NAME

INSTALL – Installation procedure for NetBSD/mvme68k

DESCRIPTION

About this Document

This document describes the installation procedure for NetBSD 1.4.3 on the *mvme68k* 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.

What is NetBSD?

The NetBSD Operating System is a fully functional Open Source UN*X-like operating system derived from the Berkeley Networking Release 2 (Net/2), 4.4BSD-Lite, and 4.4BSD-Lite2 sources. NetBSD runs on twenty different system architectures featuring eight distinct families of CPUs, and is being ported to more. The NetBSD 1.4.3 release contains complete binary releases for fifteen different machine types. (The remaining ones are not fully supported at this time and are thus not part of the binary distribution. For information on them, please see the NetBSD web site at **http://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, 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, it's likely that NetBSD wouldn't exist.

Upgrade path to NetBSD 1.4.3

If you are not installing your system ''from scratch'' but instead are going to upgrade an existing system already running NetBSD you need to know which versions you can upgrade with NetBSD 1.4.3.

NetBSD 1.4.3 is an upgrade of NetBSD 1.4.2, 1.4.1, NetBSD 1.4 and earlier major and patch releases of NetBSD such as versions 1.3.3, 1.2 etc.

The intermediate development versions of code available on the main trunk in our CVS repository (also known as ''NetBSD-current'') from *after* the point where the release cycle for 1.4 was started are designated by version identifiers such as 1.4A, 1.4B, 1.4P etc. These identifiers do not designate releases, but indicate major changes in internal kernel APIs. Note that the kernel from NetBSD 1.4.3 can *not* be used to upgrade a system running one of those intermediate development versions. Trying to use the NetBSD 1.4.3 kernel on such a system *will* in all probability result in problems.

Please also note that it is not possible to do a direct ''version'' comparison between any of the intermediate development versions mentioned above and 1.4.3 to determine if a given feature is present or absent in 1.4.3. The development of 1.4 and the subsequent patch or ''point'' releases is done on a separate branch in the CVS repository which was created when the release cylcle for 1.4 was started, and during the release cycle of 1.4 and its patch releases, selective fixes have been imported from the main development trunk (the intent is to only import fixes with no or minor impact on the stability of the release branch). So, there are features in 1.4.3 which were not in e.g. 1.4H, but the reverse is also true.

Major Changes Between 1.4.2 and 1.4.3

The complete list of changes between NetBSD 1.4.2 and 1.4.3 can be found in the file CHANGES-1.4.3 in the top directory of the source tree. The following are the highlights only.

- A driver for the Lucent Wavelan (Orinoco) 802.11B wireless Ethernet PCMCIA card has been added, see `wi`(4).
- The PCI Cyclom-4Y and -8Y multiport serial cards are now supported.
- The Macintosh LC Ethernet Adapter is now supported.
- The on-board video on Macintosh Quadra 605 is now supported.
- Bugfixes to the Alteon Gigabit Ethernet driver, `ti`(4).
- Support for several new PCI IDE controllers have been added, see `pciide`(4.)
- The i386 port may now be installed on wd2 or wd3.
- A possible CPU hog problem related to large I/Os has been fixed. Fixes SA#2000-005.
- A denial-of-service problem related to handling of IP options has been fixed. Fixes SA#2000-002.
- A problem with /etc/ftpchroot has been fixed. Fixes SA#2000-006.
- A minor problem related to locking of semaphore resources has been fixed. Fixes SA#2000-004.
- The DHCP software has been updated to ISC 2.0pl3. This also fixes SA#2000-008 -- a security problem in the dhcp client code.
- A problem with use of user-supplied data as a format string in ftpd has been fixed. Fixes SA#2000-009.
- The sparc c++rt0.o code is now compiled with -fPIC. This means that libraries with global constructors (which must link in c++rt0.o on sparc/a.out) must also be compiled -fPIC.
- Most countries can now import the full release without restriction, including the previously restricted *domestic* portion. The only ''problem spots'' are US-embargoed countries. See
  http://www.NetBSD.ORG/Misc/crypto-export.html
  for the full details.
- An extensive audit of the code (and corresponding fixes) has been done to eliminate possible buffer overruns caused by possible user-specified format strings.

In addition, many bugs have been fixed -- more than 40 problems reported through our problem tracking system have been fixed, and some other non-reported problems have also been found and fixed. See the CHANGES-1.4.3 file for the complete list.

Known caveats with 1.4.3

The following are the major known issues with NetBSD 1.4.3.

- Some of the fixes in NetBSD 1.4.3 required the introduction of a new system call, `issetugid`(2). This means that you *will* have to upgrade the kernel before installing the new user-land code.
- The more strict format checking required a minor fix to the C compiler to allow NULL format strings. Thus, you will need to update your compiler before doing a ''make build'' during a source-based upgrade.
- PCMCIA support may require manual adjustment to function properly. See **Possible** PCMCIA **issues** under **Installing the NetBSD System** for more details.

- No major changes have been made to LFS in 1.4.3, so it is still of ''beta test'' quality.
- The version of the IPF packet filtering software in NetBSD 1.4.3 must be enabled with ''ipf -E'' before filtering rules can be set. Failure to do so may result in a panic. (Enabling NAT will however auto-enable IPF.)

Major Changes Between 1.4.1 and 1.4.2

The complete list of changes between NetBSD 1.4.1 and 1.4.2 can be found in the file CHANGES-1.4.2 in the top directory of the source tree. The following are the highlights only.

- A driver for the Alteon Gigabit Ethernet cards has been added, see `ti`(4).
- A driver for the Realtek 8129/8239 Fast Ethernet PCI cards have been added, see `rl`(4).
- A driver for the DPT SmartCache and SmartRAID III or IV SCSI adapters has been added, see `dpt`(4).
- A driver for the BOCA IOAT66 6-port ISA serial adapter has been added, see `ioat`(4).
- Support for the X-surf Amiga board has been added.
- Support has been added for ext2fs revision 1, with read-only support for the ''sparse super'' and the ''filetype'' options.
- BIND has been upgraded to version 8.2.2-P5.
- The IPF packet filtering software has been updated to version 3.3.6.
- Tcpdump now does hex/ascii dumps of packet contents if asked to do so.
- An implementation of the System V user management utilities has been added.
- The name ''errno'' is now always a macro which expands to a function call. This is done to ease the integration of thread libraries with the code in both system and third-party libraries. Please include <errno.h> to access the correct definition of ''errno''.
- A utility for making temporary files for shell scripts has been added, see `mktemp`(1).
- The automounter utility `amd`(8) has been updated to fix a security problem.
- A security problem in procfs has been fixed. Procfs is not used by default in NetBSD.
- The floating point emulation on the ports using the m68k CPU has been reverted to the version in NetBSD 1.4 (the version in 1.4.1 had problems).
- Several subsystems have received substantial work, such as RAIDframe, LFS, and the package tools.

In addition, many, many bugs have been fixed -- more than 100 problems reported through our problem tracking system have been fixed, and many other non-reported problems have also been found and fixed. See the CHANGES-1.4.2 file for the complete list.

Known caveats with 1.4.2

The following are the major known issues with NetBSD 1.4.2.

- The upgrade of `ipf`(8) caused a change of the kernel API. Thus, if you are using `ipf`(8) you need to upgrade both the kernel and the user-land utilities to control that feature in order for it to work.
- As part of fixing a kernel bug, ''struct vfsnode'' needed to change size. This will unfortunately, and contrary to tradition for patch releases, cause incompatibilities for users of file system LKMs compiled for 1.4 and 1.4.1.

- Although LFS has been improved between 1.4.1 and 1.4.2, it can probably still be characterized as being of ''beta test'' quality, e.g. filling the file system still gives ''interesting'' effects. It should probably not be used to store critical data quite yet.
- The version of the IPF packet filtering software in NetBSD 1.4.2 must be enabled with ''ipf -E'' before filtering or NAT rules can be set. Failure to do so may result in a panic.

Major Changes Between 1.4 and 1.4.1

The complete list of changes between NetBSD 1.4 and 1.4.1 can be found in the file CHANGES-1.4.1 in the top directory of the source tree.

- The NetBSD/alpha port's compatibility for Digital UNIX executables has been greatly improved.
- Many bug fixes and improvements of the installation tools and utilities.
- Support for more PCI serial/parallel cards has been added.
- It is now possible to boot NetBSD/i386 on systems with 1GB of RAM.
- The floating point emulation on the ports using the m68k CPU has been upgraded.
- A fatal problem with /dev/random has been found and fixed.
- Support for Alpha 21264 ev6 based systems has been added to NetBSD/alpha.

Major Changes Between 1.3 and 1.4

The NetBSD 1.4 release is a substantial improvement over its predecessors. We have provided numerous significant functional enhancements, including support for many new devices, integration of hundreds of bug fixes, new and updated kernel subsystems, and many userland enhancements. The results of these improvements is a stable operating system fit for production use that rivals most commercially available systems.

It is impossible to completely summarize the nearly two years of development that went into the NetBSD 1.4 release. Some highlights include:

- Substantial improvements in the TCP/IP implementation, including numerous performance enhancements and bug fixes by Jason Thorpe and others.
- A new, high efficiency kernel memory pool allocator by Paul Kranenburg. This has been integrated into most kernel subsystems.
- A new, totally rewritten virtual memory subsystem, UVM, created by Chuck Cranor, which is substantially cleaner and better performing than the old Mach derived VM subsystem.
- Improved POSIX and XPG standards compliance.
- Completion of the integration of all remaining 4.4BSD Lite-2 kernel improvements and bug fixes that had not been previously integrated. (Integration of all userland components was completed before NetBSD 1.3)
- Several new ports, including macppc, bebox, sparc64, next68k, and others, have been integrated into the source tree.
- The system compilers have been upgraded to egcs 1.1.1, and the system compiler toolchain now (mostly) uses the latest versions of GNU binutils instead of the obsolete versions left over from 4.4BSD Lite.
- Everyone's favorite `ftp`(1) client has been improved even further. See the man page for details.

- A new architecture independent console driver, `wscons`(4), has been integrated into many ports.
- Numerous improvements have been made to the audio subsystem support, including support for MIDI device drivers.
- Linux compatibility support has been improved.
- A number of scheduler enhancements have yielded dramatic improvements in interactive performance and better control of background tasks.
- Several network tunneling protocols, including GRE and IP in IP, have been implemented.
- Kernel support for the CODA distributed file system has been added.
- Manuel Bouyer completed major changes to the IDE support. It is now architecture independent. Major changes have been made to the IDE code for better error handling, improved ATAPI support, 32 bit data I/O support and bus-master DMA support on PCI IDE controllers.
- Lennart Augustsson has added full USB support, permitting the use of a wide variety of Universal Serial Bus peripherals. The drivers should easily port to any future platforms that support the PCI bus. See `usb`(4) for an overview.
- RAIDframe, version 1.1, from the Parallel Data Laboratory at Carnegie Mellon University, has been integrated. Supports RAID levels 0, 1, 4, 5, and more.
- Luke Mewburn added `nsswitch.conf`(5) functionality to the system to specify the search order for system databases.
- `syslogd`(8) now supports listening on multiple sockets, to make the chrooting of servers easier.
- Most third party packages have been updated to the latest stable release.

As has been noted, there have also been innumerable bug fixes.

Kernel interfaces have continued to be refined, and more subsystems and device drivers are shared among the different ports. You can look for this trend to continue.

This is the fifth public release of NetBSD for the mvme68k series of boards, and the first to support MVME167.

The Future of NetBSD

The NetBSD Foundation has been incorporated as a non-profit organization. Its purpose is to encourage, foster and promote the free exchange of computer software, namely the NetBSD Operating System. The foundation will allow for many things to be handled more smoothly than could be done with our previous informal organization. In particular, it provides the framework to deal with other parties that wish to become involved in the NetBSD Project.

The NetBSD Foundation will help improve the quality of NetBSD by:

- providing better organization to keep track of development efforts, including co-ordination with groups working in related fields.
- providing a framework to receive donations of goods and services and to own the resources necessary to run the NetBSD Project.
- providing a better position from which to undertake promotional activities.
- periodically organizing workshops for developers and other interested people to discuss ongoing work.

We intend to begin narrowing the time delay between releases. Our ambition is to provide a full release every six to eight months.

We hope to support even *more* hardware in the future, and we have a rather large number of other ideas about what can be done to improve NetBSD.

We intend to continue our current practice of making the NetBSD-current development source available on a daily basis. In addition, Anonymous CVS access to the NetBSD source tree has been added since NetBSD 1.4.1; see

http://www.netbsd.org/Changes/#anoncvs-available

We have also added a browsable CVS repository on the web at

http://cvsweb.netbsd.org/cgi-bin/cvsweb.cgi/

We intend to integrate free, positive changes from whatever sources submit them, providing that they are well thought-out and increase the usability of the system.

Above all, we hope to create a stable and accessible system, and to be responsive to the needs and desires of NetBSD users, because it is for and because of them that NetBSD exists.

Sources of NetBSD

Please refer to **http://www.netbsd.org/Sites/net.html**.

NetBSD 1.4.3 Release Contents

The root directory of the NetBSD 1.4.3 release is organized as follows:

`.../NetBSD-1.4.3/`

`BUGS` Known bugs list (somewhat incomplete and out of date).

`CHANGES` Changes since earlier NetBSD releases.

`LAST_MINUTE` Last minute changes.

`MIRRORS` A list of sites that mirror the NetBSD 1.4.3 distribution.

`README.files`
README describing the distribution's contents.

`TODO` NetBSD's todo list (also somewhat incomplete and out of date).

`patches/` Post-release source code patches.

`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 1.4.3 has a binary distribution. There are also 'README.export-control' files sprinkled liberally throughout the distribution tree, which spell out the current restrictions related to export of this code from the United States of America. Note that these regulations were recently changed so that most countries can import the entire release without significant restrictions. See

http://www.NetBSD.ORG/Misc/crypto-export.html

which contains up-to-date information on this issue.

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:

secrsrc.tgz: This set contains the "domestic" sources. These sources may be subject to United States export regulations.
423K gzipped, 2M uncompressed

gnusrc.tgz: This set contains the "gnu" sources, including the source for the compiler, assembler, groff, and the other GNU utilities in the binary distribution sets.
19M gzipped, 84.3M uncompressed

syssrc.tgz: This set contains the sources to the NetBSD 1.4.3 kernel, config(8), and dbsym(8).
13.9M gzipped, 68.8M uncompressed

sharesrc.tgz: 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.
3.1M gzipped, 11.8M uncompressed

src.tgz: This set contains all of the NetBSD 1.4.3 sources which are not mentioned above.
14.5M gzipped, 67.5M uncompressed

Most of the above source sets are located in the `source/sets` subdirectory of the distribution tree. The secrsrc.tgz set is contained in the `source/security` subdirectory. This set contains the sources normally found in `/usr/src/domestic` – primarily Kerberos (version 4) and other cryptographic security related software. This code can now be exported from the US to most countries. The document on

http://www.NetBSD.ORG/Misc/crypto-export.html

spells out the current (rather lenient) restrictions applicable for the export of this code.

The source sets are distributed as compressed tar files. They may be unpacked into `/usr/src` with the command:

```
cat set_name.tgz | gunzip | (cd /; tar xpf - )
```

The `sets/Split/` and `security/Split/` subdirectories contain split versions of the source sets for those users who need to load the source sets from floppy or otherwise need a split distribution. The split sets are are named "set_name.xx" where "set_name" is the distribution set name, and "xx" is the sequence number of the file, starting with "aa" for the first file in the distribution set, then "ab" for the next, and so on. All of these files except the last one of each set should be exactly 240,640 bytes long. (The last file is just long enough to contain the remainder of the data for that distribution set.)

The split distributions may be reassembled and extracted with **cat** as follows:

```
cat set_name.?? | gunzip | (cd /; tar xpf - )
```

In each of the source distribution set directories, there is a file named `CKSUMS` which contains the checksums of the files in that directory, as generated by the `cksum`(1) utility. You can use cksum to check the integrity of the archives, if you suspect that one of the files is corrupt and have access to a cksum binary. Checksums based on other algorithms may also be present – see the `release`(7) man page for details.

NetBSD/mvme68k Subdirectory Structure

The mvme68k-specific portion of the NetBSD 1.4.3 release is found in the `mvme68k` subdirectory of the distribution: `.../NetBSD-1.4.3/mvme68k/`

`INSTALL.html`
`INSTALL.ps`
`INSTALL.txt`
`INSTALL.more` Installation notes; this file. The `.more` file contains underlined text using the `more`(1) conventions for indicating italic and bold display.
`binary/`
`sets/` mvme68k binary distribution sets; see below.
`security/` mvme68k security distribution; see below.
`installation/`
`miniroot/` The miniroot filesystem image.
`netboot/` Two programs needed to boot mvme68k kernels over the network.
`tapeimage/` Tape boot programs, and a RAMDISK kernel.

The NetBSD/mvme68k install distribution contains files that can be used to install NetBSD onto completely "bare" MVME147 and MVME167 boards. The files in the "mvme68k/installation/*" directories are described below.

`miniroot.gz` A gzipped copy of the miniroot filesystem. This image is to be un-gzipped and copied into the swap area of a disk.

`netbsd-rd.gz` A gzipped copy of the *RAMDISK kernel* for installing the miniroot filesystem.

`stboot` A tape boot-block, in the form required to allow 1x7-Bug to boot from tape. This is the first segment of a boot tape.

`bootst` A copy of the tape boot program, used as the second segment of a boot tape.

`sboot` A copy of the MVME147 serial boot program. This is necessary if you don't have a tape drive, but you _do_ have another system which can act as a boot and NFS server. This is also useful if you are installing a diskless NetBSD system on an MVME147 board.

`netboot` A copy of the network boot program. Used in conjunction with sboot on the MVME147 to get your system booted over a network. The MVME167 can boot this file directly using 167Bug's builtin TFTP client.

These files can be used to make a boot tape suitable for installing NetBSD/mvme68k. These files can also be used to configure an NFS server to support installation "over the network". See the section **Getting the NetBSD System onto Useful Media** for instructions on either method.

Binary Distribution Sets

The NetBSD mvme68k binary distribution sets contain the binaries which comprise the NetBSD 1.4.3 release for the mvme68k. There are eight binary distribution sets and the *security* distribution set. The binary distribution sets can be found in the `mvme68k/binary/sets` subdirectory of the NetBSD 1.4.3 distribution tree, and are as follows:

base The NetBSD 1.4.3 mvme68k **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. It includes shared library support, and excludes everything described below.

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.

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. (If you are upgrading, it's recommended that you get a copy of this set and *carefully* upgrade your configuration files by hand.)

games This set includes the games and their manual pages.

kern This set contains a NetBSD/mvme68k 1.4.3 GENERIC kernel, named `/netbsd`. You *must* install this distribution set.

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 (rather large) system dictionaries, the typesettable document set, and other files from `/usr/share`.

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

The mvme68k security distribution set is named **secr** and can be found in the

`mvme68k/binary/security`

subdirectory of the NetBSD 1.4.3 distribution tree. It contains security-related binaries which depend on cryptographic source code. You do not need this distribution set to use encrypted passwords in your password file; the **base** distribution includes a crypt library which can perform only the one-way encryption function. The security distribution includes a version of the Kerberos IV network security system, and a Kerberized version of `telnet`(1) program. The **secr** distribution set can be found only on those sites which carry the complete NetBSD distribution and which can legally obtain it. Because of United States law, it may not be legal to distribute this set to locations outside of the United States and Canada. See

http://www.NetBSD.ORG/Misc/crypto-export.html

for updated information on this issue.

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 XFree86, and tightly track XFree86 releases. They are currently equivalent to XFree86 3.3.6. 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.

xcontrib Programs that were contributed to X.

xfont Fonts needed by X.

The mvme68k 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 files are /-relative and therefore are extracted *below* the current directory. That is, if you want to extract the binaries into your system, i.e. replace the system binaries with them, you have to run the **`tar xfp`** command from /.

Note Each directory in the mvme68k binary distribution also has its own checksum files, just as the source distribution does:

All BSDSUM files are historic BSD checksums for the various files in that directory, in the format produced by the command: **`cksum -o 1`** *`file`*

All CKSUM files are POSIX checksums for the various files in that directory, in the format produced by the command: **`cksum`** *`file`*.

All MD5 files are MD5 digests for the various files in that directory, in the format produced by the command: **`cksum -m`** *`file`*.

All SYSVSUM files are historic AT&T System V UNIX checksums for the various files in that directory, in the format produced by the command: **`cksum -o -2`** *`file`*.

The MD5 digest is the safest checksum, followed by the POSIX checksum. The other two checksums are provided only to ensure that the widest possible range of system can check the integrity of the release files.

NetBSD/mvme68k System Requirements and Supported Devices

NetBSD/mvme68k 1.4.3 runs on Motorola MVME147 and MVME167 Single Board Computers.

The minimal configuration requires 4M of RAM and ~100MB of disk space. To install the entire system requires much more disk space (approx. 200MB additional space is necessary for full sources. Double that if you want to recompile it all!). To run X (clients only) or compile the system, more RAM is recommended. Good performance requires 8MB of RAM, or 16 MB when running the X Window System.

Here is a table of recommended HD partition sizes for a full install:

```
partition:      advise          needed
root (/)        20M             12M
user (/usr)     100M            80M
swap            (2 or 3 * RAM)   8M
Anything else is up to you!
```

Note that the mvme68k installation procedure uses a *miniroot* image which is placed into the swap area of the disk. The swap partition must be large enough to hold this miniroot image (> 6Mb).

The following VME147 hardware is supported:

Serial ports (RS232):
built-in console, ttyZ1, ttyZ2 and ttyZ3

Parallel port:
On-board centronics style printer port

Network interfaces:
On-board Lance Ethernet (le)

SCSI: (Most SCSI disks, tapes, CD-ROMs, etc.) On-board wd33c93 SCSI bus interface chip (async only for now).

Miscellaneous:
Battery-backed real-time clock. VMEbus RAM cards.

The following VME167 hardware is supported:

Serial ports (RS232):
built-in console, ttyC1, ttyC2 and ttyC3

Parallel port:
On-board centronics style printer port

Network interfaces:
On-board Ethernet (ie)

SCSI: (Most SCSI disks, tapes, CD-ROMs, etc.) On-board ncr53c710 SCSI I/O Processor chip.

Miscellaneous:
Battery-backed real-time clock.

If it's not on the above lists, there is no support for it in this release.

Getting the NetBSD System on to Useful Media

Installation is supported from several media types, including:

- Tape
- NFS
- CD-ROM
- FTP

Note that installing on a "bare" machine requires either a bootable tape drive or an ethernet to a compatible NFS server. MVME147 may also need to be booted over an RS232 connection.

The procedure for transferring the distribution sets onto installation media depends on the type of media. Instructions for each type of media are given below.

In order to create installation media, you will need all the files and subdirectories in these two directories:

```
.../NetBSD-1.4.3/mvme68k/installation
.../NetBSD-1.4.3/mvme68k/binary
```

Creating boot/install tapes:

Installing from tape is the simplest method of all. This method uses two tapes, one containing a bootable ramdisk and miniroot, the other containing the installation sets.

The boot tape is created as follows:

```
cd .../NetBSD-1.4.3/mvme68k/installation
set T = /dev/nrst0
mt -f $T rewind
dd if=tapeimage/stboot of=$T
dd if=tapeimage/bootst of=$T obs=8k conv=osync
gzip -dc tapeimage/netbsd-rd.gz | dd of=$T obs=8k conv=osync
gzip -dc miniroot/miniroot.gz  | dd of=$T obs=8k conv=osync
mt -f $T rewind
```

The installation set tape is created as follows:

```
cd .../NetBSD-1.4.3/mvme68k/binary/sets
set T = /dev/nrst0
mt -f $T rewind
foreach f (base etc comp games man misc text)
gzip -d < $f.tgz | dd of=$T bs=8k
end
mt -f $T rewind
```

If the tape does not work as expected, you may need to explicitly set the EOF mark at the end of each tape segment. Consult the tape-related manual pages on the system where the tapes are created for more details.

Boot/Install from NFS server:

If your machine has a disk and network connection, but no tape drive, it may be convenient for you to install NetBSD over the network. This involves temporarily booting your machine over NFS, just long enough so you can initialize its disk. This method requires that you have access to an NFS server on your network so you can configure it to support diskless boot for your machine. Configuring the NFS server is normally a task for a system administrator, and is not trivial.

If you are using a NetBSD system as the boot-server, have a look at the `diskless`(8) manual page for guidelines on how to proceed with this. If the server runs another operating system, consult the documenta-

tion that came with it (i.e. add_client(8) on SunOS).

Booting an MVME147 from ethernet is not possible without first downloading a small bootstrap program (sboot) via RS232. See the section entitiled "Installing from NFS" for details on how to accomplish this.

sboot expects to be able to download a second stage bootstrap program via TFTP after having acquired its IP address through RARP It will look for a filename derived from the machine's IP address expressed in hexadecimal, with an extension of ".147". For example, an MVME147 with IP address 130.115.144.11 will make an TFTP request for '8273900B.147'. Normally, this file is just a symbolic link to the NetBSD/mvme68k "netboot" program, which should be located in a place where the TFTP daemon can find it (remember, many TFTP daemons run in a chroot'ed environment). The netboot program may be found in the install directory of this distribution.

The MVME167 boot ROM has code builtin to boot over ethernet from a TFTP server. You should configure it to download the same "netboot" program as is used for MVME147.

The netboot program will query a bootparamd server to find the NFS server address and path name for its root, and then load a kernel from that location. The server should have a copy of the netbsd-rd kernel in the root area for your client (no other files are needed in the client root, although it might be a convenient place to put the uncompressed miniroot image) and /etc/bootparams on the server should have an entry for your client and its root directory. Note that you should rename the netbsd-rd kernel to just 'netbsd' in the client's root directory before trying to netboot the client.

The client will need access to the miniroot image, which can be provided using NFS or remote shell. If using NFS, miniroot.gz should be expanded on the server, because doing so from the RAMDISK shell is not so easy. The unzipped miniroot takes about 6Mb of space.

If you will be installing NetBSD on several clients, it may be useful to know that you can use a single NFS root for all the clients as long as they only use the netbsd-rd kernel. There will be no conflict between clients because the RAM-disk kernel will not use the NFS root. No swap file is needed; the RAM-disk kernel does not use that either.

Install/Upgrade from CD-ROM:

This method requires that you boot from another device (i.e. tape or network, as described above). You may need to make a boot tape on another machine using the files provided on the CD-ROM. Once you have booted netbsd-rd (the RAMDISK kernel) and loaded the miniroot, you can load any of the distribution sets directly from the CD-ROM. The "install" program in the miniroot automates the work required to mount the CD-ROM and extract the files.

Install/Upgrade via FTP:

This method requires that you boot from another device (i.e. tape or network, as described above). You may need to make a boot tape on another machine using the files in .../install (which you get via FTP). Once you have booted netbsd-rd (the RAM-disk kernel) and loaded the miniroot, you can load any of the distribution sets over the net using FTP. The "install" program in the miniroot automates the work required to configure the network interface and transfer the files.

This method, of course, requires network access to an FTP server. This might be a local system, or it might even be ftp.NetBSD.ORG itself. If you wish to use ftp.NetBSD.ORG as your FTP file server, you may want to keep the following information handy:

IP Address: ftp.NetBSD.ORG
Login: **anonymous**
Password: *your e-mail address*
Server path: **/pub/NetBSD/NetBSD-1.4.3/mvme68k/binary**

Note If you're not using a nameserver duing installation, you might find 204.152.184.75 handy; it's the IP address of ftp.NetBSD.ORG as of January, 1999.

Preparing your System for NetBSD Installation

mvme68k machines usually need little or no preparation before installing NetBSD, other than the usual, well advised precaution of *backing up all data* on any attached storage devices.

The following instructions should make your machine "NetBSD Ready".

Power-up your MVME147 board. You should have the *bug* prompt:

```
COLD Start

Onboard  RAM start = $00000000,  stop = $007FFFFF

147-Bug>
```

Or, if you have an MVME167 board:

```
MVME167 Debugger/Diagnostics Release Version 2.3 - 02/25/94
COLD Start

Local Memory Found =02000000 (&33554432)

MPU Clock Speed =33Mhz

167-Bug>
```

Make sure the RAM size looks ok (if you've got an 8Mb MVME147 or a 32Mb MVME167 you should have the same value as I do). Also make sure the clock is ticking:

```
1x7-Bug>time
Sunday 12/21/31 16:25:14
1x7-Bug>time
Sunday 12/21/31 16:25:15
1x7-Bug>
```

Note that NetBSD bases its year at 1968, and adds the year offset in the MVME1x7's real-time clock to get the current year. So the '31' here equates to 1999. You may have to adjust your clock using the 'set' command to comply with NetBSD 's requirements. Don't worry if the "Day of the week" is not correct, NetBSD doesn't use it. Motorola has acknowledged a year 2000 bug in some versions of the MVME147 whereby the day of the week doesn't get set correctly by the 147Bug PROM. *It does not affect* NetBSD !

Also make sure that your board's ethernet address is initialised to the correct value. You'll find the address on a label on the inside of the MVME147's front panel, and on the VMEbus P2 connector of the MVME167. On the MVME147, enter the last five digits of the address using the 'lsad' command. On the MVME167, you should use the 'cnfg' command.

To install successfully to a local SCSI disk, you need to ensure that 1x7Bug is aware of what targets are connected to the SCSI bus. This can be done by issueing the following command:

```
1x7-Bug> iot;t
```

At this point, 1x7Bug will scan for any attached SCSI devices. After a short delay, a list of SCSI devices will be displayed. 147Bug will ask if LUNs should be assigned from SCSI ids, to which you should answer Y. You should also answer Y when asked if the information is to be saved to NVRAM. 167Bug does not prompt for this information.

The following installation instructions will assume that your target SCSI disk drive appears at SCSI-id 0. If you have a tape drive, the instructions assume is is configured for SCSI-id 5. When the RAMDISK root boots, NetBSD will refer to these devices as 'sd0' and 'rst0' respectively. You may wish to note these down; you'll be using them a lot. :-)

Installing the NetBSD System

Installing NetBSD is a relatively complex process, but if you have this document in hand it should not be too difficult.

There are several ways to install NetBSD onto your disk. If your machine has a tape drive the easiest way is "Installing from tape" (details below). If your machine is on a network with a suitable NFS server, then "Installing from NFS" is the next best method. Otherwise, if you have another mvme68k machine running NetBSD you can initialize the disk on that machine and then move the disk.

Installing from tape:

Create the NetBSD/mvme68k _VER boot tape as described in the section entitled "Preparing a boot tape". Then, with the tape in the drive, type the following at the 1x7Bug prompt:

```
147-Bug> bo 5
```

```
167-Bug> bo 0,50
```

As mentioned earlier, this assumes your tape is jumpered for SCSI-id 5.

As the tape loads (which may take 20 to 30 seconds), you will see a series of status messages. It may be useful if you can capture these messages to a file, or a scrollable xterm window. In particular, you should make a note of the lines which describe the geometry of the SCSI disks detected by NetBSD. They are of the form:

```
sd0 at scsibus0 targ 0 lun 0: <CDC, 94161-9, 2506> SCSI1 0/direct fixed
sd0: 148MB, 967 cyl, 9 head, 35 sec, 512 bytes/sect x 304605 sectors
```

The information of most interest is the number of sectors; here it's 304605. You will need this number when you come to create a disklabel for that drive.

```
RAM address from VMEbus = $00000000

Booting from: VME147, Controller 5, Device 0
Loading: Operating System

Volume: NBSD

IPL loaded at:  $003F0000
>> BSD MVME147 tapeboot [$Revision: 1.5.2.1 $]
578616+422344+55540+[46032+51284]=0x11a6e4
Start @ 0x8000 ...
Copyright (c) 1996, 1997 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 1.3 (RAMDISK) #1: Sun Dec 21 16:19:04 GMT 1997
    steve@soapy.mctavish.demon.co.uk:/usr/src/sys/arch/mvme68k/compile/RAMDISK
```

```
Motorola MVME-147S: 25MHz MC68030 CPU+MMU, MC68882 FPU
real mem  = 7237632
avail mem = 6381568
using 88 buffers containing 360448 bytes of memory
mainbus0 (root)
pcc0 at mainbus0: Peripheral Channel Controller, rev 0, vecbase 0x40
clock0 at pcc0 offset 0x0 ipl 5: Mostek MK48T02, 2048 bytes of NVRAM
  .
  .
```

Note The exact text of the messages will vary depending on which MVME147 or MVME167 variant you're using.

Finally, you will see the following "welcome" message:

```
        Welcome to the NetBSD/mvme68k RAMDISK root!

This environment is designed to do only four things:
  1:  Partititon your disk (use the command:  edlabel /dev/rsd0c)
  2:  Copy a miniroot image into the swap partition  (/dev/rsd0b)
  3:  Make that partition bootable (using 'installboot')
  4:  Reboot (using the swap partition, i.e. /dev/sd0b).

Copying the miniroot can be done several ways, allowing the source
of the miniroot image to be on any of these:
    boot tape,  NFS server, TFTP server, rsh server

The easiest is loading from tape, which is done as follows:
        mt -f /dev/nrst0 rewind
        mt -f /dev/nrst0 fsf 3
        dd bs=8k if=/dev/nrst0 of=/dev/rsd0b
(For help with other methods, please see the install notes.)

To reboot using the swap partition after running installboot, first
use "halt", then at the Bug monitor prompt use a command like:
               1x7Bug> bo 0,,b:

To view this message again, type:  cat /.welcome
ssh:
```

You must now create a disklabel on the disk you wish to use for the root filesystem. This will usually be `sd0`. The disklabel is used by NetBSD to identify the starting block and size of each partition on the disk.

Partitions are named `sd0a`, `sd0b`, `sd0c` etc, up to `sd0h`. The mvme68k port of NetBSD makes some assumptions about the first three partitions on a boot disk:

`sd0a` The root filesystem.
`sd0b` The swap partition.
`sd0c` The whole disk. Also known as the *raw partition*.

The *raw partition* is special; NetBSD is able to use it even if the disk has no label. You should never create a filesystem on the Raw Partition, even on a non-boot disk.

It is good practice to put `/usr` on a different partition than / (sd0a). So, the first available partition for `/usr` is `sd0d`. Refer to the section entitled **NetBSD System Requirements and Supported Devices** for information on the recommended sizes of the /, `/usr` and `swap` partitions.

You are not required to define any partitions beyond `sd0d`, but if you have a large disk drive, you might want to create several other partitions for filesystems such as `/home` or `/usr/src`. Note that at this time you are only required to partition the root/boot disk; you will get the opportunity to partition any other disks in your system from the main **miniroot** installation program.

To create the disklabel and partitions, use the `edlabel` program, passing it the name of the Raw Partition of your root/boot disk.

```
ssh: edlabel /dev/rsd0c
edlabel menu:
print - display the current disk label
modify - prompt for changes to the label
write - write the new label to disk
quit - terminate program
edlabel>
```

The program shows what commands it recognizes; *print*, *modify*, *write* and *quit*. It will accept the first letter of a command if you don't feel like typing each one in full.

To start creating the basic partitions, you should enter **m** (modify) at the edlabel prompt, then enter the letter corresponding to the first partition, **a**.

```
edlabel> m

modify subcommands:
@: modify disk parameters
a-h: modify partition
s: standarize geometry
q: quit this subcommand
edlabel/modify> a
a (root) 0 (0/00/00) 0 (0/00/00) unused
start as <blkno> or <cyls/trks/sects>: 0
length as <nblks> or <cyls/trks/sects>: 38000
type: 4.2BSD
edlabel/modify>
```

When you enter the start and length of a partition, you can use either blocks or cylinder/track/sector notation. If this is the first time you've partitioned a disk for NetBSD, it's probably easiest to use block notation. The above example creates partition 'a', starting at block zero and with a size of 38000 blocks. Note that the usual size of a block is 512 bytes, so this creates a 19Mb partition.

The 'type' of the partition should be "4.2BSD", otherwise you won't be able to create a filesystem on it.

Next, create a swap partition (b). Note that the minimum size of this swap partition should be 8Mb, otherwise you won't be able to use a miniroot to complete the NetBSD installation!

```
edlabel/modify> b
b (swap) 0 (0/00/00) 0 (0/00/00) unused
start as <blkno> or <cyls/trks/sects>: 38000
length as <nblks> or <cyls/trks/sects>: 32768
type: swap
edlabel/modify>
```

Here, we specify a value for 'start' such that the swap partition follows immediately after partition 'a', i.e. 38000. The length of the swap partition should be a multiple of the amount of RAM you have in your system. Here, I've chosen 32768, or 16Mb. The next available block on the drive is thus 38000 + 32768. We will use this to create partition 'd' for our /usr filesystem. (Note that for a busy system, or a system with more than 8Mb of RAM, you'll be better off with a 32 or 64Mb swap partition.)

```
edlabel/modify> d
d (user) 0 (0/00/00) 0 (0/00/00) unused
start as <blkno> or <cyls/trks/sects>: 70768
length as <nblks> or <cyls/trks/sects>: 233837
type: 4.2BSD
edlabel/modify> q
edlabel>
```

As you can see, I've chosen to assign the remainder of the disk to `/usr`. Since there are 304605 sectors on the example disk (did you remember to note down the number of sectors on your disk during boot?), and partition **d** starts at sector 70768, a simple bit of arithmetic (304605 - 70768) gives **d** a size of 233837.

You now need to write this new disklabel, together with the partition details you've just entered, to disk. You might also try the 'p' command to view the partitions. Once written, you can quit back to ssh using 'q'.

```
edlabel> p


        type_num: 4
        sub_type: 0
       type_name: SCSI disk
       pack_name: fictitious
    bytes/sector: 512
   sectors/track: 35
 tracks/cylinder: 9
       cylinders: 967
sectors/cylinder: 315
partition      start          (c/t/s)      nblks          (c/t/s)  type

 a (root)          0       (0/00/00)      38000     (120/05/25)* 4.2BSD
 b (swap)      38000     (120/05/25)*     32768     (104/00/08)* swap
 c (disk)          0       (0/00/00)     304605     (967/00/00)  unused
 d (user)      70768     (224/05/33)*    233837     (742/03/02)* 4.2BSD
edlabel> w
edlabel> q
ssh:
```

Now that your disk's partitioned, you need to get the proper installation miniroot image onto it. The miniroot image is designed to be copied into the swap partition of your disk. This is a safe place which won't be overwritten by the installation procedure. From the ssh prompt, use the following commands to copy the miniroot image from tape to swap (b).

```
ssh: mt -f /dev/nrst0 rewind
ssh: mt -f /dev/nrst0 fsf 3
ssh: dd bs=8k if=/dev/nrst0 of=/dev/rsd0b
```

The disk and the miniroot must now be made bootable using the `installboot`(8) command. To do this, issue the following commands:

```
ssh: mount /dev/sd0b /mnt
ssh: installboot /mnt/usr/mdec/bootsd /bootxx /dev/rsd0b
ssh: umount /dev/sd0b
```

You can now shutdown the system.

```
ssh: halt
signal 15
ssh: syncing disks... done
unmounting /mnt (/dev/sd1b)...
unmounting / (root_device)...
halted

147-Bug> reset
Reset Local SCSI Bus [Y,N] N? y
Automatic reset of known SCSI Buses on RESET [Y,N] Automatic0
Cold/Warm Reset flag [C,W] = C?
Execute Soft Reset [Y,N] N? y
```

Resetting the MVME167 board is very similar. You should now reboot from that just installed miniroot. See the section entitled "Booting the miniroot" for details.

Installing from NFS:

Before you can install from NFS, you must have already configured your NFS server to support your machine as a bootable client. Instructions for configuring the server are found in the section entitled "Getting the NetBSD System onto Useful Media" above.

To get started on the MVME147, you need to download "sboot" into RAM (you will find **sboot** in the `install` directory of the mvme68k distribution). You can either do that through the console line or through a 2nd serial connection. For example, an MVME147 connected to a sun4/110 and accessed via `tip`(1) can be loaded as follows:

```
lo 0
~Ccat sboot
go 4000
```

Which will look like this:

```
147-Bug> lo 0
~CLocal command? cat sboot
```

away for 11 seconds

```
!

147-Bug> g 4000
Effective address: 00004000

sboot: serial line bootstrap program (end = 6018)

>>>
```

Now, if you want to do it through serial line 1, then connect serial line one to a machine. At the `147-Bug>` prompt type **tm 1** You should then login to the machine it is connected to. Then hit "^A" to escape to Bug. do **lo 1;x=cat sboot** ... then when that is done you can reconnect **tm 1** and logout. Then do **go 4000** and you've got the ">>> " prompt of sboot.

Once you've got the ">>> " prompt, you can boot the RAMDISK kernel from the server:

```
>>> b


le0: ethernet address: 8:0:3e:20:cb:87
My ip address is: 192.168.1.4
Server ip address is: 192.168.1.1
4800
Download was a success!
```

See below for the next step in booting MVME147.

The MVME167 is able to download netboot directly using TFTP. To enable this, you must first configure the networking parameters on the board as described in the section entitled "Preparing your System for NetBSD Installation. On a properly configured MVME167, all you need to type is:

```
167-Bug> nbo
```

For both boards, the boot messages are very similar:

```
Start @ 0x8000 ...
>> BSD MVME147 netboot (via sboot) [$Revision: 1.5.2.1 $]
device: le0 attached to 08:00:3e:20:cb:87
boot: client IP address: 192.168.1.4
boot: client name: soapy
root addr=192.168.1.1 path=/export/soapy
578616+422344+55540+[46032+51284]=0x11a6e4
Start @ 0x8000 ...
Copyright (c) 1996, 1997 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 1.3 (RAMDISK) #1: Sun Dec 21 16:19:04 GMT 1997
    steve@soapy.mctavish.demon.co.uk:/usr/src/sys/arch/mvme68k/compile/RAMDISK
Motorola MVME-147S: 25MHz MC68030 CPU+MMU, MC68882 FPU
real mem  = 7237632
avail mem = 6381568
using 88 buffers containing 360448 bytes of memory
mainbus0 (root)
pcc0 at mainbus0: Peripheral Channel Controller, rev 0, vecbase 0x40
clock0 at pcc0 offset 0x0 ipl 5: Mostek MK48T02, 2048 bytes of NVRAM
  .
  .
```

After the boot program loads the RAMDISK kernel, you should see the welcome screen as shown in the "tape boot" section above.

You now need to create a disklabel with partition information on the SCSI disk on which you intend to create your root filesystem. Follow the instructions in the previous section entitled "Installing from tape" to do this. (But stop short of the part which describes how to copy the miniroot from tape.)

You must now configure the network interface before you can access the NFS server containing the miniroot image. For example the command:

```
ssh: ifconfig le0 inet 192.168.1.4 up
```

will bring up the MVME147 network interface 'le0' with that address. The command:

```
ssh: ifconfig ie0 inet 192.168.1.4 up
```

will bring up the MVME167 network interface 'ie0' with that address. The next step is to copy the miniroot from your server. This can be done using either NFS or remote shell. (In the examples that follow, the server has IP address 192.168.1.1) You may then need to add a default route if the server is on a different subnet:

```
ssh: route add default 192.168.1.2 1
```

You can look at the route table using:

```
ssh: route show
```

Now mount the NFS filesystem containing the miniroot image:

```
ssh: mount -r 192.168.1.1:/export/soapy /mnt
```

The procedure is simpler if you have space for an expanded (not compressed) copy of the miniroot image. In that case:

```
ssh: dd bs=8k if=/mnt/miniroot of=/dev/rsd0b
```

Otherwise, you will need to use "zcat" to expand the miniroot image while copying. This is tricky because the "ssh" program (small shell) does not handle sh(1) pipeline syntax. Instead, you first run the reader in the background with its input set to /dev/pipe and then run the other program in the foreground with its output to /dev/pipe. The result looks like this:

```
ssh: run -bg dd obs=8k if=/dev/pipe of=/dev/rsd0b
ssh: run -o /dev/pipe zcat /mnt/install/miniroot.gz
```

To load the miniroot using rsh to the server, you would use a pair of commands similar to the above. Here is another example:

```
ssh: run -b dd obs=8k if=/dev/pipe of=/dev/rsd0b
ssh: run -o /dev/pipe rsh 192.168.1.1 zcat miniroot.gz
```

You must now make the disk bootable. Refer to the previous section on installing from tape, where it describes how to run 'installboot'. This is immediately following the part which explains how to copy the miniroot from tape.

Booting the miniroot:

Assuming the miniroot is installed on partition 'b' of the disk with SCSI-id 0, then the 147Bug boot command is:

```
147-Bug> bo 0,,b:
```

The corresponding 167Bug boot command is:

```
167-Bug> bo 0,,,b:
```

The command line parameters above are:

0 controller (usually zero)

,, or ,,,
bug argument separators

b: tell the bootstrap code to boot from partition **b**

You should see a bunch of boot messages, followed by messages from the miniroot kernel just as you did when the RAMDISK kernel booted.

You will then be prompted to enter the root device. Since the miniroot was booted from the swap partition, you should enter 'sd0b'. You will then be asked for the swap device and filesystem type. Just press return twice to accept the defaults. When asked to enter a terminal type, either accept the default, or use whatever the TERM environment variable is set to in the shell of your host system:

```
boot device: sd0
root device (default sd0a): sd0b
dump device (default sd0b): (return)
file system (default generic): (return)
root on sd0b dumps on sd0b
mountroot: trying ffs...
root file system type: ffs
init: copying out path '/sbin/init' 11
erase ^H, werase ^W, kill ^U, intr ^C
Terminal type? [vt100] return
```

Congratulations! The system should now be running the miniroot installation program.

Miniroot install program:

The miniroot's install program is very simple to use. It will guide you through the entire process, and is well automated. Additional improvements are planned for future releases.

The miniroot's install program will:

- Allow you to place disklabels on additional disks. The disk we are installing on should already have been partitioned using the RAMDISK kernel.

  Note that partition sizes and offsets are expressed in sectors. When you fill out the disklabel, you will need to specify partition types and filesystem parameters. If you're unsure what the these values should be, use the following defaults:

  ```
  fstype: 4.2BSD
  fsize: 1024
  bsize: 4096
  cpg: 16
  ```

  If the partition will be a swap partition, use the following:

  ```
  fstype: swap
  fsize: 0 (or blank)
  bsize: 0 (or blank)
  cpg: 0 (or blank)
  ```

  Note that partition 'c' is special; it covers then entire disk and should not be assigned to a filesystem.

  The number of partitions is currently fixed at 8.

- Create filesystems on target partitions.
- Allow you to set up your system's network configuration. Remember to specify host names without the domain name appended to the end. For example use 'foo' instead of 'foo.bar.org'. If, during the process of configuring the network interfaces, you make a mistake, you will be able to re-configure that interface by simply selecting it for configuration again.
- Mount target filesystems. You will be given the opportunity to manually edit the resulting /etc/fstab.
- Extract binary sets from the media of your choice.

- Copy configuration information gathered during the installation process to your root filesystem.
- Make device nodes in your root filesystem.
- Copy a new kernel onto your root partition.
- Install a new boot block.
- Check your filesystems for integrity.

First-time installation on a system through a method other than the installation program is possible, but strongly discouraged.

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 propperly configured state, with the most important ones described below.

1. Configuring `/etc/rc.conf`

   If you haven't done any configuration of `/etc/rc.conf`, 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 filesystem mounted read-write. When the system asks you to choose a shell, simply hit return to get to a prompt. If you are asked for a terminal type, respond with **`vt220`** (or whatever is appropriate for your terminal type) and hit return. At this point, you need to configure at least one file in the `/etc` directory. 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. If your `/usr` directory is on a separate partition and you do not know how to use 'ed' or 'ex', you will have to mount your `/usr` partition to gain access to 'vi'. Do the following:

   ```
   mount /usr
   export TERM=vt220
   ```

   If you have `/var` on a seperate partition, you need to repeat that step for it. After that, you can edit `/etc/rc.conf` with `vi`(1). When you have finished, type **`exit`** at the prompt to leave the single-user shell and continue with the multi-user boot.

   Other values that need to be set in `/etc/rc.conf` for a networked environment are *`hostname`* and possibly *`defaultroute`*, furthermore add an *`ifconfig_int`* for your interface <int>, along the lines of

   ```
   ifconfig_de0="inet 123.45.67.89 netmask 255.255.255.0"
   ```

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

   ```
   ifconfig_de0="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.

   Other files in `/etc` that are new to NetBSD 1.4 and may require modification or setting up include `/etc/mailer.conf`, `/etc/nsswitch.conf` and `/etc/wscons.conf`.

2. Logging in

   After reboot, you can log in as `root` at the login prompt. There is no initial password, but if you're using the machine in a networked environment, you should create an account for yourself (see below) and protect it and the "root" account with good passwords.

3. Adding accounts

   Use the `vipw`(8) command to add accounts to your system, *do not* edit `/etc/passwd` directly. See `adduser`(8) for more information on the process of how to add a new user to the system.

4. The X Window System

   If you have installed the X window system, look at the files in `/usr/X11R6/lib/X11/doc` for information.

   Don't forget to add `/usr/X11R6/bin` to your path in your shell's dot file so that you have access to the X binaries.

5. Installing 3rd party packages

   There is a lot of software freely available for Unix-based systems, almost all of which can run on NetBSD. Modifications are usually needed to when transferring programs between different Unix-like systems, so the NetBSD packages collection incorporates any such changes necessary to make that software run on NetBSD, and makes the installation (and deinstallation) of the software packages easy. There's also the option of building a package from source, in case there's no precompiled binary available.

   Precompiled binaries can be found at
   ftp://ftp.netbsd.org/pub/NetBSD/packages/
   Package sources for compiling packages can be obtained by retrieving the file
   ftp://ftp.netbsd.org/pub/NetBSD/NetBSD-current/tar_files/pkgsrc.tar.gz
   They are typically extracted into `/usr/pkgsrc` (though other locations work fine), as with the command:

   ```
   cat pkgsrc.tar.gz | gunzip | (mkdir /usr/pkgsrc;
   cd /usr/pkgsrc; tar xpf - )
   ```

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

6. Misc

   - To adjust the system to your local timezone, point the `/etc/localtime` symlink to the appropriate file under `/usr/share/zoneinfo`.
   - Edit `/etc/aliases` to forward root mail to the right place (run `newaliases`(1) afterwards.)
   - The `/etc/sendmail.cf` file will almost definitely need to be adjusted; files aiding in this can be found in `/usr/share/sendmail`. See the README file there for more information.
   - 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 *filename*
     is likely to give you more information on these files.

Upgrading a previously-installed NetBSD System

It is possible to easily upgrade your existing NetBSD/mvme68k system using the upgrade program in the miniroot. If you wish to upgrade your system by this method, simply select the 'upgrade' option once the miniroot has booted. The upgrade program with then guide you through the procedure. The upgrade program will:

- Enable the network based on your system's current network configuration.

- Mount your existing filesystems.
- Extract binary sets from the media of your choice.
- Fixup your system's existing /etc/fstab, and if necessary, changing the occurrences of 'ufs' to 'ffs' and let you edit the resulting file.
- Make new device nodes in your root filesystem.
- Copy a new kernel onto your root partition. NOTE: the existing kernel WILL NOT be backed up; doing so would be pointless, since older kernels may not be capable of running NetBSD 1.4.3 executables.
- Install a new boot block.
- Check your filesystems for integrity.

While using the miniroot's upgrade program is the preferred method of upgrading your system, it is possible to upgrade your system manually. To do this, follow the following procedure:

- Place _at least_ the 'base' binary set in a filesystem accessible to the target machine. A local filesystem is preferred, since the NFS subsystem in the NetBSD 1.4.3 kernel may be incompatible with your old binaries.
- Back up your pre-existing kernel and copy the _VER kernel into your root partition.
- Reboot with the _VER kernel into single-user mode.
- Check all filesystems:

  /sbin/fsck -pf

- Mount all local filesystems:

  /sbin/mount -a -t nonfs

- If you keep /usr or /usr/share on an NFS server, you will want to mount those filesystems as well. To do this, you will need to enable the network:

  sh /etc/netstart

- Run the `update`(8) daemon, to ensure that new programs are actually stored on disk.

  update

- Make sure you are in the root filesystem and extract the 'base' binary set:

  cd /
  pax -zrvpe -f */path/to/***base.tgz**

- Delete system files in /sbin which now live in /usr/sbin:

  cd /sbin
  rm -f mountd rtquery quotacheck dumpfs dumplfs

- Install a new boot block (you may need to be running in single-user mode to do this...):

  cd /usr/mdec
  cp bootsd /.bootsd
  ./installboot /.bootsd bootxx < *root-disk*

  (*root-disk* might be, e.g., **/dev/rsd0a**)

- Sync the filesystems:

  sync

- At this point you may extract any other binary sets you may have placed on local filesystems, or you may wish to extract additional sets at a later time. To extract these sets, use the following commands:

  cd /
  pax -zrvpe -f *path_to_set*

Note You *SHOULD NOT* extract the **etc** set if upgrading. Instead, you should extract that set into another area and carefully merge the changes by hand.

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 1.4.3

Note Only issues effecting an upgrade from NetBSD 1.3 or NetBSD 1.3.x are decribed here.

- "machine" directory/link in "/usr/include"

  Description
  Some architecture may fail to install the **comp** set because the
  `/usr/include/machine`
  directory changed to a symbolic link in NetBSD 1.4.

  Fix If this happens, you can use the command
  # **rm -r /usr/include/machine**
  to remove the old directory and it contents and reinstall the **comp** set.

Using online NetBSD documentation

Documentation is available if you first install 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)m 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. To get help on using the mailing list server, send mail to that address with an empty body, and it will reply with instructions.

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. Additionally, bug reports can be sent by mail to: netbsd-bugs@NetBSD.ORG.

Use of send-pr(1) is encouraged, however, because bugs reported with it are entered into the NetBSD bugs database, and thus can't slip through the cracks.

There are also port-specific mailing lists, to discuss aspects of each port of NetBSD. Use majordomo to find their addresses. 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 this effort, and have an idea as to how you could be useful, send us mail or subscribe to: netbsd-help@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 somewhere, then mail the appropriate list about it, or, 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 ongoing work on BSD systems, support, and encouragement.

- Also, our thanks go to:

  Mike Hibler
  Rick Macklem
  Jan-Simon Pendry
  Chris Torek

  for answering lots of questions, fixing bugs, and doing the various work they've done.

- UC Berkeley's Experimental Computing Facility provided a home for sun-lamp in the past, people to look after it, and a sense of humor. Rob Robertson, too, has added his unique sense of humor to things, and for a long time provided the primary FTP site for NetBSD.

- Vixie Enterprises for hosting the NetBSD FTP, WWW and SUP server.

- Redback Networks, Inc. for hosting the NetBSD Mail server.

- 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.

- Dave Burgess burgess@cynjut.infonet.net has been maintaining the 386BSD/NetBSD/FreeBSD FAQ for quite some time, and deserves to be recognized for it.

- The following individuals and organiztions (each in alphabetical order) have made donations or loans of hardware and/or money, to support NetBSD development, and deserve credit for it:

  Steve Allen
  Jason Birnschein
  Mason Loring Bliss
  Jason Brazile
  Mark Brinicombe
  David Brownlee
  Simon Burge
  Dave Burgess
  Ralph Campbell
  Brian Carlstrom
  James Chacon
  Bill Coldwell
  Charles Conn
  Tom Coulter
  Charles D. Cranor
  Christopher G. Demetriou
  Scott Ellis
  Hubert Feyrer
  Castor Fu
  Greg Gingerich
  William Gnadt
  Michael Graff
  Guenther Grau
  Ross Harvey
  Charles M. Hannum
  Michael L. Hitch
  Kenneth Alan Hornstein
  Jordan K. Hubbard
  Soren Jorvang
  Scott Kaplan
  Noah M. Keiserman
  John Kohl
  Chris Legrow
  Ted Lemon
  Neil J. McRae
  Perry E. Metzger
  Herb Peyerl
  Mike Price
  Dave Rand
  Michael Richardson
  Heiko W. Rupp
  Brad Salai
  Chuck Silvers
  Thor Lancelot Simon
  Bill Sommerfeld
  Paul Southworth
  Ted Spradley
  Kimmo Suominen
  Jason R. Thorpe

Steve Wadlow
Krister Walfridsson
Jim Wise
Christos Zoulas

AboveNet Communications, Inc.
Advanced System Products, Inc.
Avalon Computer Systems
Bay Area Internet Solutions
Brains Corporation, Japan
Canada Connect Corporation
Co-operative Research Centre for Enterprise Distributed Systems Technology
Demon Internet, UK
Digital Equipment Corporation
Easynet, UK
Free Hardware Foundation
Innovation Development Enterprises of America
Internet Software Consortium
MS Macro System GmbH, Germany
Numerical Aerospace Simulation Facility, NASA Ames Research Center
Piermont Information Systems Inc.
VMC Harald Frank, Germany
Warped Communications, Inc.

(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 mentioned, tell us!)

We are...

(in alphabetical order)

The NetBSD core group:

Alistair Crooks	`agc@NetBSD.ORG`
Jun-ichiro itojun Hagino	`itojun@NetBSD.ORG`
Frank van der Linden	`fvdl@NetBSD.ORG`
Luke Mewburn	`lukem@NetBSD.ORG`
Christos Zoulas	`christos@NetBSD.ORG`

The portmasters (and their ports):

Mark Brinicombe	mark@NetBSD.ORG	**arm32**
Jeremy Cooper	jeremy@NetBSD.ORG	**sun3x**
Ross Harvey	ross@NetBSD.ORG	**alpha**
Ignatios Souvatzis	is@NetBSD.ORG	**amiga**
Jun-ichiro itojun Hagino	itojun@NetBSD.ORG	**sh3**
Eduardo Horvath	eeh@NetBSD.ORG	**sparc64**
Paul Kranenburg	pk@NetBSD.ORG	**sparc**
Anders Magnusson	ragge@NetBSD.ORG	**vax**
Tsubai Masanari	tsubai@NetBSD.ORG	**macppc**
Tsubai Masanari	tsubai@NetBSD.ORG	**newsmips**
Minoura Makoto	minoura@NetBSD.ORG	**x68k**
Phil Nelson	phil@NetBSD.ORG	**pc532**
Scott Reynolds	scottr@NetBSD.ORG	**mac68k**
Darrin Jewell	dbj@NetBSD.ORG	**next68k**
Kazuki Sakamoto	sakamoto@NetBSD.ORG	**bebox**
Wolfgang Solfrank	ws@NetBSD.ORG	**ofppc**
Jonathan Stone	jonathan@NetBSD.ORG	**pmax**
Shin Takemura	takemura@NetBSD.ORG	**hpcmips**
Jason Thorpe	thorpej@NetBSD.ORG	**hp300**
Frank van der Linden	fvdl@NetBSD.ORG	**i386**
Leo Weppelman	leo@NetBSD.ORG	**atari**
Nathan Williams	nathanw@NetBSD.ORG	**sun3**
Steve Woodford	scw@NetBSD.ORG	**mvme68k**

The NetBSD 1.4.3 Release Engineering team:

Chris G. Demetriou	cgd@NetBSD.ORG
Havard Eidnes	he@NetBSD.ORG
Ted Lemon	mellon@NetBSD.ORG
Perry Metzger	perry@NetBSD.ORG
Curt Sampson	cjs@NetBSD.ORG

Developers and other contributors:

Steve Allen	wormey@NetBSD.ORG
Julian Assange	proff@NetBSD.ORG
Lennart Augustsson	augustss@NetBSD.ORG
Christoph Badura	bad@NetBSD.ORG
Robert V. Baron	rvb@NetBSD.ORG
Erik Berls	cyber@NetBSD.ORG
John Birrell	jb@NetBSD.ORG
Mason Loring Bliss	mason@NetBSD.ORG
Manuel Bouyer	bouyer@NetBSD.ORG
John Brezak	brezak@NetBSD.ORG
Allen Briggs	briggs@NetBSD.ORG
Aaron Brown	abrown@NetBSD.ORG
David Brownlee	abs@NetBSD.ORG

Frederick Bruckman	fredb@NetBSD.ORG
Jon Buller	jonb@NetBSD.ORG
Simon Burge	simonb@NetBSD.ORG
Dave Burgess	burgess@cynjut.infonet.net
Robert Byrnes	byrnes@NetBSD.org
D'Arcy J.M. Cain	darcy@NetBSD.ORG
Dave Carrel	carrel@NetBSD.ORG
Bill Coldwell	billc@NetBSD.ORG
Julian Coleman	jdc@NetBSD.ORG
Chuck Cranor	chuck@NetBSD.ORG
Aidan Cully	aidan@NetBSD.ORG
Johan Danielsson	joda@NetBSD.ORG
Matt DeBergalis	deberg@NetBSD.ORG
Rob Deker	deker@NetBSD.ORG
Chris G. Demetriou	cgd@NetBSD.ORG
Jaromir Dolecek	jdolecek@NetBSD.ORG
Andy Doran	ad@NetBSD.ORG
Roland Dowdeswell	elric@NetBSD.ORG
Matthias Drochner	drochner@NetBSD.ORG
Jun Ebihara	jun@NetBSD.ORG
Havard Eidnes	he@NetBSD.ORG
Enami Tsugutomo	enami@NetBSD.ORG
Bernd Ernesti	veego@NetBSD.ORG
Erik Fair	fair@NetBSD.ORG
Hubert Feyrer	hubertf@NetBSD.ORG
Thorsten Frueauf	frueauf@NetBSD.ORG
Castor Fu	castor@NetBSD.ORG
Ichiro Fukuhara	ichiro@NetBSD.ORG
Brian R. Gaeke	brg@dgate.org
Thomas Gerner	thomas@NetBSD.ORG
Simon J. Gerraty	sjg@NetBSD.ORG
Justin Gibbs	gibbs@NetBSD.ORG
Adam Glass	glass@NetBSD.ORG
Michael Graff	explorer@NetBSD.ORG
Brian C. Grayson	bgrayson@NetBSD.ORG
Brad Grantham	grantham@tenon.com
Matthew Green	mrg@NetBSD.ORG
Juergen Hannken-Illjes	hannken@NetBSD.ORG
Charles M. Hannum	mycroft@NetBSD.ORG
Eric Haszlakiewicz	erh@NetBSD.ORG
HAYAKAWA Koichi	haya@NetBSD.ORG
Rene Hexel	rh@NetBSD.ORG
Michael L. Hitch	mhitch@NetBSD.ORG
Christian E. Hopps	chopps@NetBSD.ORG
Ken Hornstein	kenh@NetBSD.ORG
Marc Horowitz	marc@NetBSD.ORG
Dean Huxley	dean@netbsd.org
ITOH Yasufumi	itohy@NetBSD.ORG
IWAMOTO Toshihiro	toshii@NetBSD.ORG

Matthew Jacob	mjacob@NetBSD.ORG
Lonhyn T. Jasinskyj	lonhyn@NetBSD.ORG
Chris Jones	cjones@NetBSD.ORG
Soren Jorvang	soren@NetBSD.ORG
Takahiro Kambe	taca@NetBSD.ORG
Antti Kantee	pooka@NetBSD.ORG
Lawrence Kesteloot	kesteloo@cs.unc.edu
Thomas Klausner	wiz@NetBSD.ORG
Klaus Klein	kleink@NetBSD.ORG
John Kohl	jtk@NetBSD.ORG
Kevin Lahey	kml@NetBSD.ORG
Johnny C. Lam	jlam@NetBSD.ORG
Martin J. Laubach	mjl@NetBSD.ORG
Ted Lemon	mellon@NetBSD.ORG
Joel Lindholm	joel@NetBSD.ORG
Mike Long	mikel@NetBSD.ORG
Warner Losh	imp@NetBSD.ORG
Brett Lymn	blymn@NetBSD.ORG
Paul Mackerras	paulus@NetBSD.ORG
Dan McMahill	dmcmahill@NetBSD.ORG
Neil J. McRae	neil@NetBSD.ORG
Perry Metzger	perry@NetBSD.ORG
der Mouse	mouse@NetBSD.ORG
Joseph Myers	jsm@NetBSD.ORG
Ken Nakata	kenn@NetBSD.ORG
Bob Nestor	rnestor@NetBSD.ORG
Tohru Nishimura	nisimura@NetBSD.ORG
Masaru Oki	oki@NetBSD.ORG
Greg Oster	oster@NetBSD.ORG
Herb Peyerl	hpeyerl@NetBSD.ORG
Matthias Pfaller	matthias@NetBSD.ORG
Dante Profeta	dante@NetBSD.ORG
Chris Provenzano	proven@NetBSD.ORG
Waldi Ravens	waldi@moacs.indiv.nl.net
Darren Reed	darrenr@NetBSD.ORG
Michael Richardson	mcr@NetBSD.ORG
Tim Rightnour	garbled@NetBSD.ORG
Gordon Ross	gwr@NetBSD.ORG
Heiko W. Rupp	hwr@NetBSD.ORG
SAITOH Masanobu	msaitoh@NetBSD.ORG
Curt Sampson	cjs@NetBSD.ORG
Wilfredo Sanchez	wsanchez@NetBSD.ORG
Ty Sarna	tsarna@NetBSD.ORG
SATO Kazumi	sato@NetBSD.ORG
Matthias Scheler	tron@NetBSD.ORG
Karl Schilke (rAT)	rat@NetBSD.ORG
Konrad Schroder	perseant@NetBSD.ORG
Tim Shepard	shep@NetBSD.ORG
Takao Shinohara	shin@NetBSD.ORG

Takuya SHIOZAKI	tshiozak@NetBSD.ORG
Chuck Silvers	chs@NetBSD.ORG
Thor Lancelot Simon	tls@NetBSD.ORG
Noriyuki Soda	soda@NetBSD.ORG
Bill Sommerfeld	sommerfeld@NetBSD.ORG
Bill Studenmund	wrstuden@NetBSD.ORG
Kevin Sullivan	sullivan@NetBSD.ORG
SUNAGAWA Keiki	kei@NetBSD.ORG
Kimmo Suominen	kim@NetBSD.ORG
Matt Thomas	matt@NetBSD.ORG
Christoph Toshok	toshok@NetBSD.ORG
Izumi Tsutsui	tsutsui@NetBSD.ORG
UCHIYAMA Yasushi	uch@NetBSD.ORG
Todd Vierling	tv@NetBSD.ORG
Aymeric Vincent	aymeric@NetBSD.ORG
Paul Vixie	vixie@NetBSD.ORG
Krister Walfridsson	kristerw@NetBSD.ORG
Lex Wennmacher	wennmach@NetBSD.ORG
Assar Westerlund	assar@NetBSD.ORG
Rob Windsor	windsor@NetBSD.ORG
Dan Winship	danw@NetBSD.ORG
Jim Wise	jwise@NetBSD.ORG
Colin Wood	ender@NetBSD.ORG

Legal Mumbo-Jumbo

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

This product includes software developed by the University of California, Berkeley and its contributors.

This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory.

This product includes software developed by the NetBSD Foundation, Inc. and its contributors.

This product includes software developed by Adam Glass and Charles Hannum.

This product includes software developed by Adam Glass.

This product includes software developed by Berkeley Software Design, Inc.

This product includes software developed by Charles D. Cranor and Washington University.

This product includes software developed by Charles D. Cranor.

This product includes software developed by Charles 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 Charles Hannum.

This product includes software developed by Charles M. Hannum.

This product includes software developed by Chris Provenzano.

This product includes software developed by Christian E. Hopps.

This product includes software developed by Christopher G. Demetriou for the NetBSD Project.

This product includes software developed by Christopher G. Demetriou.

This product includes software developed by Christos Zoulas.

This product includes software developed by David Jones and Gordon Ross.

This product includes software developed by Dean Huxley.

This product includes software developed by Eric S. Hvozda.

This product includes software developed by Ezra Story.

This product includes software developed by Gordon Ross.

This product includes software developed by Gordon W. Ross and Leo Weppelman.

This product includes software developed by Gordon W. Ross.

This product includes software developed by Herb Peyerl.

This product includes software developed by Ian W. Dall.

This product includes software developed by Ignatios Souvatzis for the NetBSD Project.

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 John Polstra.

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 Jonathan Stone.

This product includes software developed by Julian Highfield.

This product includes software developed by Kenneth Stailey.

This product includes software developed by Leo Weppelman.

This product includes software developed by Lloyd Parkes.

This product includes software developed by Mark Brinicombe.

This product includes software developed by Markus Wild.

This product includes software developed by Martin Husemann and Wolfgang Solfrank.

This product includes software developed by Mats O Jansson and Charles D. Cranor.

This product includes software developed by Mats O Jansson.

This product includes software developed by Matthias Pfaller.

This product includes software developed by Paul Kranenburg.

This product includes software developed by Paul Mackerras.

This product includes software developed by Peter Galbavy.

This product includes software developed by Philip A. Nelson.

This product includes software developed by Rodney W. Grimes.

This product includes software developed by Scott Bartram.

This product includes software developed by SigmaSoft, Th. Lockert.

This product includes software developed by Terrence R. Lambert.

This product includes software developed by Theo de Raadt and John Brezak.

This product includes software developed by Theo de Raadt.

This product includes software developed by TooLs GmbH.

This product includes software developed by Winning Strategies, Inc.

This product includes software developed by the Center for Software Science at the University of Utah.

This product includes software developed by the University of Calgary Department of Computer Science 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 for the FreeBSD project.

This product includes software developed for the Internet Software Consortium by Ted Lemon.

This product includes software developed for the NetBSD Project by Frank van der Linden.

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 Matthias Drochner.

This product includes software developed for the NetBSD Project by Matthieu Herrb.

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 Ted Lemon.

This product includes software developed by LAN Media Corporation and its contributors.

This product includes software developed by Michael Graff for the NetBSD Project.

This product includes software developed by Niklas Hallqvist, C Stone and Job de Haas.

This product includes software developed by the Computer Systems Laboratory at the University of Utah.

This product includes software developed by the University of Vermont and State Agricultural College and Garrett A. Wollman.

This product includes software developed by Charles D. Cranor.

This product includes software developed by Adam Glass.

This product includes software developed by Charles Hannum.

This product includes software developed by David Jones.

This product includes software developed by Gordon W. Ross.

This product includes software developed by Steve C. Woodford.