentally change the **`load-base`** or **`real-base`** Open Firmware variables and reset your machine you will, in effect, rewrite the BootROM with garbage. This will permanently damage your machine. We recommend not doing this.

Note: The most recent BootROMs available (4.1.7 and later) are a little picky about RAM. Initially, some PowerMacintosh G3 users found that their third-party RAM had been disabled, but the RAM vendors brought their RAM up to spec and it hasn't been much of an issue since then.

There is one report that FirmWare Update 4.1.9 on iMac (Summer 2000) will prevent the CD-ROM and the hard drive from operating together. You may get `wdc0:0:1: lost interrupt` problems.

Getting to Open Firmware 3

Hold down a special four-key combination when your system boots.

After the chime starts, but before it stops, hold down the four COMMAND-OPTION-O-F keys (the COMMAND key looks like a four-leaf clover or an open apple, and the OPTION key may look like a two-way switch with four straight line segments or say ALT) until you see the Open Firmware command prompt on your screen:

```
Apple PowerBook3,1 2.1f1 BootROM built on 01/29/00 at 22:38:07
Copyright 1994-2000 Apple Computer, Inc.
All Rights Reserved

Welcome to Open Firmware.
To continue booting, type "mac-boot" and press return.
To shut down, type "shut-down" and press return.
 ok
0 >
```

Now, set your system to always stop at the Open Firmware prompt.

```
0 > setenv auto-boot? false
```

Alternatively, if you are currently running MacOS X or Darwin, you can use the **`nvram`** command to set this variable before rebooting.

```
# nvram auto-boot\?=false
```

You will need to escape the question-mark or enclose the whole **nvram** argument in double-quotes to prevent your shell from trying to interpret it.

Setting up Open Firmware 3 to boot NetBSD

This section describes some steps you must take to prepare Open Firmware to boot NetBSD. Additional resources are available in the FAQ regarding how to use the Open Firmware command environment, and the Open Firmware variables you may be using:

http://www.NetBSD.org/ports/macppc/faq.html#ofw-access
http://www.NetBSD.org/ports/macppc/faq.html#ofw-variables

Double-check your Open Firmware version:

```
0 > dev /openprom
0 > .properties
name                    openprom
device_type             BootROM
model                   OpenFirmware 3
relative-addressing
supports-bootinfo
 ok
```

If you will be netbooting your system, you can look up your MAC address.

```
0 > dev enet
0 > .properties
[...]
local-mac-address   CCCCCCCC CCCC
[...]
```

Note: Some Open Firmware 3 machines have their MAC address stored incorrectly (little- vs. big-endian problem). If you look up your MAC address in MacOS, it will be different than what Open Firmware 3 uses to contact your netboot server. Your machine will still work, but its MAC address may conflict with another ethernet device on your network.

You can check your Open Firmware settings with the `printenv` command:

```
0 > printenv
-------------- Partition: common -------- Signature: 0x70 ---------------
little-endian?      false               false
real-mode?          false               false
auto-boot?          false               true
diag-switch?        false               false
[...]
use-nvramrc?        true                false
real-base           -1                  -1
[...]
input-device        keyboard            keyboard
output-device       screen              screen
```

Available Boot Media

Open Firmware is capable of booting from a variety of media (such as hard drives, CD-ROMs, and ethernet). Open Firmware is able to boot files from a variety of file systems (such as ISO9660, HFS, HFS+, and MS-DOS FAT). Unfortunately, Open Firmware is *not* able to directly boot from the NetBSD file system (FFS) or Apple's BSD-based file system (UFS), so we must put the bootloader in a location that Open Firmware is capable of understanding.

Therefore, to boot the NetBSD kernel, Open Firmware must first load a 'bootloader' (`ofwboot.xcf`) which knows how to load the NetBSD kernel.

Note: Despite a note to the contrary in earlier releases that `ofwboot.elf` is obsoleted, that is no longer the case. It has been observed that on some newish machines (notably at least some Mac Minis), use of `ofwboot.xcf` will result in an early kernel panic when loading the GENERIC kernel from disk. For these machines it appears to be a workaround to use `ofwboot.elf` instead of `ofwboot.xcf` as the boot loader. The root cause for this problem has unfortunately yet to be found.

The following bootable media are available for loading the bootloader:

- *HFS or HFS+ file system*

  This method loads `ofwboot.xcf` from an HFS or HFS+ partition which then loads the kernel from an acceptable location. The kernel (compressed or non-compressed) may be on an HFS or HFS+ partition. Due to Open Firmware pickiness, it must be on the same partition as `ofwboot.xcf`.

- *MS-DOS file system*

  In this method, Open Firmware loads the `ofwboot.xcf` bootloader from an MS-DOS file system. It may then load a NetBSD kernel from the same MS-DOS file system. This has only been thoroughly tested on floppy disks, but may work on Zip disks or FDISK-formatted hard drives. This does not work for MS-DOS partitions on a hard drive with an Apple Partition Map.

- *Ethernet (network boot)*

  You can run your entire system diskless or netboot only the files necessary to boot (i.e. the bootloader and the installation kernel). You must have root access for the UNIX-like netboot server, which must be on the same subnet as your NetBSD/macppc machine.

  Note: You must use your on-board ethernet device for netbooting. While there may be some PCI or Cardbus ethernet interfaces with Open Firmware ROMs, no one has reported success netbooting using these devices

- *Hybrid HFS/ISO9660 CD-ROM file system*

  Load the `ofwboot.xcf` bootloader from the CD-ROM's HFS file system. It can then load a NetBSD kernel from the HFS or ISO9660 file system.

  Make sure that the CD has an Apple Partition Map.

  Note: Open Firmware can only open files on the first session of a multi-session CD-R

Once the bootloader is loaded, it can open the kernel from one of the following sources:

- *Ethernet (NFS)*
- *NetBSD FFS file system*
- *MacOS X UFS file system*
- *HFS or HFS+ file system*
- *ISO9660 CD-ROM file system*
- *MS-DOS file system*

The boot CD image provided has both a 'partition zero' bootloader and `ofwboot.xcf` on a hybrid partition so it can be booted on all Open Firmware versions. It also has an ISO9660 file system with an installation kernel and the distribution sets.

Partitioning your hard drive for NetBSD

You must have at least one disk that was partitioned before running the NetBSD installer.

This is the drive that will have the bootloader, `ofwboot.xcf`. Your NetBSD partitions may either be on this same disk (using the method described in the rest of this section), or on a separate disk accessible only to NetBSD.

This section describes how to make a single disk usable by both MacOS and NetBSD — this is necessary for machines which have only one hard drive (such as the eMac, iBook, iMac, PowerBook, and PowerMacintosh Cube systems). If you do not want MacOS you must still follow this procedure, but create only a small HFS+ partition (large enough for the bootloader).

There are two partitioning tools available for NetBSD/macppc, `disklabel`(8) and `pdisk`(8). The former is used in the NetBSD **sysinst** installer, and will render your disk unusable by MacOS. It will also prevent Open Firmware 3 machines from booting from that disk. When running the installer, you will need to use the installer's "Re-install sets" option to skip the **disklabel** step.

Do not use **disklabel** or "Re-install sets" unless you will use one drive for NetBSD only and have another drive which will have the bootloader.

The process is more fully detailed in the Partitioning HOW-TO:
http://www.NetBSD.org/ports/macppc/partitioning.html

You can create a partition map with `pdisk`(8), but the disk will not be usable with MacOS 9 and earlier. If this is a concern, you will need to use Apple's Drive Setup or Disk Utility.

If you are using Apple's Drive Setup tool, make sure you have version 1.8.1 or later. This tool only runs under MacOS 9 and earlier. Drive Setup will erase the contents of your drive — it does not preserve data from any of your partitions.

Apple's Disk Utility only runs under MacOS X 10.0.0 and later. Make sure you click the "Install Mac OS 9 Disk Drivers" checkbox. Also, keep in mind that Disk Utility does *not* create the partitions that NetBSD/macppc requires. After creating the initial partition map with Disk Utility, you will need to use the NetBSD **pdisk** to change the partition types. Also, Disk Utility will erase the contents of your drive — it does not preserve data from any of your partitions.

pdisk is the most flexible (and most difficult to use) partitioning tool available. It runs on almost all OSes that macppc machines support. Download it:
https://cdn.NetBSD.org/pub/NetBSD/arch/macppc/netbsd-pdisk/
https://cdn.NetBSD.org/pub/NetBSD/arch/macppc/macos-utils/pdisk.sea.hqx

There is built-in help describing how it works. When it asks you to enter the "`Type of partition`", use `Apple_UNIX_SVR2` for NetBSD partitions, `Apple_HFS` for HFS and HFS+ partitions, and `Apple_UFS` for UFS partitions.

After you've written the partition map with **pdisk**, you will need to create the file systems. Use `newfs`(8) and `mount`(8) for NetBSD file systems, and **hfstools** to create and mount HFS file systems.
https://cdn.NetBSD.org/pub/NetBSD/packages/pkgsrc/sysutils/hfsutils/

Make the following partitions:

HFS or *HFS+*	Must be large enough to hold the bootloader, over 100 KB. May be as large as you desire for MacOS usage.
A/UX Root	Must be at least 20 MB. Alternatively, you may decide to use one partition for your entire NetBSD installation, in which case it should be at least 200 MB. NetBSD interprets an *A/UX Root* partition as the first partition (*a*) on the disk. This partition is

	not readable from MacOS.
A/UX Swap	Any size. The recommendation is your RAM size, although this is not strictly necessary for machines with a lot of RAM. NetBSD interprets an *A/UX Swap* partition as the second partition (*b*) on the disk. This partition is not readable from MacOS.
A/UX User and *A/UX Free1*	Use these for any additional partitions you may want to use under NetBSD, such as `/usr` (at least 200 MB), `/home`, `/usr/local`, or `/usr/pkg`. NetBSD interprets these partitions as normal NetBSD-style partitions. These partitions are not readable from MacOS.
HFS	Any size. You may want to leave an additional partition available to transfer files between MacOS and NetBSD. If would like to create such a partition, then see the Partitioning HOW-TO. **http://www.NetBSD.org/ports/macppc/partitioning.html#msdos**
UFS	Any size. *UFS* partitions are not readable from MacOS versions prior to X 10.0.0. If you use an *UFS* partition as your root, then it may not be recognized by the NetBSD kernel as the first partition (*a*) on the disk. You will need to compile a new kernel with the root partition explicitly defined to be the *UFS* partition.

Now would be a good time to use **`pdisk`** to determine the partition numbers for your bootloader and kernel.

Preparing the Open Firmware 3 Bootable Media

The purpose of this section is to prepare the media from which your system will boot the installer. We'll describe how to put the files in the right places on your disk(s) or netboot server and prepare it for use on your system.

If you will be running your system diskless (i.e. entirely over NFS, not using any local hard drives), then you do not need to run the installer, you only need to extract the distribution sets on the diskless server.

To get the distribution sets onto appropriate media, see the above section entitled **Getting the NetBSD System on to Useful Media**. You may want to get the distribution sets when you create the bootable media.

Note: Some MacOS **`ftp`** clients default to downloading files in 'ASCII' mode. This will render the NetBSD files useless. Make sure to set your ftp program to download in 'binary' mode.

What follows are the steps to create different types of bootable media for the NetBSD install kernel. You should only need to create one of these to get your system to boot the installer

- *Creating the NetBSD/macppc CD-R*

  Go to one of the NetBSD mirror sites and download the CD-R image `NetBSD-10.2_STABLE-macppc.iso` from the `pub/NetBSD/images` directory. **http://www.NetBSD.org/mirrors/#iso**

  - *From an UNIX-like machine*

    Get and install cdrecord. NetBSD users should install it from the package collection. Other UNIX-like systems should get it from the official website:
    https://cdn.NetBSD.org/pub/NetBSD/packages/pkgsrc/sysutils/cdrtools/
    http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html

```
# cdrecord -v speed=4 dev=/dev/cd1c NetBSD-10.2_STABLE-macppc.iso
```

You will need to substitute the correct name of the disk image file, speed for your CD writer, and the correct device for your system (for i386 it would be `/dev/cd1d`).

- *From a MacOS machine (using Toast)*

  1. Click the 'Other' button in the main window.
  2. Open the contextual menu on the 'Other' button and select 'Disk Image'
  3. Click the 'Select' button and select the disk image you downloaded.
  4. Click the 'Record' button

Skip forward to **Installing the NetBSD System**

- *Preparing an HFS or HFS+ partition*

Place `ofwboot.xcf` and the installation kernel `netbsd-GENERIC_MD.gz` at the top level of your Macintosh file system. That is, drag the two icons onto your hard drive icon (not the desktop). Make sure they're both on the same partition.

Skip forward to **Installing the NetBSD System**

- *Preparing a custom hybrid HFS/ISO9660 CD-R*

This section describes how to create your own bootable NetBSD/macppc CD-R. We recommend that you use the official NetBSD/macppc–10.2_STABLE CD-R image, as described in the section above.

Place `ofwboot.xcf` and the installation kernel `netbsd-GENERIC_MD.gz` at the top level of the CD. Make sure that the bootloader is present on the HFS partition, and the kernel is on both the ISO9660 and HFS partitions. Due to restrictions in the way that Open Firmware deals with ISO filenames, you may wish to name your kernel `netbsd.gz`. You may also place the NetBSD 10.2_STABLE distribution sets on the ISO9660 partition (not only on the HFS partition).

- *From an UNIX-like machine*

Get and install mkisofs. This is now part of the cdrecord package. NetBSD users should install it from the package collection. Other UNIX-like systems should get it from the official website:

https://cdn.NetBSD.org/pub/NetBSD/packages/pkgsrc/sysutils/cdrtools/
http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html

```
# mkisofs -o NetBSD-10.2_STABLE-macppc.iso -hfs -part -l -J -r -L -N /cdsources
# cdrecord -v speed=4 dev=/dev/cd1c NetBSD-10.2_STABLE-macppc.iso
```

You will need to substitute the correct speed for your CD writer, and the correct device for your system (for i386 it would be `/dev/cd1d`).

See the NetBSD Bootable CD-ROM HOW-TO for more detail:
http://www.NetBSD.org/docs/bootcd.html

- *From a MacOS machine (using Toast)*

  1. Click the 'Data' button in the main window
  2. Pick 'Mac OS and PC (Hybrid) CD' from the popup menu between the four buttons and the file and folder area.
  3. Drag `installation/ofwboot.xcf` and `binary/kernel/netbsd-GENERIC_MD.gz` into the window. You may also want to drag the NetBSD 10.2_STABLE distribution sets to this window as well.

4. Click the 'Record' button

Skip forward to **Installing the NetBSD System**

- *Creating an MS-DOS disk*

  Open Firmware 3 cannot boot from a floppy disk, but you can still create a bootable Zip disk with this method.

  1. First, make sure you have done a low-level format on the disk. A bad disk can produce Open Firmware errors similar to other problems.
  2. Mount the disk on your computer.
     - Insert it into a Windows or DOS machine.
     - Use 'File Exchange or 'PC Exchange'' with MacOS versions prior to X 10.0.0
     - Use `mount_msdos`(8) on any UNIX-like machine.
  3. Copy `netbsd-INSTALL.gz` and `ofwboot.xcf` to the disk. PowerPC 601 machines need to use `netbsd-INSTALL_601.gz`.
  4. Rename `netbsd-INSTALL.gz` to `netbsd.gz`, since OpenFirmware can't deal with MS-DOS filenames longer than eight characters.

  Skip forward to **Installing the NetBSD System**

- *Preparing the netboot server*

  1. Introduction

     To netboot a macppc, you must configure one or more servers to provide information and files to your macppc (the 'client'). If you are using NetBSD (any architecture) on your netboot server(s), the information provided here should be sufficient to configure everything. Additionally, you may wish to look at the `diskless`(8) manual page and the manual pages for each daemon you'll be configuring. If the server(s) are another operating system, you should consult the NetBSD Diskless HOWTO, which will walk you through the steps necessary to configure the netboot services on a variety of platforms:

     https://www.NetBSD.org/docs/network/netboot/

     You may either netboot the installer so you can install onto a locally attached disk, or you may run your system entirely over the network.

     Briefly, the netboot process involves discovery, bootstrap, kernel and file system stages. In the first stage, the client discovers information about where to find the bootstrap program. Next, it downloads and executes the bootstrap program. The bootstrap program goes through another discovery phase to determine where the kernel is located. The bootstrap program tries to mount the NFS share containing the kernel. Once the kernel is loaded, it starts executing. For RAM disk kernels, it mounts the RAM disk file system and begins executing the installer from the RAM disk. For normal (non-RAM disk) kernels, the kernel tries to mount the NFS share that had the kernel and starts executing the installation tools or `init`(8). All macppc systems use BOOTP for the discovery stage. TFTP is used in the bootstrap phase to download the bootstrap program, `ofwboot.xcf`. NFS is used in both the kernel and file system stages to download the kernel, and to access files on the file server.

     We will use '`CC:CC:CC:CC:CC:CC`' as the MAC address (ethernet hardware address) of your netboot client machine. You should have determined this address in an earlier stage. In this example, we will use '`192.168.1.10`' as the IP address of your client and '`client.test.net`' as its name. We will assume you're providing all of your netboot services on one machine called '`server.test.net`' with the client's files exported from the directory `/export/client/root`. You should, of course, replace all of these with the names, addresses, and paths appropriate to your environment.

You should set up each netboot stage in order (i.e., discovery, bootstrap, kernel, and then file system) so that you can test them as you proceed.

2. `dhcpd`(8) in `bootpd`(8) compatible mode

   Put the following lines in your `/etc/dhcpd.conf` (see `dhcpd.conf`(5) and `dhcp-options`(5) for more information):

```
ddns-update-style none;
                # Do not use any dynamic DNS features
                #
allow bootp;    # Allow bootp requests, thus the dhcp server
                # will act as a bootp server.
                #
authoritative;  # master DHCP server for this subnet
                #
subnet 192.168.1.0 netmask 255.255.255.0 {
                # Which network interface to listen on.
                # The zeros indicate the range of addresses
                # that are allowed to connect.
}
group {
                # Set of parameters common to all clients
                # in this "group".
                #
        option broadcast-address        192.168.1.255;
        option domain-name              "test.net";
        option domain-name-servers      dns.test.net;
        option routers                  router.test.net;
        option subnet-mask              255.255.255.0;
                #
                # An individual client.
                #
        host client.test.net {
                hardware ethernet       CC:CC:CC:CC:CC:CC;
                fixed-address           192.168.1.10;
                #
                # Name of the host (if the fixed address
                # doesn't resolve to a simple name).
                #
                option host-name        "client";
                #
                # Name of the bootloader or kernel
                # to download via tftp.
#
                filename                "ofwboot.xcf";

                #
                # The path on the NFS server.
                #
                option root-path "/export/client/root";
```

```
                    #
                    # The host address of the NFS server. This is mandatory for
                    # NetBSD kernels even it's the same host as the DHCP server.
#
                    # macppc machines will look for their bootloader,
                    # ofwboot.xcf, on the next-server as well.
                    #
                    next-server server.test.net;
        }
 #you may paste another "host" entry here for additional
 #clients on this network
}
```

You will need to make sure that the `dhcpd.leases` file exists.

```
# touch /var/db/dhcpd.leases
```

You will need to start the **dhcpd**. If it's already running, you will need to restart it to force it to re-read its configuration file. If the server is running NetBSD, you can achieve this with:

```
# /etc/rc.d/dhcpd restart
```

3. `tftpd`(8)

   The default configuration of the TFTP server is to run in a `chroot`(8) environment in the `/tftpboot` directory. Thus, the first order of business is to create this directory:

   ```
   # mkdir -p /tftpboot
   ```

   Next, edit `/etc/inetd.conf` and uncomment the line with the TFTP daemon:

   ```
   tftp  dgram  udp  wait  root  /usr/libexec/tftpd tftpd -l -s /tftpboot
   ```

   Now, restart `inetd`(8). If the server is running NetBSD, you can achieve this with:

   ```
   # /etc/rc.d/inetd restart
   ```

   Now, you need to copy the bootloader for your macppc machine to `/tftpboot`. Get `ofwboot.xcf`(8) from the `installation` directory of the distribution.

   ```
   # cp ofwboot.xcf /tftpboot
   ```

   Just to be sure, let's make everything readable.

   ```
   # chmod -R a+rX /tftpboot
   ```

   Sometimes, the `arp`(8) table gets messed up, and the TFTP server can't communicate with the client. In this case, it will write a log message (via `syslogd`(8)) to `/var/log/messages` saying: '`tftpd: write: Host is down`'. If this is the case, you may need to force the server to map your client's ethernet address to its IP address:

   ```
   # arp -s client CC:CC:CC:CC:CC:CC
   ```

4. `nfsd`(8), `mountd`(8), and `rpcbind`(8)

   Now your system should be able to load the bootstrap program and start looking for the kernel. Let's set up the NFS server. Create the directory you are exporting for the netboot client:

   ```
   # mkdir -p /export/client/root
   ```

   Put the following line in `/etc/exports` to enable NFS sharing:

```
/export/client/root -maproot=root client.test.net
```

If your server is currently running an NFS server, you only need to restart `mountd`(8). Otherwise, you need to start `rpcbind`(8) and `nfsd`(8). If the server is running NetBSD, you can achieve this with:

```
# /etc/rc.d/rpcbind start
# /etc/rc.d/nfsd start
# /etc/rc.d/mountd restart
```

5. NetBSD kernel and installation tools

   Now, if you place a kernel named `netbsd` in `/export/client/root` your client should boot the kernel. If you are netbooting the installer, use `binary/kernel/netbsd-GENERIC_MD.gz` (this has the installation tools in a RAM disk). Also, copy the distribution files to the client's root directory.

   Note: Some combinations of Open Firmware version and `ofwboot.xcf` version have trouble loading compressed kernels. If you have trouble loading a kernel, try uncompressing it with `gunzip`(1)

   ```
   # cp *.tgz /export/client/root
   # gunzip netbsd-GENERIC_MD.gz
   # mv netbsd-GENERIC_MD /export/client/root/netbsd
   ```

   If you are running your macppc diskless, simply use `binary/kernel/netbsd-GENERIC.gz`.

6. Client file system

   You can skip this step if you do not plan to run your client diskless after installation. Otherwise, you need to extract and set up the client's installation of NetBSD. The Diskless HOW-TO describes how to provide better security and save space on the NFS server over the procedure listed here. See *for details:*

   https://www.NetBSD.org/docs/network/netboot/nfs.html.

   - Extracting distribution sets

     ```
     # cd /export/client/root
     # tar -xpzf /path/to/files/base.tgz
     # tar -xpzf /path/to/files/etc.tgz
     ```

     Continue with the other non-essential distribution sets if desired.

   - Set up swap

     ```
     # mkdir /export/client/root/swap
     # dd if=/dev/zero of=/export/client/swap bs=4k count=4k
     # echo '/export/client/swap -maproot=root:wheel client.test.net' >> /etc/export
     # /etc/rc.d/mountd restart
     ```

     This creates a 16 MB swap file and exports it to the client.

   - Create device nodes

     ```
     # cd /export/client/root/dev
     # ./MAKEDEV all
     ```

     This procedure only works on NetBSD hosts.

   - Set up the client's `fstab`(5)

Create a file in `/export/client/root/etc/fstab` with the following lines:

```
server:/export/client/swap   none  swap  sw,nfsmntpt=/swap
server:/export/client/root   /     nfs   rw 0 0
```

- Set up the client's `rc.conf`(5)

  Edit `/export/client/root/etc/rc.conf`

  ```
  rc_configured=YES
  hostname="client"
  defaultroute="192.168.1.1"
  nfs_client=YES
  auto_ifconfig=NO
  net_interfaces=""
  ```

  Make sure rc does not reconfigure the network device since it will lose its connection to the NFS server with your root file system.

- Set up the client's `hosts`(5) file.

  Edit `/export/client/root/etc/hosts`

  ```
  ::1                     localhost
  127.0.0.1               localhost
  192.168.1.10 client.test.net client
  192.168.1.5  server.test.net server
  ```

7. Setting up the server daemons

   If you want these services to start up every time you boot your server, make sure the following lines are present in your `/etc/rc.conf`:

   ```
   dhcpd=YES        dhcpd_flags="-q"
   nfs_server=YES         # enable server daemons
   mountd=YES
   rpcbind=YES      rpcbind_flags="-l"   # -l logs libwrap
   ```

   Also, you'll need to make sure the `tftpd` line in `/etc/inetd.conf` remains uncommented.

Skip forward to **Installing the NetBSD System**

- *Preparing a SCSI or IDE drive with the CD-R image*

  If the NetBSD/macppc CD-R image is compressed, then you will need to uncompress it first.

  Find a spare bootable drive (i.e. SCSI or IDE), and use some tool to write the disk image `NetBSD-10.2_STABLE-macppc.iso` to your spare drive, and boot from that drive. For example, you could use a Zip drive, a Jaz drive, a Compact Flash drive, or even a spare hard drive. The disk image has an HFS partition with `ofwboot.xcf` which loads the installation kernel from the ISO file system on the disk image.

  - *From an UNIX-like machine (including MacOS X)*

    ```
    # dd if=NetBSD-10.2_STABLE-macppc.iso of=/dev/rsd0c
    ```

    where `/dev/rsd0c` is the 'whole disk' partition for the drive you will be using. Be certain you have typed this correctly, as it will erase the disk.

- *From a Windows machine*

  Get **`rawrite32.exe`** from
  http://www.NetBSD.org/~martin/rawrite32/

  Be certain you have selected the correct disk, as it will erase the contents.

- *From a MacOS 9 (or earlier) machine*

  Get **`suntar`** from
  http://hyperarchive.lcs.mit.edu/HyperArchive/Archive/cmp/suntar-223.hqx

  1. Open the 'Preferences' menu and pick 'Expert Mode'
  2. Open the 'File' menu and pick 'Open Device ...'
  3. Click on the "scan SCSI" button to get a list of which drives are attached.
  4. Pick the correct drive. Be certain you have selected the correct disk, as it will erase the contents.
  5. Open the 'Special' menu and pick 'Overwrite Sectors ...'
  6. Hit return (i.e. start at sector number zero)
  7. Select `NetBSD-10.2_STABLE-macppc.iso` and click "Open"
  8. Open the 'File' menu and pick 'Close Device'

Skip forward to **Installing the NetBSD System**

Preparing your Open Firmware 1.x or 2.x System for NetBSD

Getting to Open Firmware on Apple Network Servers

(Open Firmware 1.1.22)

The version of Open Firmware in the Apple Network Servers can only use a serial console. You must first hook up a serial console (38400 bps, 8 bits, no parity, 1 stop bit, no handshaking) to 'Port 2' (the '`ttya`' device in Open Firmware).

Hold down a special four-key combination on the keyboard attached to the ADB port on your system (not the serial console) when your system boots.

After the chime starts, but before it stops, hold down the four `COMMAND-OPTION-O-F` keys (the `COMMAND` key looks like a four-leaf clover or an open apple, and the `OPTION` key may look like a two-way switch with four straight line segments or say `ALT`) until you see some introductory text and the Open Firmware command prompt on your serial terminal:

```
0 >
```

Your Apple Network Server's screen will remain black.

Now, set your system to always stop at the Open Firmware prompt.

```
0 > setenv auto-boot? false
```

Skip down to the section on **Setting up Open Firmware 1 and 2 to boot NetBSD** since the next several pages are instructions for MacOS models.

Open Firmware 1 and 2 System Preparation

Open Firmware has two variables, 'input-device' and 'output-device', which specify how it accepts commands and displays output. All Open Firmware 1.0.5 and most Open Firmware 2.0.x systems will default to using the 'Modem' serial port for the console instead of the ADB keyboard and the monitor attached to the on-board video.

Unless you use a MacOS-based utility to set these variables correctly, you *will* need to hook up a serial console temporarily to configure Open Firmware to use your keyboard and screen. Some models (such as the Performa 54xx, 6360, 6400, and 6500) have the 'Modem' serial port covered with a piece of plastic since the internal modem usurps that serial port. You will either need to use Boot Variables to set the 'input-device' and 'output-device' variables to 'ttyb' (which is the Printer serial port) or remove the internal modem.

Open Firmware seems to ignore the settings on most DB15 to VGA adapters. Depending on your model, it will default to either 640 x 480 at 60 Hz or to the resolution previously selected in MacOS. Make sure that your monitor can handle these resolutions.

Now would be a good time to look at the NetBSD/macppc Model Support webpage to determine the issues with your model.

http://www.NetBSD.org/ports/macppc/models.html

In particular, some models *must* use a serial console, or they will be unable to boot NetBSD at all. All models can be set to use a serial console, if you desire to bypass the keyboard and screen.

If, after re-reading the next several sections, you still need help figuring out your 'input-device' and 'output-device' settings, see the FAQ:

http://www.NetBSD.org/ports/macppc/faq.html#ofw-input-output-devices

If you need to use a serial console, you can use a normal 'printer' cable (mini-DIN 8 to mini-DIN 8) and a MacOS tool, such as ZTerm to connect a MacOS system to your NetBSD/macppc system.

http://hyperarchive.lcs.mit.edu/HyperArchive/Archive/comm/term/zterm-101.hqx
http://homepage.mac.com/dalverson/zterm/

See the NetBSD Serial Port Primer for additional help and references:

http://www.NetBSD.org/docs/Hardware/Misc/serial.html

All Open Firmware 1 and 2 macppc systems have Open Firmware bugs. Luckily, Open Firmware has a small Non-Volatile RAM variable (NVRAM) which is reserved for FORTH commands which will be run before booting an operating system. Apple has released a freeware MacOS 9 tool called System Disk, which patches most of these bugs. We *strongly* recommend that you use this tool to patch your Open Firmware, as several systems cannot boot without these patches. Instructions for using System Disk are covered in the next section.

Unfortunately, some models are broken by or are unsupported by System Disk. If you have one of the following models, then skip down to the section on **Getting to Open Firmware (Harder, MacOS 7 thru 9)**
Apple Performa 4400, 5500, 6500, 54xx, 6400, and 6360,
Motorola Starmax 3000, 4000, 5000, and 5500,
APS Tech M*Power 604e/200,
PowerComputing PowerBase,
Umax Apus 2000, Apus 3000, C500, and C600
Umax S900

Getting to Open Firmware (MacOS X or Darwin)

When you install MacOS X or Darwin, it will install the necessary NVRAM bug fixes. Use the **nvram** command to set your system to always stop at the Open Firmware prompt, then reboot.

```
# nvram auto-boot\?=false
```

You will need to escape the question-mark or enclose the whole **nvram** argument in double-quotes to prevent your shell from trying to interpret it.

You should now see the Open Firmware command prompt on your screen:

```
Open Firmware, 1.0.5
To continue booting the MacOS type:
BYE<return>
To continue booting from the default boot device type:
BOOT<return>
 ok
0 >
```

If your screen is black, then your model has defaulted to using a serial console. You must hook up a serial console (38400 bps, 8 bits, no parity, 1 stop bit, no handshaking).

Getting to Open Firmware (Best, MacOS 8 or 9)

Download System Disk:
ftp://ftp.apple.com/developer/macosxserver/utilities/SystemDisk2.3.1.smi.bin

For a brief tutorial on how to use System Disk, see:
http://www.NetBSD.org/ports/macppc/SystemDisk-tutorial/

Launch the MacOS System Disk tool. Click on "Power User (Open Firmware)" then click on the "Advanced Options" button. Now, click on the checkbox that says "Stop Boot at Open Firmware prompt" and select "OK". Click the "Save" button and reboot your system.

Note: NVRAM patches and Open Firmware settings will be erased if you 'zap your PRAM' by holding down `COMMAND-OPTION-P-R` keys during the boot chimes, or if you accidentally boot into MacOS 9 or earlier.

If your 'output-device' is '`/chaos/control`' (i.e. you have a PowerMacintosh 7300 — 8600 system), there is a chance that your monitor will not sync. See
http://www.NetBSD.org/ports/macppc/SystemDisk-tutorial/of105patch.html

You should now see the Open Firmware command prompt on your screen:

```
Open Firmware, 1.0.5
To continue booting the MacOS type:
BYE<return>
To continue booting from the default boot device type:
BOOT<return>
 ok
0 >
```

If your screen is black, then your model does not support using the on-board video in Open Firmware. You will need to connect a serial console to the 'Modem' port of your system (38400 bps, 8 bits, no parity, 1 stop bit, no handshaking).

Note: The "Stop Boot at Open Firmware prompt" setting is persistent. It is equivalent to the Open Firmware command

```
0 > setenv auto-boot? false
```

Note: Unfortunately, there are a few models that are better off without the System Disk patches. If you find that your machine doesn't boot, then try:

```
0 > setenv use-nvramrc? false
0 > reset-all
```

Getting to Open Firmware (Harder, MacOS 7, 8, or 9)

If System Disk doesn't work because your version of MacOS is too old or because System Disk says that it doesn't support your model, then you may try using the MacOS BootVars tool.

https://cdn.NetBSD.org/pub/NetBSD/arch/macppc/macos-utils/bootvars/bootvars.sit.hqx

Note: BootVars does *not* apply the (possibly critical) NVRAM patches that System Disk does. Expect some devices to not work (such as booting from hard drives and CD-ROMs).

Look up the proper 'output-device' for your model on the NetBSD/macppc Model Support webpage.
http://www.NetBSD.org/ports/macppc/models.html
If the NetBSD/macppc Model support webpage does not list an 'output-device' for your model, then your system will default to using the on-board video. You needn't fill in the 'output-device' and 'input-device' variables.

Launch the MacOS BootVars tool. Uncheck the "auto-boot?" checkbox, then check on the "All Variables" checkbox and type **kbd** into the 'input-device' box, and the proper device name into the 'output-device' box. Click on the "write" button, and then reboot your system.

If your 'output-device' is '`/chaos/control`' (i.e. you have a PowerMacintosh 7300 — 8600 system), there is a chance that your monitor will not sync. See
http://www.NetBSD.org/ports/macppc/SystemDisk-tutorial/of105patch.html

If you have a Performa 5500 or 6500, you may need to apply NVRAMRC patches to use your built-in video. See the information in the NetBSD/macppc Model Support webpage.

You should now see the Open Firmware command prompt on your screen:

```
Open Firmware, 1.0.5
To continue booting the MacOS type:
BYE<return>
To continue booting from the default boot device type:
BOOT<return>
 ok
0 >
```

If your screen is black, then your model has defaulted to using a serial console. This is fairly common on Open Firmware 1 and 2 models if you do not use the System Disk tool to set up Open Firmware. You must hook up a serial console (38400 bps, 8 bits, no parity, 1 stop bit, no handshaking).

Note: The "auto-boot?" setting is persistent. Your system will always stop at the Open Firmware prompt. It is equivalent to the Open Firmware command

```
0 > setenv auto-boot? false
```

Getting to Open Firmware (Without using MacOS)

(Open Firmware 1.0.5, Open Firmware 2.0.x, Open Firmware 2.4)

If you don't have MacOS, then you need to hold down a special four-key combination when your system boots. Do this on the keyboard attached to the ADB port on your system (not the serial console or PS/2 port) when your system boots.

Note: Your system will *not* have the (possibly critical) NVRAM patches that System Disk applies. Expect some devices to not work (such as booting from hard drives and CD-ROMs).

After the chime starts, but before it stops, hold down the four `COMMAND-OPTION-O-F` keys (the `COMMAND` key looks like a four-leaf clover or an open apple, and the `OPTION` key may look like a two-way switch with four straight line segments or say `ALT`) until you see the Open Firmware command prompt on your screen or serial console:

```
Open Firmware, 1.0.5
To continue booting the MacOS type:
BYE<return>
To continue booting from the default boot device type:
BOOT<return>
 ok
0 >
```

If your screen is black, then your system has defaulted to using a serial console. This is fairly common on Open Firmware 1 and 2 models if you do not use the System Disk tool to set up Open Firmware. You must hook up a serial console (38400 bps, 8 bits, no parity, 1 stop bit, no handshaking).

Now, set your system to always stop at the Open Firmware prompt.

```
0 > setenv auto-boot? false
```

To use your on-board video and keyboard, look up the proper 'output-device' for your model on the NetBSD/macppc webpage
http://www.NetBSD.org/ports/macppc/models.html
Run the following commands to use your screen instead of a serial console (replace '`screen`' with the correct 'output-device' for your model):

```
0 > setenv output-device screen
0 > setenv input-device kbd
0 > reset-all
```

Now you should see the Open Firmware prompt on your screen.

If your 'output-device' is '`/chaos/control`' (i.e. you have a PowerMacintosh 7300 — 8600 system), there is a chance that your monitor will not sync. See
http://www.NetBSD.org/ports/macppc/SystemDisk-tutorial/of105patch.html

Setting up Open Firmware 1 and 2 to boot NetBSD

This section describes some steps you must take to prepare Open Firmware to boot NetBSD. Additional resources are available in the FAQ regarding how to use the Open Firmware command environment, and the Open Firmware variables you may be using:
http://www.NetBSD.org/ports/macppc/faq.html#ofw-access
http://www.NetBSD.org/ports/macppc/faq.html#ofw-variables

Double-check your Open Firmware version:

```
0 > dev /openprom
0 > .properties
name                    openprom
model                   Open Firmware, 1.0.5
relative-addressing
 ok
```

If your system has Open Firmware prior to version 3, then you must set some Open Firmware variables before NetBSD can boot. Do *not* run these commands on Open Firmware 3 machines, as you may overwrite your firmware requiring a trip to Apple for repairs.

```
0 > setenv load-base 600000
0 > setenv real-base F00000
0 > reset-all
```

The last command reboots your machine so that the settings are stored.

If you will be netbooting your system, you can look up your MAC address.

```
0 > dev enet
0 > .properties
[...]
local-mac-address   CCCCCCCC CCCC
[...]
 ok
```

Note: Some early Open Firmware 1.0.5 machines had their MAC address stored incorrectly on the motherboard (little- vs. big-endian problems). The patches the System Disk installs will correct this. Without the patch, the machine will still work, but its MAC address may conflict with another ethernet device on your network.

You can check your Open Firmware settings with the `printenv` command:

```
0 > printenv
little-endian?      false               false
real-mode?          false               false
auto-boot?          false               true
diag-switch?        false               false
[...]
use-nvramrc?        true                false
real-base           F00000              -1
[...]
load-base           600000              4000
[...]
input-device        kbd                 ttya
output-device       /chaos/control      ttya
```

Note: All Open Firmware 1.0.5 settings and nvram patches will be erased if you boot into MacOS 9 or earlier. You will need to re-enter them before booting NetBSD again.

Note: Open Firmware 2.0.x and Open Firmware 2.4 systems will set the `real-base` environment variable to its default value (which prevents NetBSD from booting) if you boot into MacOS 9 or earlier.

Available Boot Media

Open Firmware is capable of booting from a variety of media (such as hard drives, floppy disks, CD-ROMs, and ethernet). Open Firmware is able to boot files from a variety of file systems (such as ISO9660, and MS-DOS FAT). Unfortunately, Open Firmware is *not* able to directly boot from the NetBSD file system (FFS) or Apple's file systems (HFS, HFS+, or UFS), so we must put the bootloader in a location that Open Firmware is capable of understanding.

Therefore, to boot the NetBSD kernel, Open Firmware must first load a 'bootloader' which knows how to load the NetBSD kernel. Open Firmware 1 and 2 take either a two or three stage approach, depending on the boot media. In the two step approach, Open Firmware loads `ofwboot.xcf` from the boot media, which then loads the kernel. In the three step approach (used in the 'partition zero' method), Open Firmware loads a primary bootloader `bootxx` from a disk which then loads the secondary bootloader `ofwboot` (which is functionally identical to `ofwboot.xcf`) which then loads the kernel.

Note: Despite a note to the contrary in earlier releases that `ofwboot.elf` is obsoleted, that is no longer the case. It has been observed that on some newish machines (notably at least some Mac Minis), use of `ofwboot.xcf` will result in an early kernel panic when loading the GENERIC kernel from disk. For these machines it appears to be a workaround to use `ofwboot.elf` instead of `ofwboot.xcf` as the boot loader. The root cause for this problem has unfortunately yet to be found.

The following bootable media are available for loading the bootloader:

- '*Partition zero*'

  This method loads the primary bootloader `bootxx` located in the Apple Partition Map, which then loads the secondary bootloader `ofwboot` which then loads the kernel. This is what **sysinst** will place on your hard drive during the installation procedure. Additionally, the NetBSD/macppc CD-R images and boot floppies use this method, loading a kernel from the floppy's FFS partition or the CD-R's ISO9660 partition.

- *MS-DOS file system*

  In this method, Open Firmware loads the `ofwboot.xcf` bootloader from an MS-DOS file system. It may then load a NetBSD kernel from the same MS-DOS file system. This has only been thoroughly tested on floppy disks, but may work on Zip disks or FDISK-formatted hard drives. This does not work for MS-DOS partitions on a hard drive with an Apple Partition Map.

- *Ethernet (network boot)*

  You can run your entire system diskless or netboot only the files necessary to boot (i.e. the bootloader and the installation kernel). You must have root access for the UNIX-like netboot server, which must be on the same subnet as your NetBSD/macppc machine.

  Note: You must use your on-board ethernet device for netbooting. While there may be some PCI or Cardbus ethernet interfaces with Open Firmware ROMs, no one has reported success netbooting using these devices

- *ISO9660 CD-ROM file system*

  Load the `ofwboot.xcf` bootloader from an ISO9660 CD-ROM. It can then load a NetBSD kernel from the ISO9660 CD-ROM.

  Open Firmware 1.0.5 and 2.0.x machines will *not* boot from the ISO9660 file system if you create a hybrid HFS/ISO format CD-R. Make sure that the CD *only* has ISO9660 data and does not have an Apple Partition Map.

Note: Open Firmware can only open files on the first session of a multi-session CD-R

Once the bootloader is loaded, it can open the kernel from one of the following sources:

- *Ethernet (NFS)*
- *NetBSD FFS file system*

  Such as the boot floppy, or an installed NetBSD partition.
- *MacOS X UFS file system*
- *ISO9660 CD-ROM file system*
- *MS-DOS file system*

In theory, Open Firmware 2.4 systems should be able to load the bootloader from an HFS or HFS+ partition or a hybrid CD-R, but users have reported that it doesn't work.

The boot floppy images provided have a 'partition zero' bootloader and a NetBSD file system with an installation kernel. The boot CD image provided has both a 'partition zero' bootloader and `ofwboot.xcf` on a hybrid partition so it can be booted on all Open Firmware versions. It also has an ISO9660 file system with an installation kernel and the distribution sets.

Partitioning your hard drive for NetBSD

You must use the NetBSD installer to partition your disk if you want it to be bootable. With this release of NetBSD, there is no way to dual-boot MacOS and NetBSD on one hard drive.

You can use the instructions in this section to partition a disk that may also be used with MacOS, although a disk prepared in this way *will not boot* NetBSD. That means, your root partition (`/`) must be on a drive prepared with the NetBSD installer, but the partitions not necessary to boot (for example `/usr`, `/home`, or `/export`) may be on the same disk as MacOS.

Unless you are planning to use partitions on the same disk as MacOS, skip forward to **Preparing the Open Firmware 1 or 2 Bootable Media**

There are two partitioning tools available for NetBSD/macppc, `disklabel`(8) and `pdisk`(8). The former is used in the NetBSD **`sysinst`** installer, and will render your disk unusable by MacOS.

The process is more fully detailed in the Partitioning HOW-TO:
http://www.NetBSD.org/ports/macppc/partitioning.html

You can create a partition map with `pdisk`(8), but the disk will not be usable with MacOS 9 and earlier. If this is a concern, you will need to use Apple's Drive Setup or Disk Utility.

If you are using Apple's Drive Setup tool, make sure you have version 1.8.1 or later. This tool only runs under MacOS 9 and earlier. Drive Setup will erase the contents of your drive — it does not preserve data from any of your partitions.

Apple's Disk Utility only runs under MacOS X 10.0.0 and later. Make sure you click the "Install Mac OS 9 Disk Drivers" checkbox. Also, keep in mind that Disk Utility does *not* create the partitions that NetBSD/macppc requires. After creating the initial partition map with Disk Utility, you will need to use the NetBSD **`pdisk`** to change the partition types. Also, Disk Utility will erase the contents of your drive — it does not preserve data from any of your partitions.

`pdisk` is the most flexible (and most difficult to use) partitioning tool available. It runs on almost all OSes that macppc machines support. Download it:
https://cdn.NetBSD.org/pub/NetBSD/arch/macppc/netbsd-pdisk/
https://cdn.NetBSD.org/pub/NetBSD/arch/macppc/macos-utils/pdisk.sea.hqx

There is built-in help describing how it works. When it asks you to enter the "`Type of partition`", use `Apple_UNIX_SVR2` for NetBSD partitions, `Apple_HFS` for HFS and HFS+ partitions, and `Apple_UFS` for UFS partitions.

After you've written the partition map with **`pdisk`**, you will need to create the file systems. Use `newfs`(8) and `mount`(8) for NetBSD file systems, and **`hfstools`** to create and mount HFS file systems.

https://cdn.NetBSD.org/pub/NetBSD/packages/pkgsrc/sysutils/hfsutils/

Make the following partitions:

A/UX Swap	Any size. The recommendation is your RAM size, although this is not strictly necessary for machines with a lot of RAM. NetBSD interprets an *A/UX Swap* partition as the second partition (*b*) on the disk. This partition is not readable from MacOS.
A/UX User and *A/UX Free1*	Use these for any additional partitions you may want to use under NetBSD, such as `/usr` (at least 200 MB), `/home`, `/usr/local`, or `/usr/pkg`. NetBSD interprets these partitions as normal NetBSD-style partitions. These partitions are not readable from MacOS.
HFS	Any size. You may want to leave an additional partition available to transfer files between MacOS and NetBSD. If would like to create such a partition, then see the Partitioning HOW-TO. **http://www.NetBSD.org/ports/macppc/partitioning.html#msdos**
UFS	Any size. *UFS* partitions are not readable from MacOS versions prior to X 10.0.0. If you use an *UFS* partition as your root, then it may not be recognized by the NetBSD kernel as the first partition (*a*) on the disk. You will need to compile a new kernel with the root partition explicitly defined to be the *UFS* partition.

Now would be a good time to use **`pdisk`** to determine the partition numbers for your bootloader and kernel.

Preparing the Open Firmware 1 or 2 Bootable Media

The purpose of this section is to prepare the media from which your system will boot the installer. We'll describe how to put the files in the right places on your disk(s) or netboot server and prepare it for use on your system.

If you will be running your system diskless (i.e. entirely over NFS, not using any local hard drives), then you do not need to run the installer, you only need to extract the distribution sets on the diskless server.

To get the distribution sets onto appropriate media, see the above section entitled **Getting the NetBSD System on to Useful Media**. You may want to get the distribution sets when you create the bootable media.

Note: Some MacOS **`ftp`** clients default to downloading files in 'ASCII' mode. This will render the NetBSD files useless. Make sure to set your ftp program to download in 'binary' mode.

What follows are the steps to create different types of bootable media for the NetBSD install kernel. You should only need to create one of these to get your system to boot the installer

- *Creating the NetBSD/macppc CD-R*

  Go to one of the NetBSD mirror sites and download the CD-R image `NetBSD-10.2_STABLE-macppc.iso` from the `pub/NetBSD/images` directory. **http://www.NetBSD.org/mirrors/#iso**

- *From an UNIX-like machine*

  Get and install cdrecord. NetBSD users should install it from the package collection. Other UNIX-like systems should get it from the official website:

  https://cdn.NetBSD.org/pub/NetBSD/packages/pkgsrc/sysutils/cdrtools/
  http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html

  ```
  # cdrecord -v speed=4 dev=/dev/cd1c NetBSD-10.2_STABLE-macppc.iso
  ```

  You will need to substitute the correct name of the disk image file, speed for your CD writer, and the correct device for your system (for i386 it would be `/dev/cd1d`).

- *From a MacOS machine (using Toast)*

  1. Click the 'Other' button in the main window.
  2. Open the contextual menu on the 'Other' button and select 'Disk Image'
  3. Click the 'Select' button and select the disk image you downloaded.
  4. Click the 'Record' button

Skip forward to **Installing the NetBSD System**

- *Creating the NetBSD/macppc install floppies*

  1. First, make sure you have done a low-level format on the disks. A bad floppy can produce Open Firmware errors similar to other problems.
  2. If the boot floppy images are compressed, then you will need to uncompress them first.
  3. Use a tool to write the boot floppy images `installation/floppy/boot1.fs` and `installation/floppy/boot2.fs` to two floppy disks. PowerPC 601 machines should use `installation/floppy/boot601_1.fs` and `installation/floppy/boot601_2.fs` instead.

     - *From an UNIX-like machine*

       ```
       # dd if=boot1.fs of=/dev/rfd0a bs=36k
       # dd if=boot2.fs of=/dev/rfd0a bs=36k
       ```

     - *From a Windows machine*

       Get either the **rawrite.exe** program from the NetBSD/i386 distribution or **rawrite32.exe** from
       http://www.NetBSD.org/~martin/rawrite32/

     - *From a MacOS 9 (or earlier) machine*

       Get **suntar** from
       http://hyperarchive.lcs.mit.edu/HyperArchive/Archive/cmp/suntar-223.hqx

       1. Open the 'Preferences' menu and pick 'Expert Mode'
       2. Open the 'Special' menu and pick 'Overwrite Sectors ...'
       3. Insert the first floppy disk
       4. Hit return (i.e. start at sector number zero)
       5. Select `boot1.fs` and click "Open"

6. Open the 'File' menu and pick 'Eject'
7. Repeat the process for the second floppy

Skip forward to **Installing the NetBSD System**

- *Creating a custom ISO9660 CD-R*

  This section describes how to create your own bootable NetBSD/macppc CD-R. We recommend that you use the official NetBSD/macppc−10.2_STABLE CD-R image, as described in the section above.

  Place `ofwboot.xcf` and the installation kernel `netbsd-GENERIC_MD.gz` at the top level of the CD. Due to restrictions in the way that Open Firmware deals with ISO filenames, you may wish to name your kernel `netbsd.gz`. You may also place the NetBSD 10.2_STABLE distribution sets on the disk.

  – *From an UNIX-like machine*

  Get and install mkisofs. This is now part of the cdrecord package. NetBSD users should install it from the package collection. Other UNIX-like systems should get it from the official website:
  https://cdn.NetBSD.org/pub/NetBSD/packages/pkgsrc/sysutils/cdrtools/
  http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html

```
# mkisofs -o NetBSD-10.2_STABLE-macppc.iso -l -J -r -L -N /cdsources
# cdrecord -v speed=4 dev=/dev/cd1c NetBSD-10.2_STABLE-macppc.iso
```

  You will need to substitute the correct speed for your CD writer, and the correct device for your system (for i386 it would be `/dev/cd1d`).

  See the NetBSD Bootable CD-ROM HOW-TO for more detail:
  http://www.NetBSD.org/docs/bootcd.html

  – *From a MacOS machine (using Toast)*

  1. Click the 'Other' button in the main window
  2. Open the contextual menu on the 'Other' button and select 'ISO 9660'
  3. Click the 'Select' button.
  4. Click the 'Settings' tab, open the 'Naming' popup menu, and pick 'Allow Macintosh Names'
  5. Click on the 'Files' tab
  6. Drag `installation/ofwboot.xcf` and `binary/kernel/netbsd-GENERIC_MD.gz` into the window. You may also want to drag the NetBSD 10.2_STABLE distribution sets to this window as well.
  7. Click the 'Done' button
  8. Click the 'Record' button

  Skip forward to **Installing the NetBSD System**

- *Creating an MS-DOS disk*

  Unfortunately, the installation kernel is now too large to fit onto a floppy disk, but you can still create a bootable Zip disk with this method.

  1. First, make sure you have done a low-level format on the disk. A bad disk can produce Open Firmware errors similar to other problems.
  2. Mount the disk on your computer.

  - Insert it into a Windows or DOS machine.
  - Use 'File Exchange or 'PC Exchange'' with MacOS versions prior to X 10.0.0
  - Use `mount_msdos`(8) on any UNIX-like machine.

3. Copy `netbsd-INSTALL.gz` and `ofwboot.xcf` to the disk. PowerPC 601 machines need to use `netbsd-INSTALL_601.gz`.
4. Rename `netbsd-INSTALL.gz` to `netbsd.gz`, since OpenFirmware can't deal with MS-DOS filenames longer than eight characters.

Skip forward to **Installing the NetBSD System**

- *Preparing the netboot server*

1. Introduction

   To netboot a macppc, you must configure one or more servers to provide information and files to your macppc (the 'client'). If you are using NetBSD (any architecture) on your netboot server(s), the information provided here should be sufficient to configure everything. Additionally, you may wish to look at the `diskless`(8) manual page and the manual pages for each daemon you'll be configuring. If the server(s) are another operating system, you should consult the NetBSD Diskless HOWTO, which will walk you through the steps necessary to configure the netboot services on a variety of platforms:
   https://www.NetBSD.org/docs/network/netboot/

   You may either netboot the installer so you can install onto a locally attached disk, or you may run your system entirely over the network.

   Briefly, the netboot process involves discovery, bootstrap, kernel and file system stages. In the first stage, the client discovers information about where to find the bootstrap program. Next, it downloads and executes the bootstrap program. The bootstrap program goes through another discovery phase to determine where the kernel is located. The bootstrap program tries to mount the NFS share containing the kernel. Once the kernel is loaded, it starts executing. For RAM disk kernels, it mounts the RAM disk file system and begins executing the installer from the RAM disk. For normal (non-RAM disk) kernels, the kernel tries to mount the NFS share that had the kernel and starts executing the installation tools or `init`(8). All macppc systems use BOOTP for the discovery stage. TFTP is used in the bootstrap phase to download the bootstrap program, `ofwboot.xcf`. NFS is used in both the kernel and file system stages to download the kernel, and to access files on the file server.

   We will use '`CC:CC:CC:CC:CC:CC`' as the MAC address (ethernet hardware address) of your netboot client machine. You should have determined this address in an earlier stage. In this example, we will use '`192.168.1.10`' as the IP address of your client and '`client.test.net`' as its name. We will assume you're providing all of your netboot services on one machine called '`server.test.net`' with the client's files exported from the directory `/export/client/root`. You should, of course, replace all of these with the names, addresses, and paths appropriate to your environment.

   You should set up each netboot stage in order (i.e., discovery, bootstrap, kernel, and then file system) so that you can test them as you proceed.

2. `dhcpd`(8) in `bootpd`(8) compatible mode

   Put the following lines in your `/etc/dhcpd.conf` (see `dhcpd.conf`(5) and `dhcp-options`(5) for more information):

```
ddns-update-style none;
                 # Do not use any dynamic DNS features
```

```
                #
allow bootp;    # Allow bootp requests, thus the dhcp server
                # will act as a bootp server.
                #
authoritative;  # master DHCP server for this subnet
                #
subnet 192.168.1.0 netmask 255.255.255.0 {
                # Which network interface to listen on.
                # The zeros indicate the range of addresses
                # that are allowed to connect.
}
group {
                # Set of parameters common to all clients
                # in this "group".
                #
        option broadcast-address        192.168.1.255;
        option domain-name              "test.net";
        option domain-name-servers      dns.test.net;
        option routers                  router.test.net;
        option subnet-mask              255.255.255.0;
                #
                # An individual client.
                #
        host client.test.net {
                hardware ethernet       CC:CC:CC:CC:CC:CC;
                fixed-address           192.168.1.10;
                #
                # Name of the host (if the fixed address
                # doesn’t resolve to a simple name).
                #
                option host-name        "client";
                #
                # Name of the bootloader or kernel
                # to download via tftp.
#
                filename                "ofwboot.xcf";

                #
                # The path on the NFS server.
                #
                option root-path "/export/client/root";

                #
                # The host address of the NFS server. This is mandatory for
                # NetBSD kernels even it’s the same host as the DHCP server.
#
                # macppc machines will look for their bootloader,
                # ofwboot.xcf, on the next-server as well.
                #
                next-server server.test.net;
        }
 #you may paste another "host" entry here for additional
```

```
 #clients on this network
}
```

You will need to make sure that the `dhcpd.leases` file exists.

```
# touch /var/db/dhcpd.leases
```

You will need to start the **dhcpd**. If it's already running, you will need to restart it to force it to re-read its configuration file. If the server is running NetBSD, you can achieve this with:

```
# /etc/rc.d/dhcpd restart
```

3. `tftpd`(8)

   The default configuration of the TFTP server is to run in a `chroot`(8) environment in the `/tftpboot` directory. Thus, the first order of business is to create this directory:

   ```
   # mkdir -p /tftpboot
   ```

   Next, edit `/etc/inetd.conf` and uncomment the line with the TFTP daemon:

   ```
   tftp   dgram   udp   wait   root   /usr/libexec/tftpd tftpd -l -s /tftpboot
   ```

   Now, restart `inetd`(8). If the server is running NetBSD, you can achieve this with:

   ```
   # /etc/rc.d/inetd restart
   ```

   Now, you need to copy the bootloader for your macppc machine to `/tftpboot`. Get `ofwboot.xcf`(8) from the `installation` directory of the distribution.

   ```
   # cp ofwboot.xcf /tftpboot
   ```

   Just to be sure, let's make everything readable.

   ```
   # chmod -R a+rX /tftpboot
   ```

   Sometimes, the `arp`(8) table gets messed up, and the TFTP server can't communicate with the client. In this case, it will write a log message (via `syslogd`(8)) to `/var/log/messages` saying: '`tftpd: write: Host is down`'. If this is the case, you may need to force the server to map your client's ethernet address to its IP address:

   ```
   # arp -s client CC:CC:CC:CC:CC:CC
   ```

4. `nfsd`(8), `mountd`(8), and `rpcbind`(8)

   Now your system should be able to load the bootstrap program and start looking for the kernel. Let's set up the NFS server. Create the directory you are exporting for the netboot client:

   ```
   # mkdir -p /export/client/root
   ```

   Put the following line in `/etc/exports` to enable NFS sharing:

   /export/client/root -maproot=root client.test.net

   If your server is currently running an NFS server, you only need to restart `mountd`(8). Otherwise, you need to start `rpcbind`(8) and `nfsd`(8). If the server is running NetBSD, you can achieve this with:

   ```
   # /etc/rc.d/rpcbind start
   # /etc/rc.d/nfsd start
   # /etc/rc.d/mountd restart
   ```

5. NetBSD kernel and installation tools

   Now, if you place a kernel named `netbsd` in `/export/client/root` your client should boot the kernel. If you are netbooting the installer, use `binary/kernel/netbsd-GENERIC_MD.gz` (this has the installation tools in a RAM disk). Also, copy the distribution files to the client's root directory.

   Note: Some combinations of Open Firmware version and `ofwboot.xcf` version have trouble loading compressed kernels. If you have trouble loading a kernel, try uncompressing it with `gunzip`(1)

   ```
   # cp *.tgz /export/client/root
   # gunzip netbsd-GENERIC_MD.gz
   # mv netbsd-GENERIC_MD /export/client/root/netbsd
   ```

   If you are running your macppc diskless, simply use `binary/kernel/netbsd-GENERIC.gz`.

6. Client file system

   You can skip this step if you do not plan to run your client diskless after installation. Otherwise, you need to extract and set up the client's installation of NetBSD. The Diskless HOW-TO describes how to provide better security and save space on the NFS server over the procedure listed here. See *for details:*

   https://www.NetBSD.org/docs/network/netboot/nfs.html.

   - Extracting distribution sets

     ```
     # cd /export/client/root
     # tar -xpzf /path/to/files/base.tgz
     # tar -xpzf /path/to/files/etc.tgz
     ```

     Continue with the other non-essential distribution sets if desired.

   - Set up swap

     ```
     # mkdir /export/client/root/swap
     # dd if=/dev/zero of=/export/client/swap bs=4k count=4k
     # echo '/export/client/swap -maproot=root:wheel client.test.net' >> /etc/export
     # /etc/rc.d/mountd restart
     ```

     This creates a 16 MB swap file and exports it to the client.

   - Create device nodes

     ```
     # cd /export/client/root/dev
     # ./MAKEDEV all
     ```

     This procedure only works on NetBSD hosts.

   - Set up the client's `fstab`(5)

     Create a file in `/export/client/root/etc/fstab` with the following lines:

     ```
     server:/export/client/swap   none  swap  sw,nfsmntpt=/swap
     server:/export/client/root   /     nfs   rw 0 0
     ```

   - Set up the client's `rc.conf`(5)

     Edit `/export/client/root/etc/rc.conf`

     ```
     rc_configured=YES
     hostname="client"
     defaultroute="192.168.1.1"
     ```

```
nfs_client=YES
auto_ifconfig=NO
net_interfaces=""
```

Make sure rc does not reconfigure the network device since it will lose its connection to the NFS server with your root file system.

- Set up the client's `hosts`(5) file.

  Edit `/export/client/root/etc/hosts`

  ```
  ::1                     localhost
  127.0.0.1               localhost
  192.168.1.10 client.test.net client
  192.168.1.5  server.test.net server
  ```

7. Setting up the server daemons

   If you want these services to start up every time you boot your server, make sure the following lines are present in your `/etc/rc.conf`:

   ```
   dhcpd=YES         dhcpd_flags="-q"
   nfs_server=YES          # enable server daemons
   mountd=YES
   rpcbind=YES       rpcbind_flags="-l"   # -l logs libwrap
   ```

   Also, you'll need to make sure the `tftpd` line in `/etc/inetd.conf` remains uncommented.

Skip forward to **Installing the NetBSD System**

- *Preparing a SCSI or IDE drive with the CD-R image*

  If the NetBSD/macppc CD-R image is compressed, then you will need to uncompress it first.

  Find a spare bootable drive (i.e. SCSI or IDE), and use some tool to write the disk image `NetBSD-10.2_STABLE-macppc.iso` to your spare drive, and boot from that drive. For example, you could use a Zip drive, a Jaz drive, a Compact Flash drive, or even a spare hard drive. The disk image has a 'partition zero' bootloader which ultimately loads the installation kernel from the ISO file system on the disk image.

  - *From an UNIX-like machine (including MacOS X)*

    ```
    # dd if=NetBSD-10.2_STABLE-macppc.iso of=/dev/rsd0c
    ```

    where `/dev/rsd0c` is the 'whole disk' partition for the drive you will be using. Be certain you have typed this correctly, as it will erase the disk.

  - *From a Windows machine*

    Get **`rawrite32.exe`** from
    http://www.NetBSD.org/~martin/rawrite32/

    Be certain you have selected the correct disk, as it will erase the contents.

  - *From a MacOS 9 (or earlier) machine*

    Get **`suntar`** from
    http://hyperarchive.lcs.mit.edu/HyperArchive/Archive/cmp/suntar-223.hqx

    1. Open the 'Preferences' menu and pick 'Expert Mode'

2. Open the 'File' menu and pick 'Open Device ...'
3. Click on the "scan SCSI" button to get a list of which drives are attached.
4. Pick the correct drive. Be certain you have selected the correct disk, as it will erase the contents.
5. Open the 'Special' menu and pick 'Overwrite Sectors ...'
6. Hit return (i.e. start at sector number zero)
7. Select `NetBSD-10.2_STABLE-macppc.iso` and click "Open"
8. Open the 'File' menu and pick 'Close Device'

Installing the NetBSD System

Open Firmware boot syntax

The syntax of the Open Firmware **boot** command is:

```
boot boot-device [boot-file] [-as]
```

where the `boot-device` describes where to find the bootloader, `boot-file` describes where to find the NetBSD kernel, and the options specify how you want to boot.

You use the `boot-device` to tell Open Firmware where to find `ofwboot`(8) by listing the device, the partition (if it's a disk), and the filename of the bootloader (if using `ofwboot.xcf`).

If the `boot-file` is on the same device and partition (if it's a disk) as the `boot-device` then you can just specify the kernel filename. Otherwise, you need to specify the full Open Firmware path to the kernel.

The **-a** flag will ask you for the location of the next item to load (i.e. the bootloader will ask where the kernel is (if unspecified), or the kernel will ask where the root file system is). The **-s** flag will boot into 'single-user' mode.

The exact command you will be using depends on which version of Open Firmware your machine has and which device you will be booting from. Sometimes you may have to guess as we don't know all of the combinations of models, device names, and file names. In general the format is: `device:[partition][,\filename]`. Keep in mind for the future that you may be able to have your `boot-device` and `boot-file` on entirely different devices (such as the bootloader netbooted from `enet` and the kernel loaded from a hard drive on the `ultra0` ATA/IDE bus).

We'll try to walk you through the process of figuring out what Open Firmware calls your device, partition, and file names. To start with, Open Firmware keeps a "device tree" with all of the devices it finds in your system. You can get a listing of the nodes in this device tree with the **dev** and **ls** commands. **dev** is similar to the unix **cd** command and is used to change between the nodes in the Open Firmware device tree (similar to a file system). **ls** of course is similar to the unix **ls** command and is used to list the contents of the current device node. To get a listing of all the devices available in your system, use the following commands:

```
0 > dev /
0 > ls
```

Open Firmware has device aliases which are simple names for the full hardware path to a device (similar to **alias** in `csh`(1)). You can find out what device aliases Apple created on your machine with the **devalias** command. For example, here are the devaliases on a PowerMacintosh 7300:

```
0 > devalias
vci0                /chaos@F0000000
pci1                /bandit@F2000000
pci2                /bandit@F4000000
```

```
fd                    /bandit/gc/swim3
kbd                   /bandit/gc/via-cuda/adb/keyboard
ttya                  /bandit/gc/escc/ch-a
ttyb                  /bandit/gc/escc/ch-b
enet                  /bandit/gc/mace
scsi                  /bandit/gc/53c94
scsi-int              /bandit/gc/mesh
 ok
```

On most systems, you'll find the devices you're looking for. Typical aliases are:

`hd`	internal hard drive
`cd`	CD-ROM drive
`zip`	internal Zip drive
`enet`	ethernet
`fd`	floppy drive
`scsi`	SCSI bus
`scsi-int`	internal SCSI bus (on systems with multiple SCSI busses)
`ata`	ATA/IDE bus
`ideN`	ATA/IDE bus number N
`ultraN`	Ultra/66 or Ultra/100 IDE bus number N

Note that some of these items are the device itself, and some are a bus. When you only have the devalias to a bus, you need to specify which device on that bus you want to use. You can use the Open Firmware **`dev`** and **`ls`** commands. For example, here are the devices on the internal SCSI bus of a PowerMacintosh 7300:

```
0 > dev scsi-int
0 > ls
FF83C850: /sd@0,0
FF83D480: /st@0,0
 ok
```

In this case, Open Firmware seems to be saying there are two devices, both at address zero (one is a SCSI disk 'sd@0,0' and the other is a SCSI tape 'st@0,0).' Unfortunately, older systems will only list the naming convention and not the actual devices currently connected, but that's OK -- we've got more tricks up our sleeve.

If you've got ATA/IDE drives, you have all the device information you need (since Apple only ever ships drives as "master" which is typically something like `ata-disk@0, ATA-Disk@0, atapi-disk, or disk@0 ).`

You can find out the devices on your SCSI bus with the **`show-children`** command:

```
0 > dev scsi-int
0 > show-children
Target 0
  Unit 0  Disk     IBM     DCAS-32160      S65A
Target 3
  Unit 0  Removable Read Only device    SONY    CD-ROM CDU-8005 1.0j
 ok
```

Open Firmware calls SCSI IDs "`Target`". The "`Unit`" number is the Logical Unit Number (LUN). This is almost always zero. Thus, this PowerMacintosh system has an IBM hard drive (DCAS-32160) at SCSI ID 0, and a Sony CD-ROM drive (CDU-8005) at SCSI ID 3.

Now, we've got enough information to construct the device name for Open Firmware. Just stick everything together to describe to Open Firmware what you want. For example, Open Firmware calls the CD-ROM drive in this system `scsi-int/sd@3`.

To determine if a device is bootable, type:

```
0 > dev scsi-int/sd@3
0 > words
load          write         read          seek          close      open
write-blocks  read-blocks   max-transfer  block-size    dma-sync   dma-map-out
dma-map-in    dma-free      dma-alloc
 ok
```

If the **word** "`open`" is present in the list, then the device is almost certainly bootable.

Next, you need to figure out what partition Open Firmware thinks your bootloader is located on if you're going to boot from a disk. If you're using a "partition zero" bootloader, the answer is obvious: 0. Thus, your `boot-device` for the NetBSD/macppc CD-R image on an Open Firmware 1.0.5 system would be `scsi-int/sd@3:0` since the image has a 'partition zero' bootloader.

Other situations get a little trickier, as we know of no way to get a partition map from within Open Firmware, and it uses a different numbering scheme than either NetBSD or MacOS 9 (or earlier). You can use **pdisk** to get a listing of the partitions on a disk. See the Partitioning HOW-TO for help:
http://www.NetBSD.org/ports/macppc/partitioning.html#pdisk

Typically, MS-DOS and ISO9660 formatted disks have their file systems at partition 1. Typically, Drive Setup formatted disks have their file systems starting at partition number 9. Often, if you omit the partition number, Open Firmware looks in the first partition it understands holding a valid file system.

Open Firmware uses a comma (,) to separate the partition number from the filename. It uses a backslash (the \ character) to separate directories. The bootloader uses forward slashes (the / character) to separate directories when specifying the `boot-file`. Thus, to specify the top of the file system on a CD-ROM in the example PowerMacintosh 7300 system, you'd use: `scsi-int/sd@3:,\`

Now, to confirm that you and Open Firmware are looking at the same files, you can get a directory listing of the file system on your device with the **dir** command. This command is only useful with file systems that Open Firmware understands and is able to boot from. On Open Firmware 1.0.5, 1.1.22, and 2.0.x systems, you can use **dir** on ISO9660 (not hybrid) and MS-DOS file systems. On Open Firmware 2.4, you can use it on HFS, HFS+, hybrid, ISO9960, and MS-DOS file systems. On Open Firmware 3, you can use it on HFS, HFS+, hybrid (not pure ISO9660), and MS-DOS file systems. The one exception to this rule is that Open Firmware cannot list files on a disk with a 'partition zero' bootloader (including the NetBSD/macppc CD-R image and installation floppies).

```
0 > dir fd:,\
FINDER  .DAT 022 2 2B8
DESKTOP .    022 0 0
RESOURCE.FRK 012 3 0
NETBSD~1.GZ  020 5 1FDFCA
TRASH   .    010 B00 0
OFWBOOT .XCF 020 A75 D8F4
 ok
```

You can see that this MS-DOS formatted disk has a bunch of stuff, as well as the two important files: `NETBSD~1.GZ` and `OFWBOOT.XCF`. Note that MacOS shortened `netbsd-GENERIC_MD.gz` to `NETBSD~1.GZ` since MS-DOS file systems can only natively hold 8 characters for the filename.

Note: Keep in mind that Open Firmware is often case-sensitive when it comes to filenames.

Note: You may need to append a `;1` to the filename when using a ISO 9660 file system. This "version number" is part of the ISO 9660 specification and will show up in the directory listing if it is present on the disk. For example:

```
0 > boot cd:,\OFWBOOT.XCF;1 NETBSD.MACPPC;1
```

Note: PowerPC 601 machines need to specify the `NETBSD.601` kernel when booting from the install CD, and use the `kern-GENERIC_601.tgz` and `netbsd-INSTALL_601.gz` kernels.

If the **dir** command showed you the files you're looking for, then you've figure out how to tell Open Firmware to look for your bootloader! In this case, your '`boot-device`' is '`fd:,\OFWBOOT.XCF`' and your '`boot-file`' is '`fd:,/NETBSD~1.GZ`'.

For additional help, see `ofwboot`(8) and the FAQ on topics like how to use the Open Firmware command environment and how to boot from a device attached to a PCI card which has Open Firmware support:
http://www.NetBSD.org/ports/macppc/faq.html#ofw-use
http://www.NetBSD.org/ports/macppc/faq.html#boot-pci

Examples of Open Firmware boot commands

Here are some examples of the commands you might use to boot your system:

- *Booting the NetBSD/macppc install CD-R*
  (Open Firmware 1.0.5, Open Firmware 1.1.22, Open Firmware 2.0.x, Open Firmware 2.4)

  Here are some examples of what you might use to boot from CD-ROM (Apple usually sets their CD-ROM drives to SCSI ID 3):

  ```
  0 > boot cd:0 NETBSD.MACPPC
  0 > boot cd:0 NETBSD.601
  0 > boot scsi-int/sd@3:0 NETBSD.MACPPC
  0 > boot scsi/sd@3:0 NETBSD.MACPPC
  0 > boot ata/atapi-disk:0 NETBSD.MACPPC
  0 > boot ide1/disk@0:0 NETBSD.MACPPC
  ```

- *Booting the NetBSD/macppc install CD-R*
  (Open Firmware 3)

  ```
  0 > boot cd:,\ofwboot.xcf netbsd.macppc
  ```

- *Booting the NetBSD install floppies*
  (Open Firmware 1.0.5, Open Firmware 1.1.22, Open Firmware 2.0.x, Open Firmware 2.4)

  The first floppy disk has a 'partition zero' bootloader. When it has loaded the kernel, it will ask you to insert the second floppy disk.

  Note: This is not a normal MacOS boot floppy -- you must enter Open Firmware and type a boot command.

  All you need to do is:

  ```
  0 > boot fd:0
  ```

  It is common to see "READ TIMEOUT@" from the floppy. This means that Open Firmware is having difficulty reading the media. See the section below on **Common Problems and Error Messages** for more information about this and what you can do to get a successful boot.

  You can eject a floppy by typing:

```
0 > eject fd
```

- *Booting an IDE or SCSI drive with an HFS or HFS+ partition*
  (Open Firmware 2.4, Open Firmware 3)

  MacOS drives have several system-level partitions reserved for MacOS drivers. You may find that your first HFS or HFS+ partition might be as high as partition 9. You may need to keep trying higher partition numbers until you find the one that has your bootloader. You can use **pdisk** to print out your partition map:

  http://www.NetBSD.org/ports/macppc/partitioning.html#pdisk

  You should use the Open Firmware **dir** command to get a directory listing of the files on your hard drive.

  If bootloader is on a different partition from the NetBSD kernel, you will need to specify where to find the kernel.

  Remember, that SCSI Zip disks are usually ID 5 or 6. Internal hard drives are usually SCSI ID 0.

  Here are some examples of what you might use to boot the installer kernel located on an HFS or HFS+ partition:

```
0 > boot hd:9,\ofwboot.xcf netbsd-GENERIC_MD.gz
0 > boot scsi/sd@0:9,\ofwboot.xcf netbsd-GENERIC_MD.gz
0 > boot ide0/disk@0:10,\ofwboot.xcf netbsd-GENERIC_MD.gz
0 > boot ultra1:9,ofwboot.xcf netbsd-GENERIC_MD.gz
```

- *Booting a custom CD-ROM*
  (All Open Firmware versions)

  Open Firmware doesn't understand long filenames (created with the RockRidge and Joliet extensions), so you may need to figure out what your file is called. You should use the Open Firmware **dir** command to get a listing of the files on your CD.

  For instance, **Toast** for MacOS creates CDs with long filenames, but uses MS-DOS style short names as well. Instead of referencing `netbsd-GENERIC` you would need to open `NETBSD-G.ENE` The **mkisofs** program has a similar problem, `netbsd-GENERIC_MD.gz` becomes `NETBSD-GENERIC_MD.GZ`.

  Another thing to note is that you *must* use the same case when specifying the filename to load that Open Firmware uses. Also, keep in mind what format your CD-R has to be for your version of Open Firmware (pure ISO versus hybrid ISO/HFS).

  We'll assume that you've wisely renamed your kernel to `netbsd.gz` to avoid many of these name issues. Here are some examples of what you might use to boot from CD-ROM (Apple usually sets their CD-ROM drives to SCSI ID 3):

```
0 > boot cd:,\ofwboot.xcf netbsd.gz
0 > boot scsi-int/sd@3:,\OFWBOOT.XCF NETBSD.GZ
0 > boot scsi/sd@3:,\OFWBOOT.XCF NETBSD.GZ
0 > boot ata/atapi-disk:,\OFWBOOT.XCF NETBSD.GZ
0 > boot ide1/disk@0:,\OFWBOOT.XCF NETBSD.GZ
```

- *Booting an MS-DOS floppy*
  (Open Firmware 1.0.5, Open Firmware 1.1.22, Open Firmware 2.0.x, Open Firmware 2.4)

  Note: This is not a normal MacOS boot floppy -- you must enter Open Firmware and type a boot command.

Use the Open Firmware **dir** command to get a listing of the files on the floppy. Typically you'll find filenames like `OFWBOOT.XCF` and `NETBSD~1.GZ`.

```
0 > boot fd:,\OFWBOOT.XCF NETBSD~1.GZ
```

You can eject a floppy by typing:

```
0 > eject fd
```

- *Booting over the ethernet*
  (Open Firmware 1.0.5, Open Firmware 1.1.22, Open Firmware 2.0.x, Open Firmware 2.4)

  You can try the simple form (i.e. that you are booting from ethernet):

  ```
  0 > boot enet:,ofwboot.xcf
  ```

  Or you may be more specific, specifying the bootloader filename and the kernel name:

  ```
  0 > boot enet:,ofwboot.xcf enet:,/netbsd-GENERIC_MD.gz
  ```

  Note: Some Open Firmware 1.0.5 machines have their MAC address stored incorrectly. Make sure that your netboot server is using the same MAC address that your macppc client is using. See the section on **Setting up Open Firmware 1 and 2 to boot NetBSD** to figure out your MAC address.

  Note: Some machines cannot load compressed kernels over ethernet. Uncompress them first.

- *Booting over the ethernet*
  (Open Firmware 3)

  You can try the simple form (i.e. that you are booting from ethernet):

  ```
  0 > boot enet:0
  ```

  Or you may be more specific, specifying the bootloader filename and the kernel name:

  ```
  0 > boot enet:0,ofwboot.xcf enet:0,/netbsd-GENERIC_MD.gz
  ```

- *Booting an IDE or SCSI drive with a 'partition zero' bootloader*
  (Open Firmware 1.0.5, Open Firmware 1.1.22, Open Firmware 2.0.x, Open Firmware 2.4)

  You do not specify a file to load, since the 'partition zero' bootloader knows what to do. You would boot such a system if you have dumped the CD-R image to your drive, or if you have an already-installed NetBSD/macppc system

  Remember, that SCSI Zip disks are usually ID 5 or 6. Internal hard drives are usually SCSI ID 0.

  Here are some examples of what you might use to boot from such a drive:

  ```
  0 > boot zip:0
  0 > boot scsi-int/sd@0:0
  0 > boot scsi/sd@0:0
  0 > boot ata/ata-disk@0:0
  0 > boot ata/ATA-Disk@0:0
  0 > boot ide0/disk@0:0
  ```

Example of a normal boot

Of course, a lot of the information in this example depends on your model and what your boot method is, but we'll include this anyways just so you get an idea of what to expect (user-typed commands are in **bold**).

```
Apple PowerBook3,1 2.1f1 BootROM built on 01/29/00 at 22:38:07
Copyright 1994-2000 Apple Computer, Inc.
All Rights Reserved

Welcome to Open Firmware.
To continue booting, type "mac-boot" and press return.
To shut down, type "shut-down" and press return.
 ok
0 > boot enet:,ofwboot.xcf netbsd-GENERIC_MD.gz
loading XCOFF
tsize=C280 dsize=14AC bsize=2620 entry=600000
SECTIONS:
.text    00600000 00600000 0000C280 000000E0
.data    0060D000 0060D000 000014AC 0000C360
.bss     0060E4B0 0060E4B0 00002620 00000000
loading .text, done..
loading .data, done..
clearing .bss, done..

>> NetBSD/macppc OpenFirmware Boot, Revision 1.3
>> (tsubai@mint.iri.co.jp, Sun Nov 26 01:41:27 JST 2000)
1701508+177748 [100+68176+55886]=0x1e9468

 start=0x100000
Copyright (c) 1996, 1997, 1998, 1999, 2000, 2001
    The NetBSD Foundation, Inc.  All rights reserved.
Copyright (c) 1982, 1986, 1989, 1991, 1993
    The Regents of the University of California.  All rights reserved.

NetBSD 1.5.1 (INSTALL) #0: Thu Mar 15 00:52:56 PST 2001
    mw@al:/usr/src/sys/arch/macppc/compile/INSTALL
total memory = 192 MB
avail memory = 172 MB
using 2483 buffers containing 9932 KB of memory
[...]
erase ^H, werase ^W, kill ^U, intr ^C, status ^T
Terminal type? [vt100]
Erase is backspace.
(I)nstall, (S)hell or (H)alt ?
```

Common Problems and Error Messages

This is a brief list of some of the Open Firmware problems you may run into. See the NetBSD/macppc FAQ for a thorough list.

http://www.NetBSD.org/ports/macppc/faq.html#boot-trouble

Note: You may find it necessary to remove all non-Apple devices from your machine. Some users have found this necessary.

- Black screen

  If your system is a PowerBook, NetBSD may have turned down the brightness of the backlight. Use the buttons to turn your brightness back up.

  Open Firmware 3.x and newer often won't setup the console if you don't drop to a prompt, and NetBSD doesn't know how to setup the console itself. To get the system to boot when you have auto-boot? set to true, you can try this workaround. First, try one of the following commands:

  ```
  setenv boot-command " screen" output boot
  setenv boot-command ." screen" output boot
  setenv  boot-command  ."  Booting  NetBSD..."  cr  "  screen"  output
  boot
  ```

  You should follow the command with:

  ```
  reset-all
  ```

  This will cause Open Firmware to output some text to the screen prior to booting, thus forcing it to setup the console.

  Otherwise, you need a serial console. See the section entitled **Open Firmware 1 and 2 System Preparation**

- Grey screen with flashing question mark

  This means that your system is trying to boot MacOS. You might get to this state if you didn't properly enter Open Firmware, or if your Open Firmware boot command has bad syntax.

  If you typed a bad boot command, Open Firmware tries to boot from the value stored in the `boot-device` variable, whose default value is the MacOS ROM.

  The grey screen with the icon is generated by the MacOS ROM. A flashing question-mark or broken folder means that your machine is looking for a bootable MacOS file system, but can't find one. A globe icon means that your machine is looking for a netboot server. A happy face or spinning disc means it's starting to boot MacOS.

  The boot floppy does not have a bootable MacOS file system, it has a bootable NetBSD file system.

  Reboot, re-enter Open Firmware and check your syntax carefully.

  Note: Don't forget to check your Open Firmware environment variables, as they may have been changed by your brief excursion into MacOS.

- Information on your screen seems garbled or out of sync

  If you have a PowerMacintosh 7300 through 8600, then you need to read the section on System Disk and the patches it applies. See the section entitled **Open Firmware 1 and 2 System Preparation**

  If you have a Twentieth Anniversary Macintosh, Open Firmware does not work with the internal display, you will need to set up a serial console.

- “`DEFAULT CATCH!`”

  This is a general message from Open Firmware to the effect that it had trouble loading a file.

  If your machine is Open Firmware version 1.0.5, 2.0.x, or 2.4, this error does sometimes appear randomly. You might try the boot command a second time (this is known to work on some models). It's also an indication that either your floppy disk is bad, or the floppy drive is bad. Try doing a low-level format on the floppy, re-copy the files, and try again.

There are several models that cannot be booted while using the on-board video and keyboard. Try using a serial console.

- "`CLAIM failed`"

  This is a general message from Open Firmware to the effect that it failed to allocate some memory or memory is messed up.

  First, make sure you have the Open Firmware variable `load-base` set correctly. If your system is Open Firmware version 1.0.5, 1.1.22, 2.0.x, or 2.4, see the section above on **Setting up Open Firmware 1 and 2 to boot NetBSD**

  Do not change `load-base` on an Open Firmware 3 system.

  Reset your system and try using a different boot command (this is the only way some people have gotten their Open Firmware 2.4 systems to work):

  ```
  0 > 0 bootr boot-device boot-file
  ```

  If you tried one boot command and it failed, then you tried a second and got the `CLAIM failed` message, then this is an indication that you should reboot between attempts. Use the Open Firmware **`reset-all`** command.

  Contradictorily, if your machine is Open Firmware version 1.0.5, 2.0.x or 2.4, this error does sometimes appear randomly. You might try the boot command a second time (this is known to work on some models).

  There are several models that cannot be booted while using the on-board video and keyboard. Try using a serial console.

- "`can't OPEN`"

  Open Firmware either can't open the device you specified (because it is not present or the device path is mistyped) or the file you specified. Check your typing and check to make sure that the media has the files you think it has.

  Use the NetBSD **`pdisk`** command to list the partition map and the Open Firmware **`dir`** command to verify that the file(s) you tried accessing are really there.

- "`unrecognized Client Program formatstate not valid`"

  This is a general Open Firmware error message indicating that the filename you tried to open either doesn't exist or is in the wrong format. For Open Firmware 1 and 2 machines, it must be an XCOFF file (such as `ofwboot.xcf`) and for Open Firmware 3 machines, it must be either XCOFF or ELF (such as a kernel). Make sure that you have use binary mode to FTP the files, and that they are properly uncompressed.

- "`bad partition number, using 0no bootable HFS partition`"

  If you're trying to boot an Open Firmware 1.0.5, 1.1.22, or 2.0.x system, this probably means that your media (i.e. hard drive or CD-ROM) has an HFS file system on it (such as a hybrid CD-R or a hard drive with MacOS partitions).

- "`READ TIMEOUT@`"

  Open Firmware is having trouble reading your floppy disk. Things to try:
  - Try booting from the floppy again

- Try a different floppy disk
- Try writing the floppy on the same machine you're trying to boot
- Clean the floppy drive
- Use another boot method

As disk drives get older, especially with portables, they can get a little bit out of alignment so that they do not consistently read disks written on other floppy drives. Strongly consider writing the floppies on the same machine that will be reading them.

- "`TFTP timeout`"

  Either the server's TFTP server isn't running, or you're using a model with Open Firmware 1.0.5 and not specifying the location of the bootloader. Unfortunately, on the early models, `ofwboot.xcf` gets confused and doesn't work right unless you load it explicitly from the `boot-device`, even if the bootp or DHCP server provides the correct information. You need to boot with a command like:

  ```
  0 > boot enet:,ofwboot.xcf
  ```

- "`enet:,/netbsd.ram.gz: Inappropriate file type or format`"

  Some systems booting over ethernet can't load compressed kernels. Use `gunzip`(1) to uncompress the kernel before attempting to netboot.

- Bootloader hangs before the copyright notice and the kernel configuration

  You forgot to set `real-base` in Open Firmware or it got erased by your booting into MacOS.

  Make sure you have the Open Firmware variable `real-base` set correctly. If your system is Open Firmware version 1.0.5, 1.1.22, 2.0.x, or 2.4, see the section above on **Setting up Open Firmware 1 and 2 to boot NetBSD**

  Do not set `real-base` on an Open Firmware 3 system.

- Hang after configuring devices, but before doing anything else

  Actually, this can have many causes. The most likely is a keyboard problem. First, try plugging the USB keyboard directly into the computer (i.e. not through a hub) and unplugging the mouse.

  If you're trying a model that's not on the supported list (such as a new laptop model), they keyboard may not be supported yet.

Milestone

If you've reached this point, then you must've gotten the NetBSD installer to boot. Congratulations! That was the hard part. From now through the rest of this document, there should be no more Open Firmware specific problems, so read everything because it applies to all models.

Running the sysinst installation program

1. *Introduction*

   Using **sysinst**, installing NetBSD is a relatively easy process. Still, you should read this document and have it available during the installation process. This document tries to be a good guide to the installation, and as such, covers many details for the sake of completeness. Do not let this discourage you; the install program is not hard to use.

2. *General*

   The following is a walk-through of the steps you will take while installing NetBSD on your hard disk. **sysinst** is a menu-driven program that guides you through the installation process. Sometimes questions will be asked, and in many cases the default answer will be displayed in brackets ("[ ]") after the question. If you wish to stop the installation, you may press `CONTROL-C` at any time, but if you do, you'll have to begin the installation process again from scratch by running the `/sysinst` program from the command prompt. It is not necessary to reboot.

3. *Quick install*

   First, let's describe a quick install. The other sections of this document go into the installation procedure in more detail, but you may find that you do not need this. If you want detailed instructions, skip to the next section. This section describes a basic installation, using a CD / DVD as the install medium.

- What you need.
  - The distribution sets (in this example, they are on the CD or DVD).
  - Some form of bootable media, described above.
  - A minimum of 16 MB of memory installed.
  - An optical drive.
  - A hard drive with at least 600 MB of free space for a complete base install, not including room for swap. If you wish to install the X Window System as well, you will need at least 225 MB more.
- The Quick Installation
  - Boot the system as described above. You should be at the **`sysinst`** main menu.

```
.***********************************************.
* NetBSD-10.2_STABLE Install System             *
*                                               *
*>a: Install NetBSD to hard disk                *
* b: Upgrade NetBSD on a hard disk              *
* c: Re-install sets or install additional sets *
* d: Reboot the computer                        *
* e: Utility menu                               *
* f: Config menu                                *
* x: Exit Install System                        *
.***********************************************.
```

  - If you wish, you can configure some network settings immediately by choosing the **Utility menu** and then **Configure network**. It isn't actually required at this point, but it may be more convenient. Go back to the main menu.
  - Choose **Utility menu** then **Run /bin/sh** for Open Firmware 3 systems. You'll need to create the NetBSD filesystem(s) and an `fstab`. Now exit the shell, return to the main menu, and choose **Re-install**.

    Choose **Install** for Open Firmware 1.0.5, 1.1.22, 2.0.x, and 2.4 systems.
  - You will be guided through the setup of your disk.
  - You will be asked to choose which distribution sets to install.
  - When prompted, choose **CD-ROM** as the install medium if booted from CD-ROM. The default values for the path and device should be OK.
  - After the installation process has completed, you will be brought back to the main menu, where you should select **Reboot.**

- Once the system reaches the Open Firmware prompt, you will need to type the correct command to boot from your hard drive.
- NetBSD will now boot. If you didn't set a password for the `root` user when prompted by **`sysinst`**, logging in as `root` and setting a password should be your first task. You are also advised to read `afterboot`(8).

4. *Booting NetBSD*

   You may want to read the boot messages, to notice your disk's name and capacity. Its name will be something like `sd0` or `wd0` and the geometry will be printed on a line that begins with its name. As mentioned above, you may need your disk's geometry when creating NetBSD's partitions. You will also need to know the name, to tell **`sysinst`** which disk to use. The most important thing to know is that `wd0` is NetBSD's name for your first IDE disk, `wd1` the second, etc. `sd0` is your first SCSI disk, `sd1` the second, etc.

   Once NetBSD has booted and printed all the boot messages, you will be presented with a welcome message and a main menu. It will also include instructions for using the menus.

5. *Network configuration*

   If you do not intend to use networking during the installation, but you do want your machine to be configured for networking once the system is installed, you should first go to the **Utility menu** and select the **Configure network** option. If you only want to temporarily use networking during the installation, you can specify these parameters later. If you are not using the Domain Name System (DNS), you can give an empty response when asked to provide a server.

6. *Preparing a disk which will be used for Open Firmware 3 systems*

   Skip this step if you are installing NetBSD on an Open Firmware 1 or 2 system.

   Go to the **Utility Menu**, and select the **Run /bin/sh** option which will give you a shell prompt. From this shell prompt, you will do some of the steps that the normal install procedure runs automatically. Unfortunately, at the moment, our install tools aren't smart enough to deal with drives with Apple Partition Maps and will overwrite important information describing your partitions.

   You may need to type one of the following commands to get your delete key to work properly, depending on your keyboard:

   ```
   # stty erase '^h'
   # stty erase '^?'
   ```

   Type the following command (replacing `wd0` with the name of your destination hard drive):

   ```
   # disklabel wd0
   ```

   This will print out the partition info that was generated by **`pdisk`**, Drive Setup, or Disk Utility. Note that, as discussed above in the **Partitioning your hard drive for NetBSD** section, your *A/UX Root* typically is the first partition (*a*) and your *A/UX Swap* typically is the second partition (*b*). You may also find that your *A/UX User* partition is the seventh partition (*g*). For example:

   ```
       # disklabel wd0
   [...]
   #        size   offset     fstype   [fsize bsize cpg/sgs]
     a:   426613   837432     4.2BSD        0     0     0   # (Cyl. 1622*- 2449*)
     b:   204800   632632       swap                      # (Cyl. 1226*- 1622*)
     c:  2134305        0     unused        0     0         # (Cyl.    0 - 4136*)
     d:   426616     1216        HFS                      # (Cyl.    2*- 829*)
     e:   204800   427832        HFS                      # (Cyl.  829*- 1226*)
   ```

```
  f:       21  2134284    unknown                       # (Cyl. 4136*- 4136*)
  g:   870239  1264045     4.2BSD        0     0     0   # (Cyl. 2449*- 4136*)
disklabel: boot block size 0
disklabel: super block size 0
```

Now, you need to create file systems on the partitions that NetBSD will be using.

Do not modify any partitions labeled `HFS`, `UFS`, *or* `unknown`. The partitions you will be using have their `fstype` listed as `4.2BSD`.

Run the **newfs** command on the `4.2BSD` partitions:

```
     # newfs /dev/wd0a
newfs: /dev/wd0a: not a character-special device
Warning: 120 sector(s) in last cylinder unallocated
/dev/wd0a:      426612 sectors in 827 cylinders of 4 tracks, 129 sectors
        208.3MB in 52 cyl groups (16 c/g, 4.03MB/g, 1024 i/g)
super-block backups (for fsck -b #) at:
     32,   8432,  16832,  25232,  33056,  41456,  49856,  58256,  66080,
  74480,  82880,  91280,  99104, 107504, 115904, 124304, 132128, 140528,
 148928, 157328, 165152, 173552, 181952, 190352, 198176, 206576, 214976,
 223376, 231200, 239600, 248000, 256400, 264224, 272624, 281024, 289424,
 297248, 305648, 314048, 322448, 330272, 338672, 347072, 355472, 363296,
 371696, 380096, 388496, 396320, 404720, 413120, 421520,
newfs: ioctl (WDINFO): Invalid argument
newfs: /dev/wd0a: can't rewrite disk label
```

You can ignore the ˜ 'not a character-special device', ˜ 'sector(s) in last cylinder unallocated', ˜ 'ioctl (WDINFO): Invalid argument', and ˜ 'can't rewrite disk label' warnings.

Now you need to mount your destination root partition:

```
# mount /dev/wd0a /mnt
```

Make an `fstab` file for your new system (right now, you only really need to include `/`, `/usr`, and `swap`), for example:

```
# mkdir /mnt/etc
# cat > /mnt/etc/fstab
/dev/wd0a / ffs rw 1 1
/dev/wd0b none swap sw 0 0
/dev/wd0g /usr ffs rw 1 2
```

If you mess up while typing, you can press `CONTROL-U` to erase everything on the current line, or `CONTROL-C` to cancel the file creation, so you can start over. `CONTROL-D` finishes and writes the file to disk.

Great, now create the mountpoints for the file systems you listed in the `fstab`:

```
# mkdir /mnt/usr
```

Clean up and return to **sysinst**:

```
# cd /
# umount /mnt
# exit
```

7. *Installation drive selection and parameters*

To start the installation onto a dedicated NetBSD drive (Open Firmware 1 or 2), select **Install NetBSD to hard disk** from the main menu. To start the installation onto a drive with an Apple Partition Map

(Open Firmware 3), select **Re-install sets or install additional sets** from the main menu.

The first thing is to identify the disk on which you want to install NetBSD. **sysinst** will report a list of disks it finds and ask you for your selection. You should see disk names like `wd0`, `wd1`, `sd0` or `sd1`.

If **sysinst** reports

```
I can not find any hard disk for use by NetBSD
```

or the drive you wish to install onto is missing, then you should look at the *FAQ entry:*
https://www.NetBSD.org/ports/macppc/faq.html#nodisk.

8. *Selecting which sets to install*

   The next step is to choose which distribution sets you wish to install. Options are provided for full, minimal, and custom installations. If you choose sets on your own, **base**, **etc**, and a kernel must be selected.

9. *Partitioning the disk*

   You can skip a few steps, down to '*Getting the distribution sets*', if you are installing onto a drive with an Apple Partition Map (Open Firmware 3), i.e., you selected **Re-install sets or install additional sets** from the main menu.

10. *Editing the NetBSD disklabel*

    The partition table of the NetBSD part of a disk is called a *disklabel*. In actuality, NetBSD/macppc uses an Apple Partition Map. The installer creates something like a real Apple Partition Map, but it is not compatible with Mac OS or Open Firmware, which is one of the reasons why you cannot use this installer to partition a disk that can be used with Mac OS or Open Firmware 3 systems.

    If your disk already has a disklabel written to it, you can choose **`Use existing partition sizes`**. Otherwise, select **`Set sizes of NetBSD partitions`**.

    After you have chosen your partitions and their sizes (or if you opted to use the existing partitions), you will be presented with the layout of the NetBSD disklabel and given one more chance to change it. For each partition, you can set the type, offset and size, block and fragment size, and the mount point. The type that NetBSD uses for normal file storage is called **4.2BSD**. A swap partition has a special type called **swap**. Some partitions in the disklabel have a fixed purpose.

`a`	Root partition (/)
`b`	Swap partition.
`c`	The entire disk.
`d-p`	Available for other use. Traditionally, `g` is the partition mounted on `/usr`, but this is historical practice and not a fixed value.

    You will then be asked to name your disk's disklabel. The default response will be OK for most purposes. If you choose to name it something different, make sure the name is a single word and contains no special characters. You don't need to remember this name.

11. *Preparing your hard disk*

    You are now at the point of no return. Nothing has been written to your disk yet, but if you confirm that you want to install NetBSD, your hard drive will be modified. If you are sure you want to proceed, select **yes**.

    The install program will now label your disk and create the file systems you specified. The file systems will be initialized to contain NetBSD bootstrapping binaries and configuration files. You will see messages on your screen from the various NetBSD disk preparation tools that are running. There should be no errors in this section of the installation. If there are, restart from the beginning of the installation process. Otherwise, you can continue the installation program after pressing the return key.

Note: The bootstrapping code installed in this step will *not* boot a machine with Open Firmware 3. You will still need to have `ofwboot.xcf` on an HFS or HFS+ partition.

12. *Getting the distribution sets*

    The NetBSD distribution consists of a number of *sets* that come in the form of gzipped tar files. At this point, you will be presented with a menu which enables you to choose from one of the following methods of installing the sets. Some of these methods will first transfer the sets to your hard disk, others will extract the sets directly.

    For all these methods, the first step is to make the sets available for extraction. The sets can be made available in a few different ways. The following sections describe each of the methods. After reading about the method you will be using, you can continue to the section labeled 'Extracting the distribution sets'.

13. *Installation from CD-ROM*

    When installing from a CD-ROM, you will be asked to specify the device name for your CD-ROM drive (usually `cd0`) and the directory name on the CD-ROM where the distribution files are.

    `sysinst` will then check that the files are actually present in the specified location and proceed to the extraction of the sets.

14. *Installation using FTP*

    To install using FTP, you first need to configure your network setup if you haven't already done so. **`sysinst`** will help you with this, asking if you want to use DHCP. If you do not use DHCP, you can enter network configuration details yourself. If you do not have DNS set up for the machine that you are installing on, you can just press RETURN in answer to this question, and DNS will not be used.

    You will also be asked to specify the host that you want to transfer the sets from, the directory on that host, the account name and password used to log into that host using FTP, and optionally a proxy server to use. If you did not set up DNS, you will need to specify an IP address instead of a hostname for the FTP server.

    `sysinst` will then transfer the set files from the remote site to your hard disk.

15. *Installation using NFS*

    To install using NFS, you first need to configure your network setup if you haven't already done so. **`sysinst`** will do this for you, asking you if you want to use DHCP. If you do not use DHCP, you can enter network configuration details yourself. If you do not have DNS set up for the machine that you are installing on, you can just press RETURN in answer to this question, and DNS will not be used.

    You will also be asked to specify the host that you want to transfer the sets from and the directory on that host that the files are in. This directory should be mountable by the machine you are installing on, i.e., correctly exported to your machine.

    If you did not set up DNS, you will need to specify an IP address instead of a hostname for the NFS server.

16. *Installation from an unmounted file system*

    In order to install from a local file system, you will need to specify the device that the file system resides on (for example `wd1e`), the type of the file system, and the directory on the specified file system where the sets are located. **`sysinst`** will then check if it can indeed access the sets at that location. Remember, NetBSD/macppc doesn't grok HFS or HFS+ partitions

17. *Installation from a local directory*

    This option assumes that you have already done some preparation yourself. The sets should be located in a directory on a file system that is already accessible. **sysinst** will ask you for the name of this directory.

18. *Extracting the distribution sets*

    A progress bar will be displayed while the distribution sets are being extracted.

    After all the files have been extracted, the device node files will be created. If you have already configured networking, you will be asked if you want to use this configuration for normal operation. If so, these values will be installed in the network configuration files.

19. *Configure additional items*

    The next menu will allow you to select a number of additional items to configure, including the time zone that you're in, to make sure your clock has the right offset from UTC, the root user's shell, and the initial root password.

    You can also enable installation of binary packages, which installs the `pkgin`(1) tool for managing binary packages for third-party software. This will feel familiar to users of package tools such as **apt-get** or **yum**. If you prefer to install third-party software from source, you can install the `pkgsrc`(7) tree.

    Finally, you can enable some daemons such as `sshd`(8), `ntpd`(8), or `mdnsd`(8).

20. *Finalizing your installation*

    Congratulations, you have successfully installed NetBSD 10.2_STABLE.

Finalizing Open Firmware settings

Now, you can reboot to get to the Open Firmware prompt. You still need to figure out how to get Open Firmware to boot the operating system(s) of your choice. First, try to get NetBSD/macppc running. Using the methods described in the section **Examples of Open Firmware boot commands** figure out the boot command for your installation of NetBSD. Try booting. Once you've got the syntax worked out, decide which operating systems you'll be using regularly.

- *Booting NetBSD exclusively*
  If you'll only be using NetBSD on your macppc system, then simply set the Open Firmware '`boot-device`' and '`boot-file`' variables to the values you just determined. Also, you might want to enable the system to always boot NetBSD when powered on or reset. Once you set up auto-booting you can get to the Open Firmware prompt again by using the `shutdown`(8) command to halt the system.

  If you are not using a 'partition zero' style boot scheme (e.g. Open Firmware 3 models), then you would type something like the following:

  ```
  0 > setenv auto-boot? true
  0 > setenv boot-device ide0/disk@0:8,\ofwboot.xcf
  0 > setenv boot-file ide0/disk@0:13,/netbsd
  0 > reset-all
  ```

  The last command resets the system so that these settings are stored.

  If you are using a 'partition zero' style boot scheme, you would type something like the following:

  ```
  0 > setenv auto-boot? true
  0 > setenv boot-device scsi/sd@0:0
  0 > setenv boot-file netbsd
  0 > reset-all
  ```

Replace **`scsi/sd@0:0`** with the actual device you will be booting from.

- *Additional Open Firmware tips*
  If you find that your system tries booting before your hard drive has spun up, try one of the two following '`boot-command`' settings:

  ```
  0 > setenv boot-command catch 5000 ms boot
  0 > setenv boot-command begin ['] boot catch 1000 ms cr again
  ```

  Also, you may be able to pause a system at the Open Firmware prompt if you have '`auto-boot?`' set to '`true`' by holding down any key while the system is resetting. Set the following '`boot-command`' (this may not work on system with Open Firmware 1.0.5) (this method should interrupt booting, even when holding down any key while using a serial console):

  ```
  0 > setenv boot-command key? invert if boot then
  ```

- *Booting NetBSD and MacOS X or Darwin*
  Alas, this takes a little more work. Usually, when you select a system to boot in the "Startup Disk" panel of the "System Preferences" application, it stores the Open Firmware path to that device in the '`boot-device`' variable. So, instead of writing the NetBSD device path to Open Firmware, you'll store the paths to your operating systems in NVRAM. Open Firmware cannot deal with nested **`devalias`** entries. You must, therefore, enter the *entire* path to your device. See the NetBSD/macppc **`nvedit`** HOW-TO for more help:
  http://www.NetBSD.org/ports/macppc/nvedit.html

  ```
  0 > printenv boot-device
  boot-device /pci@f2000000/mac-io@17/ata-4@1f000/@0:10,\\:tbxi
   ok
  0 > nvalias osx /pci@f2000000/mac-io@17/ata-4@1f000/@0:10,\\:tbxi
  0 > nvalias bsd /pci@f2000000/mac-io@17/ata-4@1f000/@0:9,ofwboot.xcf
  0 > nvstore
  0 > setenv use-nvramrc? true
  0 > reset-all
  ```

  Now, when the system is reset, it will stop at the Open Firmware prompt and you can type one of the following to boot an operating system:

  ```
  0 > boot osx
  0 > boot bsd
  ```

- *Booting NetBSD and MacOS 9 or earlier*
  For Open Firmware 3 systems, the procedure is identical to the section above on **Booting NetBSD and MacOS X or Darwin**

  For older systems, you're in for more hassle. Booting an Open Firmware 1.0.5, 2.0.x, or 2.4 system into MacOS 9 or earlier will erase some or all of your Open Firmware settings. Try it and see which Open Firmware variables survive.

  If only '`real-base`' is lost, you can compile a kernel that does not require changing the '`real-base`'. Just build a kernel that is less than 4 MB uncompressed. This is easy if you remove all of the USB devices from the config file. Once you've got a smaller kernel, just follow the procedure in the section above on **Booting NetBSD and MacOS X or Darwin**

  If everything is lost when you boot into MacOS 9 or earlier, you will need to make a custom BootVars configuration that you run before trying to boot NetBSD/macppc since System Disk does not preserve enough information when it saves a configuration.

1. If your system supports System Disk, run it and click "Save" to install the NVRAMRC patches.
2. Run BootVars, click the "All Variables" button.
3. Fill in `real-base`, `boot-device`, `boot-file`, `input-device`, and `output-device`.
4. From the "File" menu, pick "Save". When you run this file, it will load BootVars with all your settings. To boot NetBSD click the "Write&reboot" button.
5. Now your system will always boot NetBSD. To boot MacOS, use the `shutdown`(8) command to halt the system at the Open Firmware prompt and use the Open Firmware "`bye`" or "`mac-boot`" command to boot MacOS.

- *Other boot techniques*
  See the FAQ for some vague information on how to use the Linux **yaboot** system to boot NetBSD.
  http://www.NetBSD.org/ports/macppc/faq.html#yaboot

Post installation steps

Once you've got the operating system running, there are a few things you need to do in order to bring the system into a properly configured state. The most important steps are described below.

1. Before all else, read `postinstall`(8).
2. Configuring `/etc/rc.conf`

   If you or the installation software haven't done any configuration of `/etc/rc.conf` (**sysinst** normally will), the system will drop you into single user mode on first reboot with the message

   ```
   /etc/rc.conf is not configured. Multiuser boot aborted.
   ```

   and with the root file system (`/`) mounted read-only. When the system asks you to choose a shell, simply press RETURN to get to a `/bin/sh` prompt. If you are asked for a terminal type, respond with **vt100** (or whatever is appropriate for your terminal type) and press RETURN. You may need to type one of the following commands to get your delete key to work properly, depending on your keyboard:

   ```
   # stty erase '^h'
   # stty erase '^?'
   ```

   At this point, you need to configure at least one file in the `/etc` directory. You will need to mount your root file system read/write with:

   ```
   # /sbin/mount -u -w /
   ```

   Change to the `/etc` directory and take a look at the `/etc/rc.conf` file. Modify it to your tastes, making sure that you set `rc_configured=YES` so that your changes will be enabled and a multi-user boot can proceed. Default values for the various programs can be found in `/etc/defaults/rc.conf`, where some in-line documentation may be found. More complete documentation can be found in `rc.conf`(5).

   When you have finished editing `/etc/rc.conf`, type **exit** at the prompt to leave the single-user shell and continue with the multi-user boot.

   Other values that may need to be set in `/etc/rc.conf` for a networked environment are *hostname* and possibly *defaultroute*. You may also need to add an *ifconfig_int* for your ⟨int⟩ network interface, where your on-board interfaces may be one of `mc0`, `bm0`, `tlp0` or `gem0`. For example:

   ```
   ifconfig_mc0="inet 192.0.2.123 netmask 255.255.255.0"
   ```

   or, if you have *myname.my.dom* in `/etc/hosts`:

   ```
   ifconfig_mc0="inet myname.my.dom netmask 255.255.255.0"
   ```

To enable proper hostname resolution, you will also want to add an `/etc/resolv.conf` file or (if you are feeling a little more adventurous) run `named`(8). See `resolv.conf`(5) or `named`(8) for more information.

Instead of manually configuring networking, DHCP can be used by setting `dhcpcd=YES` in `/etc/rc.conf`.

3. Logging in

   After reboot, you can log in as `root` at the login prompt. If you didn't set a password in **sysinst**, there is no initial password. You should create an account for yourself (see below) and protect it and the "root" account with good passwords. By default, root login from the network is disabled (even via `ssh`(1)). One way to become root over the network is to log in as a different user that belongs to group "wheel" (see `group`(5)) and use `su`(1) to become root.

4. Adding accounts

   Use the `useradd`(8) command to add accounts to your system. Do not edit `/etc/passwd` directly! See `vipw`(8) and `pwd_mkdb`(8) if you want to edit the password database.

5. The X Window System

   If you installed the X Window System, you may want to read the chapter about X in the *NetBSD Guide:*
   https://www.NetBSD.org/docs/guide/en/chap-x.html:

6. Installing third-party packages

   If you wish to install any of the software freely available for UNIX-like systems you are strongly advised to first check the NetBSD package system, pkgsrc. pkgsrc automatically handles any changes necessary to make the software run on NetBSD. This includes the retrieval and installation of any other packages the software may depend upon.

   - More information on the package system is available at
     https://www.NetBSD.org/docs/software/packages.html
   - A list of available packages suitable for browsing is at
     https://cdn.NetBSD.org/pub/pkgsrc/current/pkgsrc/README.html
   - Precompiled binaries can be found at
     https://cdn.NetBSD.org/pub/pkgsrc/packages/NetBSD/
     usually in the `macppc/10.2_STABLE/All` subdir. If you installed `pkgin`(1) in the **sysinst** post-installation configuration menu, you can use it to automatically install binary packages over the network. Assuming that `/usr/pkg/etc/pkgin/repositories.conf` is correctly configured, you can install them with the following commands:

     ```
     # pkgin install tcsh bash perl apache xfce4 firefox
     ...
     ```

     The above command will install the Tenex-csh and Bourne Again shells, the Perl programming language, Apache web server, Xfce desktop environment and the Firefox web browser as well as all the packages they depend on.

     If it was not automatically installed, `pkgin`(1) can be installed on a fresh NetBSD system with `pkg_add`(1):

     ```
     export PKG_PATH=https://cdn.netbsd.org/pub/pkgsrc/packages/NetBSD/$(uname -p)/$(uname
     pkg_add pkgin
     ```

- If you wish to use the `pkgsrc`(7) framework for compiling packages and did not install it from the `sysinst`(8) post-installation configuration menu, you can obtain it by retrieving the file

  https://cdn.NetBSD.org/pub/pkgsrc/stable/pkgsrc.tar.gz.

  It is typically extracted into `/usr/pkgsrc` (though other locations work fine) with the commands:

  ```
  # cd /usr
  # tar -zxpf pkgsrc.tar.gz
  ```

  After extracting, see the `doc/pkgsrc.txt` file in the extraction directory (e.g., `/usr/pkgsrc/doc/pkgsrc.txt`) for more information.

7. Misc

- Edit `/etc/mail/aliases` to forward root mail to the right place. Don't forget to run `newaliases`(1) afterwards.
- Edit `/etc/rc.local` to run any local daemons you use.
- Many of the `/etc` files are documented in section 5 of the manual; so just invoking

  ```
  # man 5 filename
  ```

  is likely to give you more information on these files.

Upgrading a previously-installed NetBSD System

The easiest way to upgrade to NetBSD 10.2_STABLE is with binaries, and that is the method documented here.

To do the upgrade, you must boot the install kernel using one of the methods described above. You must also have at least the **base** and **kern** binary distribution sets available. Finally, you must have sufficient disk space available to install the new binaries. Since files already installed on the system are overwritten in place, you only need additional free space for files which weren't previously installed or to account for growth of the sets between releases.

Since upgrading involves replacing the kernel, boot blocks, and most of the system binaries, it has the potential to cause data loss. You are strongly advised to *back up* any important data on the NetBSD partition or on another operating system's partition on your disk before beginning the upgrade process. Since installation of the bootloader will prevent Mac OS from using the disk and will prevent Open Firmware 3 machines from booting, there is an option to bypass this step. Make sure you know whether or not to install the bootloader.

The upgrade procedure is similar to an installation, but without the hard disk partitioning.

Fetching the binary sets is done in the same manner as the installation procedure; refer to the installation part of the document for help. File systems are checked before unpacking the sets.

After a new kernel has been copied to your hard disk, your machine is a complete NetBSD 10.2_STABLE system. However, that doesn't mean that you're finished with the upgrade process. You will probably want to update the set of device nodes you have in `/dev`. If you've changed the contents of `/dev` by hand, you will need to be careful about this, but if not, you can just cd into `/dev`, and run the command:

```
# sh MAKEDEV all
```

`sysinst` will attempt to merge the settings stored in your `/etc` directory with the new version of NetBSD using the `postinstall`(8) utility. However, `postinstall`(8) is only able to deal with changes that are easily automated. It is recommended that you use the `etcupdate`(8) tool to merge any remaining configuration changes.

Compatibility Issues With Previous NetBSD Releases

Users upgrading from previous versions of NetBSD may wish to bear the following problems and compatibility issues in mind when upgrading to NetBSD 10.2_STABLE.

Note that **sysinst** will automatically invoke

```
postinstall fix
```

and thus all issues that are fixed by **postinstall** by default will be handled.

In NetBSD 9 and earlier, filesystems listed in `/etc/fstab` would be mounted before non-legacy **zfs** filesystems. Starting from NetBSD 10 this order has been reversed.

If you have ever run a version of NetBSD -current between April 18, 2020 and September 23, 2022 (the version numbers used in the affected time range are between 9.99.56 and 9.99.106) your FFS file systems might have broken extended attributes stored.

You should follow this *guide:*
https://wiki.netbsd.org/features/UFS2ea/
before booting the updated system multi-user for the first time.

Note that you do not need to do anything special if you never did run any affected kernel, especially if you have never run NetBSD -current.

The display drivers used for modern GPUs and the whole subsystem supporting it (DRM/KMS) have been updated to a newer version. Unfortunately not all issues with this have been resolved before the NetBSD 10.0 release. You can find a list of issues in the *Open issues with new DRM/KMS:*
https://wiki.netbsd.org/releng/netbsd-10/
section of the release engineering wiki page.

A number of things have been removed from the NetBSD 10.2_STABLE release. See the "Components removed from NetBSD" section near the beginning of this document for a list.

Using online NetBSD documentation

Documentation is available if you installed the manual distribution set. Traditionally, the "man pages" (documentation) are denoted by '`name(section)`'. Some examples of this are

- `intro`(1),
- `man`(1),
- `apropos`(1),
- `passwd`(1), and
- `passwd`(5).

The section numbers group the topics into several categories, but three are of primary interest: user commands are in section 1, file formats are in section 5, and administrative information is in section 8.

The *man* command is used to view the documentation on a topic, and is started by entering **man** [*section*] *topic*. The brackets [] around the section should not be entered, but rather indicate that the section is optional. If you don't ask for a particular section, the topic with the lowest numbered section name will be displayed. For instance, after logging in, enter

```
# man passwd
```

to read the documentation for `passwd`(1). To view the documentation for `passwd`(5), enter

```
# man 5 passwd
```

instead.

If you are unsure of what man page you are looking for, enter

```
# apropos subject-word
```

where *subject-word* is your topic of interest; a list of possibly related man pages will be displayed.

Administrivia

If you've got something to say, do so! We'd like your input. There are various mailing lists available via the mailing list server at `majordomo@NetBSD.org`. See **https://www.NetBSD.org/mailinglists/** for details.

There are various mailing lists set up to deal with comments and questions about this release. Please send comments to: `netbsd-comments@NetBSD.org`.

To report bugs, use the `send-pr`(1) command shipped with NetBSD, and fill in as much information about the problem as you can. Good bug reports include lots of details.

Bugs also can be submitted and queried with the web interface at
https://www.NetBSD.org/support/send-pr.html

There are also port-specific mailing lists, to discuss aspects of each port of NetBSD. Use majordomo to find their addresses, or visit **https://www.NetBSD.org/mailinglists/**

If you're interested in doing a serious amount of work on a specific port, you probably should contact the 'owner' of that port (listed below).

If you'd like to help with NetBSD, and have an idea as to how you could be useful, send us mail or subscribe to: `netbsd-users@NetBSD.org`.

As a favor, please avoid mailing huge documents or files to these mailing lists. Instead, put the material you would have sent up for FTP or WWW somewhere, then mail the appropriate list about it. If you'd rather not do that, mail the list saying you'll send the data to those who want it.

Thanks go to

- The former members of UCB's Computer Systems Research Group, including (but not limited to):

  Keith Bostic
  Ralph Campbell
  Mike Karels
  Marshall Kirk McKusick

  for their work on BSD systems, support, and encouragement.
- The Internet Systems Consortium, Inc. for hosting the NetBSD FTP, CVS, AnonCVS, mail, mail archive, GNATS, SUP, Rsync and WWW servers.
- The Internet Research Institute in Japan for hosting the server which runs the CVSweb interface to the NetBSD source tree.
- The Columbia University Computer Science Department for hosting the build cluster.
- The many organizations that provide NetBSD mirror sites.
- Without CVS, this project would be impossible to manage, so our hats go off to Brian Berliner, Jeff Polk, and the various other people who've had a hand in making CVS a useful tool.
- We list the individuals and organizations that have made donations or loans of hardware and/or money, to support NetBSD development, and deserve credit for it at **https://www.NetBSD.org/donations/** (If you're not on that list and should be, tell us! We probably were not able to get in touch with you, to verify that you wanted to be listed.)

- Finally, we thank all of the people who've put sweat and tears into developing NetBSD since its inception in January, 1993. (Obviously, there are a lot more people who deserve thanks here. If you're one of them, and would like to be mentioned, tell us!)

Legal Mumbo-Jumbo

All product names mentioned herein are trademarks or registered trademarks of their respective owners.

The following notices are required to satisfy the license terms of the software that we have mentioned in this document:

NetBSD is a registered trademark of The NetBSD Foundation, Inc.
This product includes software developed by the University of California, Berkeley and its contributors.
This product includes software developed by the NetBSD Foundation.
This product includes software developed by The NetBSD Foundation, Inc. and its contributors.
This product includes software developed for the NetBSD Project. See https://www.NetBSD.org/ for information about NetBSD.
This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)
This product includes cryptographic software written by Eric Young (eay@mincom.oz.au)
This product includes software designed by William Allen Simpson.
This product includes software developed at Ludd, University of Luleå.
This product includes software developed at Ludd, University of Luleå, Sweden and its contributors.
This product includes software developed at the Information Technology Division, US Naval Research Laboratory.
This product includes software developed by Aaron Brown and Harvard University.
This product includes software developed by Adam Ciarcinski for the NetBSD project.
This product includes software developed by Adam Glass.
This product includes software developed by Adam Glass and Charles M. Hannum.
This product includes software developed by Alex Zepeda.
This product includes software developed by Alex Zepeda, and Colin Wood for the NetBSD Project.
This product includes software developed by Allen Briggs.
This product includes software developed by Amancio Hasty and Roger Hardiman
This product includes software developed by Ben Gray.
This product includes software developed by Berkeley Software Design, Inc.
This product includes software developed by Bill Paul.
This product includes software developed by Bodo Möller.
This product includes software developed by Boris Popov.
This product includes software developed by Brini.
This product includes software developed by Bruce M. Simpson.
This product includes software developed by Causality Limited.
This product includes software developed by Charles Hannum.
This product includes software developed by Charles M. Hannum.
This product includes software developed by Charles M. Hannum, by the University of Vermont and State Agricultural College and Garrett A. Wollman, by William F. Jolitz, and by the University of California, Berkeley, Lawrence Berkeley Laboratory, and its contributors.
This product includes software developed by Christian E. Hopps.
This product includes software developed by Christian E. Hopps, Ezra Story, Kari Mettinen, Markus Wild, Lutz Vieweg and Michael Teske.
This product includes software developed by Christopher G. Demetriou.
This product includes software developed by Christopher G. Demetriou for the NetBSD Project.
This product includes software developed by Chuck Silvers.
This product includes software developed by Cisco Systems, Inc.
This product includes software developed by Colin Wood.
This product includes software developed by Colin Wood for the NetBSD Project.
This product includes software developed by Computing Services at Carnegie Mellon University (http://www.cmu.edu/computing/).
This product includes software developed by Daan Vreeken.
This product includes software developed by Daishi Kato
This product includes software developed by Daniel Widenfalk and Michael L. Hitch.
This product includes software developed by Daniel Widenfalk for the NetBSD Project.
This product includes software developed by David Jones and Gordon Ross
This product includes software developed by David Miller.
This product includes software developed by Dean Huxley.
This product includes software developed by Emmanuel Dreyfus
This product includes software developed by Eric S. Hvozda.
This product includes software developed by Eric S. Raymond
This product includes software developed by Eric Young (eay@cryptsoft.com)
This product includes software developed by Eric Young (eay@mincom.oz.au)

This product includes software developed by Ezra Story.
This product includes software developed by Ezra Story and by Kari Mettinen.
This product includes software developed by Ezra Story, by Kari Mettinen and by Bernd Ernesti.
This product includes software developed by Ezra Story, by Kari Mettinen, and Michael Teske.
This product includes software developed by Ezra Story, by Kari Mettinen, Michael Teske and by Bernd Ernesti.
This product includes software developed by Frank van der Linden for the NetBSD Project.
This product includes software developed by Gardner Buchanan.
This product includes software developed by Garrett D'Amore.
This product includes software developed by Gary Thomas.
This product includes software developed by Gordon Ross
This product includes software developed by Harvard University.
This product includes software developed by Harvard University and its contributors.
This product includes software developed by Hellmuth Michaelis and Joerg Wunsch
This product includes software developed by Henrik Vestergaard Draboel.
This product includes software developed by Herb Peyerl.
This product includes software developed by Hidetoshi Shimokawa.
This product includes software developed by Hubert Feyrer for the NetBSD Project.
This product includes software developed by Ian W. Dall.
This product includes software developed by Intel Corporation and its contributors.
This product includes software developed by Internet Initiative Japan Inc.
This product includes software developed by Internet Research Institute, Inc.
This product includes software developed by James R. Maynard III.
This product includes software developed by Jared D. McNeill.
This product includes software developed by Jason L. Wright
This product includes software developed by Jason R. Thorpe for And Communications, http://www.and.com/
This product includes software developed by Joachim Koenig-Baltes.
This product includes software developed by Jochen Pohl for The NetBSD Project.
This product includes software developed by Joerg Wunsch
This product includes software developed by John Birrell.
This product includes software developed by John P. Wittkoski.
This product includes software developed by John Polstra.
This product includes software developed by Jonathan R. Stone for the NetBSD Project.
This product includes software developed by Jonathan Stone.
This product includes software developed by Jonathan Stone and Jason R. Thorpe for the NetBSD Project.
This product includes software developed by Jonathan Stone for the NetBSD Project.
This product includes software developed by Julian Highfield.
This product includes software developed by K. Kobayashi
This product includes software developed by K. Kobayashi and H. Shimokawa
This product includes software developed by Kazuhisa Shimizu.
This product includes software developed by Kazuki Sakamoto.
This product includes software developed by Kenneth Stailey.
This product includes software developed by Kiyoshi Ikehara.
This product includes software developed by Klaus Burkert,by Bernd Ernesti, by Michael van Elst, and by the University of California, Berkeley and its contributors.
This product includes software developed by Kyma Systems.
This product includes software developed by Leo Weppelman and Waldi Ravens.
This product includes software developed by Lloyd Parkes.
This product includes software developed by Lutz Vieweg.
This product includes software developed by Marc Horowitz.
This product includes software developed by Marcus Comstedt.
This product includes software developed by Mark Brinicombe.
This product includes software developed by Mark Brinicombe for the NetBSD Project.
This product includes software developed by Mark Tinguely and Jim Lowe
This product includes software developed by Markus Wild.
This product includes software developed by Marshall M. Midden.
This product includes software developed by Masanobu Saitoh.
This product includes software developed by Masaru Oki.
This product includes software developed by Matt DeBergalis
This product includes software developed by Matthew Fredette.
This product includes software developed by Michael Smith.
This product includes software developed by Microsoft
This product includes software developed by Mika Kortelainen
This product includes software developed by Mike Pritchard.
This product includes software developed by Mike Pritchard and contributors.
This product includes software developed by Minoura Makoto.
This product includes software developed by MINOURA Makoto, Takuya Harakawa.

This product includes software developed by Niels Provos.
This product includes software developed by Niklas Hallqvist.
This product includes software developed by Niklas Hallqvist, Brandon Creighton and Job de Haas.
This product includes software developed by Paolo Abeni.
This product includes software developed by Paul Kranenburg.
This product includes software developed by Paul Mackerras.
This product includes software developed by Paul Mackerras <paulus@samba.org>.
This product includes software developed by Pedro Roque Marques <pedro_m@yahoo.com>
This product includes software developed by Per Fogelstrom.
This product includes software developed by Peter Galbavy.
This product includes software developed by Phase One, Inc.
This product includes software developed by Philip A. Nelson.
This product includes software developed by QUALCOMM Incorporated.
This product includes software developed by Ravikanth.
This product includes software developed by RiscBSD.
This product includes software developed by Roar Thronæs.
This product includes software developed by Rodney W. Grimes.
This product includes software developed by Roger Hardiman
This product includes software developed by Rolf Grossmann.
This product includes software developed by Ross Harvey.
This product includes software developed by Ross Harvey for the NetBSD Project.
This product includes software developed by Scott Bartram.
This product includes software developed by Scott Stevens.
This product includes software developed by Shingo WATANABE.
This product includes software developed by Softweyr LLC, the University of California, Berkeley, and its contributors.
This product includes software developed by Stephan Thesing.
This product includes software developed by Steven M. Bellovin
This product includes software developed by Takashi Hamada.
This product includes software developed by Takumi Nakamura.
This product includes software developed by Tatoku Ogaito for the NetBSD Project.
This product includes software developed by Tommi Komulainen <Tommi.Komulainen@iki.fi>.
This product includes software developed by TooLs GmbH.
This product includes software developed by Trimble Navigation, Ltd.
This product includes software developed by Waldi Ravens.
This product includes software developed by WIDE Project and its contributors.
This product includes software developed by Winning Strategies, Inc.
This product includes software developed by Yen Yen Lim and North Dakota State University
This product includes software developed by Zembu Labs, Inc.
This product includes software developed by the Alice Group.
This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory.
This product includes software developed by the Computer Systems Laboratory at the University of Utah.
This product includes software developed by the Harvard University and its contributors.
This product includes software developed by the Kungliga Tekniska Högskolan and its contributors.
This product includes software developed by the Network Research Group at Lawrence Berkeley Laboratory.
This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.OpenSSL.org/)
This product includes software developed by the PocketBSD project and its contributors.
This product includes software developed by the RiscBSD kernel team
This product includes software developed by the RiscBSD team.
This product includes software developed by the SMCC Technology Development Group at Sun Microsystems, Inc.
This product includes software developed by the University of California, Lawrence Berkeley Laboratories.
This product includes software developed by the University of California, Lawrence Berkeley Laboratory.
This product includes software developed by the University of California, Lawrence Berkeley Laboratory and its contributors.
This product includes software developed by the University of Vermont and State Agricultural College and Garrett A. Wollman.
This product includes software developed by the University of Vermont and State Agricultural College and Garrett A. Wollman, by William F. Jolitz, and by the University of California, Berkeley, Lawrence Berkeley Laboratory, and its contributors.
This product includes software developed by the Urbana-Champaign Independent Media Center.
This product includes software developed for the FreeBSD project
This product includes software developed for the NetBSD Project by Allegro Networks, Inc., and Wasabi Systems, Inc.
This product includes software developed for the NetBSD Project by Bernd Ernesti.
This product includes software developed for the NetBSD Project by Christopher G. Demetriou.
This product includes software developed for the NetBSD Project by Eiji Kawauchi.
This product includes software developed for the NetBSD Project by Frank van der Linden
This product includes software developed for the NetBSD Project by Genetec Corporation.
This product includes software developed for the NetBSD Project by Jason R. Thorpe.
This product includes software developed for the NetBSD Project by John M. Vinopal.
This product includes software developed for the NetBSD Project by Jonathan Stone.

This product includes software developed for the NetBSD Project by Kyma Systems LLC.
This product includes software developed for the NetBSD Project by Matthias Drochner.
This product includes software developed for the NetBSD Project by Perry E. Metzger.
This product includes software developed for the NetBSD Project by Piermont Information Systems Inc.
This product includes software developed for the NetBSD Project by Shigeyuki Fukushima.
This product includes software developed for the NetBSD Project by SUNET, Swedish University Computer Network.
This product includes software developed for the NetBSD Project by Wasabi Systems, Inc.
This product includes software developed or owned by Caldera International, Inc.
This product includes software developed under OpenBSD by Per Fogelstrom.
This product includes software developed under OpenBSD by Per Fogelstrom Opsycon AB for RTMX Inc, North Carolina, USA.
This software was developed by Holger Veit and Brian Moore for use with "386BSD" and similar operating systems. "Similar operating systems" includes mainly non-profit oriented systems for research and education, including but not restricted to "NetBSD", "FreeBSD", "Mach" (by CMU).
The Institute of Electrical and Electronics Engineers and The Open Group, have given us permission to reprint portions of their documentation.

In the following statement, the phrase "this text" refers to portions of the system documentation.

Portions of this text are reprinted and reproduced in electronic form in NetBSD, from IEEE Std 1003.1, 2004 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2004 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between these versions and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document.

The original Standard can be obtained online at http://www.opengroup.org/unix/online.html.

This notice shall appear on any product containing this material.

In the following statement, "This software" refers to the parallel port driver:

> This software is a component of "386BSD" developed by William F. Jolitz, TeleMuse.

Some files have the following copyright:

> Mach Operating System
> Copyright (c) 1991,1990,1989 Carnegie Mellon University
> All Rights Reserved.
>
> Permission to use, copy, modify and distribute this software and its documentation is hereby granted, provided that both the copyright notice and this permission notice appear in all copies of the software, derivative works or modified versions, and any portions thereof, and that both notices appear in supporting documentation.
>
> CARNEGIE MELLON ALLOWS FREE USE OF THIS SOFTWARE IN ITS CONDITION. CARNEGIE MELLON DISCLAIMS ANY LIABILITY OF ANY KIND FOR ANY DAMAGES WHATSOEVER RESULTING FROM THE USE OF THIS SOFTWARE.
>
> Carnegie Mellon requests users of this software to return to
> Software Distribution Coordinator or Software.Distribution@CS.CMU.EDU
> School of Computer Science
> Carnegie Mellon University
> Pittsburgh PA 15213-3890
>
> any improvements or extensions that they make and grant Carnegie the rights to redistribute these changes.

Some files have the following copyright:

> Copyright (c) 1994, 1995 Carnegie-Mellon University.
> All rights reserved.
>
> Author: Chris G. Demetriou
>
> Permission to use, copy, modify and distribute this software and its documentation is hereby granted, provided that both the copyright notice and this permission notice appear in all copies of the software, derivative works or modified versions, and any portions thereof, and that both notices appear in supporting documentation.
> CARNEGIE MELLON ALLOWS FREE USE OF THIS SOFTWARE IN ITS "AS IS" CONDITION. CARNEGIE MELLON DISCLAIMS ANY LIABILITY OF ANY KIND FOR ANY DAMAGES WHATSOEVER RESULTING FROM THE USE OF THIS SOFTWARE.
>
> Carnegie Mellon requests users of this software to return to
> Software Distribution Coordinator or Software.Distribution@CS.CMU.EDU
> School of Computer Science
> Carnegie Mellon University
> Pittsburgh PA 15213-3890
>
> any improvements or extensions that they make and grant Carnegie the rights to redistribute these changes.

Some files have the following copyright:

The End