NAME

INSTALL – Installation procedure for NetBSD/mvme68k.

CONTENTS

DESCRIPTION

About this Document

This document describes the installation procedure for NetBSD 1.5.3 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 UNIX-like 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 thirty-one different system architectures featuring twelve distinct families of CPUs, and is being ported to more. The NetBSD 1.5.3 release contains complete binary releases for fifteen different machine types. (The sixteen remaining 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, 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, it's likely that NetBSD wouldn't exist.

Upgrade path to NetBSD 1.5.3

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 1.5.3.

NetBSD 1.5.3 is an upgrade of NetBSD 1.5, 1.5.1, 1.5.2, and earlier major and maintenance releases of NetBSD.

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 1.5 was started are designated by version identifiers such as 1.5A, 1.5B, etc. These identifiers do not designate releases, but indicate major changes in internal kernel APIs. Note that the kernel from NetBSD 1.5.3 can *not* be used to upgrade a system running one of those intermediate development versions. Trying to use the NetBSD 1.5.3 kernel on such a system *will* probably 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 1.5.3 to determine if a given feature is present or absent in 1.5.3. The development of 1.5 and the subsequent maintenance releases is done on a separate branch in the CVS repository. The branch was created when the release cycle for 1.5 was started, and during the release cycle of 1.5 and its maintenance releases, selected fixes and enhancements have been imported from the main development trunk. So, there are features in 1.5.3 which were not in, e.g. 1.5B, and vice versa.

Major Changes Between 1.5.2 and 1.5.3

Some highlights are:

- lpd(8) has been fixed to deal with a security issue (SA2001-018).
- Drivers for Mylex DAC960, Compaq EISA array controllers, and I2O block/SCSI devices added. This has caused ca(4) to be replaced by ld(4). Drivers for the Mylex DAC960 management interface and DPT/Adaptec I2O RAID management interface has also been added. See dpti(4) and mlxctl(8).
- A driver for the 3ware Escalade 5000 and 6000 series RAID controllers has been added, see twe(4).
- Various different fixes have been applied to the network device drivers ep(4), ex(4), rtk(4), sip(4), ti(4), tl(4), and wi(4).
- Support for some more variants of rtk(4) on CardBus has been added.
- The ne(4) driver has been extended to support some more pcmcia cards.
- Support for more pciide(4) controllers added: HPT-370A, Acard ATP-850/860, and AMD-768. Ultra-DMA 100 support added for CMD 0649.
- A problem with the NFS server code, exposed on NetBSD/alpha, where use of ‘‘..’’ would return info for ‘‘.’’ has been fixed.
- Support for ‘‘other-endian’’ file systems has been improved.
- The layout algorithm for FFS file systems has been substantially improved, resulting in better performance, both due to improved locality between files and their corresponding directory, and lessened pressure on the buffer cache.
- The performance of soft dependencies in FFS has been significantly improved in some circumstances.
- Handling of init/fini section support and DWARF2 exception handling has been added to the C runtime startup code.
- OpenSSH has been upgraded to version 3.0.2.
- A security issue arising from a race between set-uid execution and use of ptrace has been fixed (SA2002-001).
- The package tools has been extended to provide for optional digital signatures on binary packages. Additionally, binary package version number handling has been rewritten.
- A driver for Creative Labs SBLive! EMU10000 has been added; see emuxki(4).
- A driver for ESS Allegro-1 / Maestro-3 has been added; see esa(4).
- Support for the 53c1010-33 and 53c1510D has been added to the siop(4) driver.
- IPFilter upgraded to 3.4.23.
- The boot code on NetBSD/macppc has been improved.
- sendmail(8) has been upgraded to version 8.11.6.
- A long file name buffer overrun in gzip has been fixed (SA2002-002).
- IPSEC policy check has been fixed for forwarded IPv4 packets (SA2002-003).
- Two security problems in OpenSSH has been fixed (SA2002-004, SA2002-005).
- A buffer overrun in the resolver part of the C library has been fixed (SA2002-006). To fix this problem in the BIND user utilities, BIND has been upgraded to version 8.3.3.

- Various other kernel stability and robustness fixes has been applied.

In addition, many bugs have been fixed—more than 70 problems reported through our problem tracking system have been fixed, and several other non-reported problems have also been found and fixed. See the CHANGES-1.5.3 file for the complete list.

Known Caveats with 1.5.3

The following are the major known issues with NetBSD 1.5.3.

- The upgrade of IPFilter to version 3.4.23 is not backward compatible. This means that the user-land tools used to manage IPFilter will have to be upgraded when the kernel is upgraded.
- Due to xsrc changes which are not backward compatible on the arm32 port, the X bits for arm32 is built from the 1.5.2 sources.

Major Changes Between 1.5.1 and 1.5.2

The main purpose of NetBSD 1.5.2 is to fix some problems discovered shortly after the release of NetBSD 1.5.1. Some highlights are:

- `telnetd`(8) has been fixed to deal with a security issue (SA2001-012).
- A weakness in the OpenSSL libcrypto's pseudo random number generator has been fixed (SA2001-013).
- Floating point emulation on i386 (to run on FPU-less CPUs) which was broken late in 1.5.1's cycle has been restored to working order.
- Bugs in the compiler have been worked around to produce a working miniroot image on sparc64.
- The script-based installers on amiga, atari and mvme68k have been updated to deal with new behaviour from `mount`(8).
- `dump`(8) has been fixed to deal with a local security issue (SA2001-014).
- Several system calls have had defensive length checks applied on their arguments (SA2001-015).
- The DHCP software has been upgraded to ISC Version 3, Release Candidate 10.

Additionally, a few more bugs have been fixed. See the CHANGES-1.5.2 file for the complete list of changes.

Major Changes Between 1.5 and 1.5.1

The complete list of changes between NetBSD 1.5 and 1.5.1 can be found in the file CHANGES-1.5.1 in the top directory of the source tree. The following are highlights only:

- A driver for the Aironet/Cisco wireless PCMCIA cards has been added; see `an`(4).
- NFS client performance has been improved, typically by 40% for writes but possibly up to 100% in certain setups.
- The `siop`(4) driver has improved in performance and robustness.
- Support for cloning pseudo-interfaces has been added. See `ifconfig`(8).
- Support for 802.1Q virtual LANs has been added. See `vlan`(4).
- The `isp`(4) driver has been upgraded to (among other things) work on MacPPC.
- BIND has been upgraded to version 8.2.3 (SA2001-001).

- Support for booting from RAIDframe RAID1 mirrors on i386 added.
- The lfs(4) file system has again been substantially updated, but is still experimental.
- Ultra/66 support has been added for capable VIA chipsets, and Ultra/100 support has been added for the HPT370, Promise and Intel ICH2 controllers in the pciide(4) driver. Support for Intel 82801BAM controllers has also been added, and handling of Ali controllers has been improved.
- OpenSSH has been updated to deal with a security issue (SA2001-003).
- Sendmail has been upgraded to version 8.11.3.
- The ex(4) driver has added support for 3Com 3c555, 3c556 and 3c556B MiniPCI Ethernet cards.
- A driver for the on-board audio hardware found on many Apple PowerMacs has been added; see awacs(4).
- The sip(4) driver has been fixed to properly support the dp83815, as found in current Netgear FA311 10/100 cards.
- ftpd(8) has been updated to deal with two security issues (SA2000-018 and SA2001-005).
- ntpd(8) has been updated to deal with a security issue (SA2001-004).
- telnetd(8) has been updated to deal with a security issue (SA2000-017).
- A vulnerability on i386 related to USER_LDT has been fixed (SA2001-002).
- The Linux emulation has been enhanced to prepare for the support of using the Linux version of VMware.
- IP checksumming speed has been improved on i386 compared to NetBSD 1.5 by about 10%.
- Support for the Socket Communications LP-E Type II PCMCIA NE2000 clone card has been added to ne(4).
- The DHCP software has been upgraded to ISC version 3, Beta 2, patchlevel 24, to fix core dumps in dhclient(8), among other things. Please note that the new dhcpd(8) forces you to configure a "ddns-update-style" of either "ad-hoc", "interim" or "none".
- Various fixes and enhancements to INET6 and IPSEC code; among them improved interaction between IPF/Nat and IPSEC.
- The Heimdal kerberos(8) implementation has been upgraded to version 0.3e.
- Support for Accton EN2242 and other AmdTek AN985 cards added to the tlp(4) driver.
- Several country-specific keyboard mappings have been added for USB keyboards.
- A driver for Yamaha YMF724/740/744/745-based sound cards has been added, see yds(4).
- The maximum number of BSD disklabel partitions on the i386 port has been increased from 8 to 16.
- Drivers for the AC'97 based audio sound chips ESS Technology Maestro 1, 2, and 2E (see esm(4)), NeoMagic 256 (see neo(4)), and Cirrus Logic CrystalClear PCI Audio CS4281 (see clct(4)) have been added.

In addition, many bugs have been fixed—more than 95 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-1.5.1 file for the complete list.

The NetBSD Packages Collection (pkgsrc) which is used to maintain, build, track dependencies, and maintain NetBSD-specific fixes to third-party programs, has received a major overhaul for NetBSD 1.5.1. Some highlights are:

- Many new packages were added to the collection, which now includes about 2100 packages. Many of them are also available as pre-compiled binaries on ftp.netbsd.org and its mirrors. Many packages have been modified and enhanced to compile and function properly on big-endian (m68k, sparc), and LP64 architectures (alpha, sparc64).
- KDE2 and KOffice together consitutes a fully integrated office environment with no license problems, available for i386, alpha and many other architectures.
- Mozilla 0.9, KDE2's Konqueror, and Links 0.95 are just a few examples of the web browsers available.
- A support package for running VMware on NetBSD/i386 was added, it's called suse_vmware. The official VMware code, a valid license, and Wasabi Systems' compatibility package are all needed to run VMware.
- Internal changes of the pkgsrc system include strong checksums to prevent malicious distribution files, as well as restructuring of the package structure in pkgsrc for faster extraction of pkgsrc tar files and upgrades via SUP and CVS.

Please note that at the moment, **sysinst** will not assist you in installing pre-built third-party binary packages or the pkgsrc system itself, so you will have to manually install packages using **pkg_install** or fetch and extract the pkgsrc.tgz tar file to get started.

Lastly, it should be noted that the X11 binaries shipped in NetBSD 1.5.3 is still based on XFree86 version 3.3.6. Several newer graphics cards are inadequately supported by that code base, but on the other hand support for several older graphics cards is not available in newer XFree86 code. NetBSD is in the process of moving to XFree86 version 4, and is currently maintaining both the XFree86 3.3.6 and the XFree86 version 4 code in the **xsrc** source set, and you may at compile time pick which sources to build and install. To ease installation, testing and use of the XFree86 version 4 code, a binary snapshot based on XFree86 version 4.0.3 will be made available for at least the i386 architecture shortly after the release of NetBSD 1.5.3.

Changes Between The NetBSD 1.4 and 1.5 Releases

The NetBSD 1.5 release provides 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 result of these improvements is a stable operating system fit for production use that rivals most commercially available systems.

It is impossible to completely summarize over one year of development that went into the NetBSD 1.5.3 release. Some highlights include:

Kernel

- Ports to new platforms including: arc, cobalt, hpcmips, news68k, sgimips, and sparc64.
- Improved performance and stability of the UVM virtual memory subsystem.
- Implementation of generic kernel locking code, as well as a restructure and re-tuning of the scheduler, to be used by the future symmetric multi-processing (SMP) implementation.
- Improved compatibility support for Linux, OSF1, and SVR4 programs.
- New compatibility support for Win32 programs.
- Support for dynamically loaded ELF kernel modules.
- Kernel process tracing using `ktruss`(1).

- Deletion of swap devices using `swapctl`(8).
- Easier hot-pluggability of keyboards and mice using a new wscons device—wsmux.
- Improved PCMCIA and CardBus support, including support for detaching of devices and cards, resulting in better support for notebooks and PDA devices.
- Numerous hardware improvements, including areas such as: audio, UDMA/66 support for ATA drives, USB, and wireless networking.

Networking

- Addition of IP version 6 (IPv6) and IPsec to the networking stack, from the KAME project. This includes addition of kernel code for IPv6/IPsec, IPv4/v6 dual-stack user applications and supporting libraries. Due to this, the shlib major version for `pcap`(3) is incremented and you may need to recompile userland tools. The KAME IPv6 part includes results from the unified-IPv6 effort.

File system

- Significant Fast file system (FFS) performance enhancements via integration of Kirk McKusick's soft updates and trickle sync code.
- Support for the Windows NT 'NTFS' file system (read-only at this stage).
- Support for revision 1 of the Linux 'ext2fs' file system.
- Enhanced stability and usability of LFS (the BSD log-structured file system).
- Various RAIDframe enhancements including: auto-detection of RAID components and auto-configuration of RAID sets, and the ability to configure the root file system (`/`) on a RAID set.
- Support for Microsoft Joliet extensions to the ISO9660 CD file system.
- Improved file system vnode locking mechanisms, thus resolving a source of several panics in the past.
- Support for NFS and RPC over IPv6.
- The server part of NFS locking (implemented by `rpc.lockd`(8)) now works.

Security

- Strong cryptographic libraries and applications integrated, including the AES cipher Rijndael, the OpenSSL library, more complete Kerberos IV and Kerberos V support, and an SSH server and client.
- `sysctl`(3) interfaces to various elements of process and system information, allowing programs such as `ps`(1), `dmesg`(1) and the like to operate without recompilation after kernel upgrades, and remove the necessity to run setgid kmem (thus improving system security).
- Disable various services by default, and set the default options for disabled daemons to a higher level of logging.
- Several code audits were performed. One audit replaced string routines that were used without bounds checking, and another one identified and disabled places where format strings were used in unsafe ways, allowing arbitrary data to be entered by (possibly) malicious users to overwrite application code, and leading from Denial of Service attacks to compromised systems.
- `sshd`(8) and `ssh`(1) now require `rnd`(4) kernel random number devices.

System administration and user tools

- Conversion of the rc(8) system startup and shutdown scripts to an 'rc.d' mechanism, with separate control scripts for each service, and appropriate dependency ordering provided by rcorder(8).
- postfix(1) provided as alternative mail transport agent to sendmail(8).
- User management tools useradd(8), usermod(8), userdel(8), groupadd(8), groupmod(8), and groupdel(8) added to the system.
- Incorporation of a login class capability database (/etc/login.conf) from BSD/OS.
- Improved support for usernames longer than eight characters in programs such as at(1) and w(1).
- Many enhancements to ftpd(8) providing features found in larger and less secure FTP daemons, such as user classes, connection limits, improved support for virtual hosting, transfer statistics, transfer rate throttling, and support for various IETF ftpext working group extensions.
- The ftp(1) client has been improved even further, including transfer rate throttling, improved URL support, command line uploads. See the man page for details.

Miscellaneous

- Updates to the NetBSD source code style code (located in /usr/share/misc/style) to use ANSI C only (instead of K&R) and reflect current (best) practice, and begin migrating the NetBSD source code to follow it.
- Implementation of many SUSv2 features to the curses(3) library, including support for color.
- Updates of most third party packages that are shipped in the base system, including file(1), ipfilter(4), ppp(4), and sendmail(8) to the latest stable release.
- Many new packages in the *pkgsrc* system, including standard desktops like KDE and GNOME as well as latest Tcl/Tk and perl and many of the components of the Java Enterprise platform. The package framework itself now has full wildcard dependency support.

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 major release of NetBSD for the mvme68k series of boards.

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.

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

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

NetBSD 1.5.3 Release Contents

The root directory of the NetBSD 1.5.3 release is organized as follows:

`.../NetBSD-1.5.3/`

`CHANGES`	Changes since earlier NetBSD releases.
`LAST_MINUTE`	Last minute changes.
`MIRRORS`	A list of sites that mirror the NetBSD 1.5.3 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 1.5.3 has a binary distribution. There are also `README.export-control` files sprinkled liberally throughout the distribution tree, which point out that there are some portions of the distribution that may be subject to export regulations of the United States, e.g. code under `src/crypto` and `src/sys/crypto`. It is your responsibility to determine whether or not it is legal for you to export these portions and to act accordingly.

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.
22.3 MB gzipped, 98.8 MB uncompressed

pkgsrc This set contains the ‘‘pkgsrc’’ sources, which contain the infrastructure to build third-party packages.
7.4 MB gzipped, 73.0 MB uncompressed

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.
3.3 MB gzipped, 13.2 MB uncompressed

src This set contains all of the base NetBSD 1.5.3 sources which are not in **gnusrc**, **sharesrc**, or **syssrc**.
24.8 MB gzipped, 123.1 MB uncompressed

syssrc This set contains the sources to the NetBSD 1.5.3 kernel for all architectures; `config`(8); and `dbsym`(8).
18.0 MB gzipped, 90.9 MB uncompressed

xsrc This set contains the sources to the X Window System.
78.1 MB gzipped, 393.6 MB uncompressed

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. They may be unpacked into `/usr/src` with the command:

```
# ( cd / ; tar -zxpf - ) < set_name.tgz
```

The `sets/Split/` subdirectory contains 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 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.?? | ( cd / ; tar -zxpf - )
```

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

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

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

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

`SYSVSUM` 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 subdirectory structure

The mvme68k-specific portion of the NetBSD 1.5.3 release is found in the `mvme68k` subdirectory of the distribution: `.../NetBSD-1.5.3/mvme68k/`
`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.VME147.gz`	A kernel for MVME147 boards.
		`netbsd.VME162.gz`	A kernel for MVME162-LX boards.
		`netbsd.VME167.gz`	A kernel for MVME167 boards.
	`sets/`	mvme68k binary distribution sets; see below.	
`installation/`			
	`miniroot/`	mvme68k miniroot file system image; see below.	
	`netboot/`	Two programs needed to boot mvme68k kernels over the network; see below.	
	`tapeimage/`	Tape boot programs, and a RAMDISK kernel; see below.	

Binary distribution sets

The NetBSD mvme68k binary distribution sets contain the binaries which comprise the NetBSD 1.5.3 release for the mvme68k. There are eight binary distribution sets. The binary distribution sets can be found in the `mvme68k/binary/sets` subdirectory of the NetBSD 1.5.3 distribution tree, and are as follows:

base The NetBSD 1.5.3 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.
15.5 MB gzipped, 41.2 MB uncompressed

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.
11.8 MB gzipped, 39.5 MB uncompressed

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.
0.1 MB gzipped, 0.6 MB uncompressed

games This set includes the games and their manual pages.
3.0 MB gzipped, 7.4 MB uncompressed

kern This set contains a NetBSD/mvme68k 1.5.3 `GENERIC` kernel, named `/netbsd`. You *must* install this distribution set.
0.7 MB gzipped, 1.4 MB uncompressed

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.
5.6 MB gzipped, 22.7 MB uncompressed

misc This set includes the (rather large) system dictionaries, the typesettable document set, and other files from `/usr/share`.
2.2 MB gzipped, 7.5 MB uncompressed

text This set includes NetBSD's text processing tools, including `groff`(1), all related programs, and their manual pages.
1.3 MB gzipped, 4.7 MB uncompressed

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. The binaries shipped with NetBSD 1.5.3 are based on XFree86 version 3.3.6. NetBSD is in the process of moving to XFree86 version 4, and the X source set actually contains source for both XFree86 3.3.6 and XFree86 4, and the ability to decide at compile-time which one to build and install. The X Window System binary sets distributed with NetBSD are:

xbase The basic files needed for a complete X client environment. This does not include the X servers.
2.6 MB gzipped, 8.2 MB uncompressed

xcomp The extra libraries and include files needed to compile X source code.
1.7 MB gzipped, 7.2 MB uncompressed

xcontrib Programs that were contributed to X.
0.2 MB gzipped, 0.7 MB uncompressed

xfont Fonts needed by X.
6.2 MB gzipped, 7.5 MB uncompressed

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 -xpf`** command from `/`.

The following are included in the `mvme68k/installation` directory:

`miniroot/`

`miniroot.gz` A copy of the miniroot file system. This filesystem image is copied into the swap partition of the disk which you intend to boot NetBSD from. Normally, this step will be performed manually from the ramdisk boot environment.

`netboot/`

`sboot` An MVME147 bootstrap program in Motorola S-Record format. This is required when you wish to boot an MVME147 over the network since the ROM has no built-in network support. Instructions for getting this program into memory are discussed later.

`netboot` A standalone 2nd stage bootstrap program loaded over the network via TFTP. This is responsible for fetching and starting the NetBSD mvme68k kernel from an NFS server.

`tapeimage/`

`stboot` This file contains a boot sector for Motorola MVME boards. It must be the first file written to a tape in order to make the tape bootable.

`bootst` This file contains a bootstrap program which knows how to load the NetBSD mvme68k ramdisk image from tape. This must be the second file written to the tape.

`netbsd-rd.gz` This is the NetBSD mvme68k ramdisk image. It contains a `GENERIC` kernel and a built in RAMDISK with just enough tools to partition a disk, dump the miniroot kernel to it and make the disk bootable. This must be the third file written to the tape.

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

NetBSD/mvme68k System Requirements and Supported Devices

NetBSD/mvme68k 1.5.3 runs on Motorola MVME147 , MVME162-LX and MVME167 Single Board Computers.

The minimal configuration requires 8 MB of RAM and ~200 MB of disk space. To install the entire system requires much more disk space (approx. 600 MB 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 16 MB of RAM, or 32 MB when running the X Window System.

Note that you can install NetBSD 1.5.3 on a system with only 4 MB of onboard RAM, but you will need to use a VMEbus RAM card with at least another 4 MB to augment the onboard memory in order to actually install the system.

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

Partition	Suggested	Needed
/ (root)	32 MB	26 MB
/usr	200 MB	150 MB
/var	32 MB	6 MB
swap	*2-3 *RAM*	16 MB

Anything else is up to you!

Note that the NetBSD/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 (> 7.5 MB).

Supported VME147 hardware

- Serial ports (RS232)
  - built-in console
  - ttyZ1
  - ttyZ2
  - 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)

- Miscellaneous:
  - Battery-backed real-time clock
  - VMEbus RAM cards
  - Any VMEbus interface cards with Machine Independent drivers

Supported VME162-LX hardware

- Serial ports (RS232)
  - built-in console
  - ttyZ1
  - ttyZ2
  - ttyZ3
- Network interfaces:
  - On-board Intel Ethernet (ie)
- SCSI
  - Most SCSI disks, tapes, CD-ROMs, etc
  - On-board ncr53c710 SCSI I/O processor chip
- Miscellaneous:
  - Battery-backed real-time clock
  - VMEbus RAM cards
  - Any VMEbus interface cards with Machine Independent drivers

Supported VME167 hardware

- Serial ports (RS232)
  - built-in console
  - ttyZ1
  - ttyZ2
  - ttyZ3
- Parallel port
  - On-board centronics style printer port
- Network interfaces:
  - On-board Intel Ethernet (ie)
- SCSI
  - Most SCSI disks, tapes, CD-ROMs, etc
  - On-board ncr53c710 SCSI I/O Processor chip
- Miscellaneous:
  - Battery-backed real-time clock
  - VMEbus RAM cards
  - Any VMEbus interface cards with Machine Independent drivers

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-1.5.3/mvme68k/installation
```

```
.../NetBSD-1.5.3/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-1.5.3/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-1.5.3/mvme68k/binary/sets
# set T = /dev/nrst0
# mt -f $T rewind
# for f in base etc comp games man misc text; do
    gzip -d < $f.tgz | dd of=$T bs=8k
# done
# 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 documentation 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 entitled **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 MVME162-LX and MVME167 boot ROMs have 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 7.5 MB 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:

```
No IP Address: ftp.netbsd.org
Login: anonymous
Password: <your e-mail address>
Server path: /pub/NetBSD/NetBSD-1.5.3/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 October, 2000.

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 exception to the above is that MVME162-LX and MVME167 boards require a jumper to be removed before NetBSD can be installed. On MVME162-LX pins 1-2 of jumper J11 *must* be removed. On MVME167 pins 1-2 of jumper J1 *must* be removed.

Once you've made any necessary jumper changes, 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 MVME162-LX or MVME167 board (the following boot message is from MVME167; MVME162-LX is similar):

```
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 8 MB MVME147 or a 32 MB MVME167 you should have the same value as we do). Also make sure the clock is ticking:

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

Note that NetBSD bases its year at 1968, and adds the year offset in the system'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, as 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 MVME162-LX and MVME167. On the MVME147, enter the last five digits of the address using the **`lsad`** command. On the MVME162-LX and MVME167, you should use the **`cnfg`** command.

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

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

At this point, Bug 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. 16xBug 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 1.5.3 boot tape as described in the section entitled **Preparing a boot tape**. Then, with the tape in the drive, type the following at the Bug prompt:

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

```
16x-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: <Maker, Disk, Foo> SCSI1 0/direct fixed
sd0: 800 MB, 800 cyl, 16 head, 128 sec, 512 bytes/sect x 1638400 sectors
```

The information of most interest is the number of sectors; for the ficticious disk above, it's 1638400. You will need this number when you come to create a disklabel for that drive.

Here is an example of an MVME147 system booting from tape:

```
RAM address from VMEbus = $00000000

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

Volume: NBSD

IPL loaded at:  $003F0000
>> BSD MVME147 tapeboot
578616+422344+55540+[46032+51284]=0x11a6e4
Start @ 0x8000 ...
Copyright (c) 1996, 1997, 1998, 1999, 2000
    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 (RAMDISK) #1: Sun Oct 29 16:19:04 GMT 2000
    steve@fatbob:/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 mvme68k 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 file system (/). 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 file system (/).
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 file system on the Raw Partition, even on a non-boot disk.

It is good practice to put /usr on a different partition than / (root, AKA 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 / (root), /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 file systems 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>: 65536
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 65536 blocks. Note that the usual size of a block is 512 bytes, so this creates a 32 MB partition.

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

Next, create a swap partition (b). Note that the minimum size of this swap partition should be 8 MB, otherwise you won't be able to use a mini-root 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>: 65536
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. 65536. The length of the swap partition should be a multiple of the amount of RAM you have in your system. Here, we've chosen 32768, or 16 MB. The next available block on the drive is thus 65536 + 32768. We will use this to create partition 'd' for our /usr file system. (Note that for a busy system, or a system with more than 8 MB of RAM, you'll be better off with a 32 or 64 MB swap partition.)

```
edlabel/modify> d
d (user) 0 (0/00/00) 0 (0/00/00) unused
start as <blkno> or <cyls/trks/sects>: 98304
length as <nblks> or <cyls/trks/sects>: 1540096
type: 4.2BSD
edlabel/modify> q
edlabel>
```

As you can see, we've chosen to assign the remainder of the disk to /usr. Since there are 1638400 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 98304, a simple bit of arithmetic (1638400 - 98304) gives **d** a size of 1540096.

Note that the above partition sizes are just guidelines. If your disk is large enough, you should resize the partitions appropriately and perhaps also create a /var partition as well.

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: 128
```

```
 tracks/cylinder: 16
       cylinders: 800
sectors/cylinder: 2048
partition      start          (c/t/s)      nblks          (c/t/s)  type

 a (root)          0        (0/00/00)      65536        (32/00/00)  4.2BSD
 b (swap)      65536       (32/00/00)      32768        (48/00/00)  swap
 c (disk)          0        (0/00/00)    1638400       (800/00/00)  unused
 d (user)      98304       (48/00/00)    1540096       (752/00/00)  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 install-boot(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 / (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 MVME162-LX and MVME167 boards are 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 press CONTROL-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 MVME162-LX and MVME167 boards are 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 MVME162-LX or MVME167, all you need to type is:

```
16x-Bug> nbo
```

For all board types, the boot messages are very similar:

```
Start @ 0x8000 ...
>> BSD MVME147 netboot (via sboot)
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, 1998, 1999, 2000
    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 (RAMDISK) #1: Sun Oct 29 16:19:04 GMT 2000
    steve@fatbob:/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 file system (/). 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 MVME162-LX or 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 file system 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 162Bug or 167Bug boot command is:

16x−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 file system 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 file system parameters. If you're unsure what 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 file system.

  The number of partitions is currently fixed at 8.

- Create file systems 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 file systems. 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 file system (/).
- Make device nodes in your root file system under /dev.
- Copy a new kernel onto your root partition (/).
- Install a new boot block.
- Check your file systems 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 properly configured state, with the most important ones described below.

1. Configuring /etc/rc.conf

   If you or the installation software haven't done any configuration of /etc/rc.conf (**sysinst** usually 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-write. 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 **vt220** (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 filesystem 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 multiuser 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).

   If your /usr directory is on a separate partition and you do not know how to use **ed**, you will have to mount your /usr partition to gain access to **ex** or **vi**. Do the following:

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

If you have /var on a separate 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 <int> network interface, 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 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. Unless you've set a password in **sysinst**, there is no initial password. 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. Unless you have connected an unusual terminal device as the console you can just press RETURN when it prompts for Terminal type? [...].

3. Adding accounts

   Use the useradd(8) command to add accounts to your system, *do not* edit /etc/passwd directly. See useradd(8) for more information on 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 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. This automatically handles any changes necessary to make the software run on NetBSD, retrieval and installation of any other packages on which the software may depend, and simplifies installation (and deinstallation), both from source and precompiled binaries.

   - More information on the package system is at
     http://www.netbsd.org/Documentation/software/packages.html

- A browsable listing of available packages is at
  ftp://ftp.netbsd.org/pub/NetBSD/packages/pkgsrc/README.html
- Precompiled binaries can be found at
  ftp://ftp.netbsd.org/pub/NetBSD/packages,
  usually in the 1.5.3/mvme68k/All subdir. You can install them with the following commands:

  ```
  # export PKG_PATH=\
  ftp://ftp.netbsd.org/pub/NetBSD/packages/1.5.3/mvme68k/All\;\
  ftp://ftp.netbsd.org/pub/NetBSD/packages/1.5/mvme68k/All
  # pkg_add -v tcsh
  # pkg_add -v cvs
  # pkg_add -v apache
  # pkg_add -v perl
  ...
  ```

  The above commands will install the tcsh shell, the CVS source code management system, the Apache web server and the perl programming language as well as all the packages they depend on.
- Package sources for compiling packages on your own 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), with the commands:

  ```
  # mkdir /usr/pkgsrc
  # ( cd /usr/pkgsrc ; tar -zxpf - ) < pkgsrc.tar.gz
  ```

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

6. Misc
   - Edit /etc/mail/aliases to forward root mail to the right place. Don't forget to run newaliases(1) afterwards.
   - The /etc/mail/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 5** *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:

1. Enable the network based on your system's current network configuration.
2. Mount your existing file systems.
3. Extract binary sets from the media of your choice.
4. Fixup your system's existing /etc/fstab, changing the occurrences of **ufs** to **ffs** and let you edit the resulting file.
5. Make new device nodes in your root file system under /dev.
6. Don't forget to extract the **kern** set from the distribution.

   Note: The existing kernel *will not* be backed up; doing so would be pointless, since older kernels may not be capable of running NetBSD 1.5.3executables.
7. Install a new boot block.
8. Check your file systems 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:

1. Place *at least* the **base** binary set in a file system accessible to the target machine. A local file system is preferred, since the NFS subsystem in the NetBSD 1.5.3 kernel may be incompatible with your old binaries.
2. Back up your pre-existing kernel and copy the 1.5.3 kernel into your root partition (/).
3. Reboot with the 1.5.3 kernel into single-user mode.
4. Check all file systems:

   # **/sbin/fsck -pf**
5. Mount all local file systems:

   # **/sbin/mount -a -t nonfs**
6. If you keep /usr or /usr/share on an NFS server, you will want to mount those file systems as well. To do this, you will need to enable the network:

   # **sh /etc/rc.d/network start**
7. Make sure you are in the root file system (/) and extract the **base** binary set:

   # **cd /**
   # **pax -zrvpe -f** */path/to/***base.tgz**
8. Install a new boot block:

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

E.g.: *root-disk* could be /dev/rsd0a.

9. Sync the file systems:

   # **sync**

10. At this point you may extract any other binary sets you may have placed on local file systems, 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 1.5.3.

General issues

- /etc/rc modified to use /etc/rc.d/*

  Prior to NetBSD 1.5, /etc/rc was a traditional BSD style monolithic file; each discrete program or substem from /etc/rc and /etc/netstart has been moved into separate scripts in /etc/rc.d/.

  At system startup, /etc/rc uses rcorder(8) to build a dependency list of the files in /etc/rc.d and then executes each script in turn with an argument of 'start'. Many rc.d scripts won't start unless the appropriate rc.conf(5) entry in /etc/rc.conf is set to 'YES.'

  At system shutdown, /etc/rc.shutdown uses rcorder(8) to build a dependency list of the files in /etc/rc.d that have a ''KEYWORD: shutdown'' line, reverses the resulting list, and then executes each script in turn with an argument of 'stop'. The following scripts support a specific shutdown method: cron, inetd, local, and xdm.

  Local and third-party scripts may be installed into /etc/rc.d as necessary. Refer to the other scripts in that directory and rc(8) for more information on implementing rc.d scripts.

Issues affecting an upgrade from NetBSD 1.4 or prior

- named(8) leaks version information

  Previous releases of NetBSD disabled a feature of named(8) where the version number of the server could be determined by remote clients. This feature has not been disabled in NetBSD 1.5, because there is a named.conf(5) option to change the version string:

  ```
  option {
          version "newstring";
  };
  ```

- sysctl(8) pathname changed

  sysctl(8) is moved from /usr/sbin/sysctl to /sbin/sysctl. If you have hardcoded references to the full pathname (in shell scripts, for example) please be sure to update those.

- sendmail(8) configuration file pathname changed

  Due to sendmail(8) upgrade from 8.9.x to 8.10.x, /etc/sendmail.cf is moved to /etc/mail/sendmail.cf. Also, the default sendmail.cf(5) refers different pathnames than before. For example, /etc/aliases is now located at /etc/mail/aliases, /etc/sendmail.cw is now called /etc/mail/local-host-names, and so forth. If you have customized sendmail.cf(5) and friends, you will need to move the files to the new locations. See /usr/share/sendmail/README for more information.

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),
- apropros(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. 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, or visit **http://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 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 or WWW 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 sunlamp 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, SUP, and WWW servers.
- Redback Networks, Inc. for hosting the NetBSD mail and GNATS server.
- The Helsinki University of Technology in Finland for hosting the NetBSD CVS server.
- The Internet Research Institute in Japan for hosting the server which runs the CVSweb interface to the NetBSD source tree.
- The many organisations 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.
- 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 organizations (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
  Søren Jørvang
  Scott Kaplan
  Noah M. Keiserman

Harald Koerfgen
John Kohl
Chris Legrow
Ted Lemon
Norman R. McBride
Neil J. McRae
Perry E. Metzger
Toru Nishimura
Herb Peyerl
Mike Price
Dave Rand
Michael Richardson
Heiko W. Rupp
Brad Salai
Chuck Silvers
Thor Lancelot Simon
Bill Sommerfeld
Paul Southworth
Eric and Rosemary Spahr
Ted Spradley
Kimmo Suominen
Jason R. Thorpe
Steve Wadlow
Krister Walfridsson
Jim Wise
Reinoud Zandijk
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
Distributed Processing Technology
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.
Precedence Technologies Ltd
Salient Systems Inc.
VMC Harald Frank, Germany
Warped Communications, Inc.
Whitecross Database Systems Ltd.

(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:

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**
Simon Burge	simonb@netbsd.org	**pmax**
Jeremy Cooper	jeremy@netbsd.org	**sun3x**
Matt Fredette	fredette@netbsd.org	**sun2**
Ross Harvey	ross@netbsd.org	**alpha**
Jun-ichiro itojun Hagino	itojun@netbsd.org	**sh3**
Ben Harris	bjh21@netbsd.org	**arm26**
Eduardo Horvath	eeh@netbsd.org	**sparc64**
Darrin Jewell	dbj@netbsd.org	**next68k**
Søren Jørvang	soren@netbsd.org	**cobalt**
Søren Jørvang	soren@netbsd.org	**sgimips**
Wayne Knowles	wdk@netbsd.org	**mipsco**
Paul Kranenburg	pk@netbsd.org	**sparc**
Anders Magnusson	ragge@netbsd.org	**vax**
Minoura Makoto	minoura@netbsd.org	**x68k**
Phil Nelson	phil@netbsd.org	**pc532**
Tohru Nishimura	nisimura@netbsd.org	**luna68k**
NONAKA Kimihiro	nonaka@netbsd.org	**prep**
Scott Reynolds	scottr@netbsd.org	**mac68k**
Kazuki Sakamoto	sakamoto@netbsd.org	**bebox**
Noriyuki Soda	soda@netbsd.org	**arc**
Wolfgang Solfrank	ws@netbsd.org	**ofppc**
Ignatios Souvatzis	is@netbsd.org	**amiga**
Jonathan Stone	jonathan@netbsd.org	**pmax**
Shin Takemura	takemura@netbsd.org	**hpcmips**
Jason Thorpe	thorpej@netbsd.org	**alpha**
Jason Thorpe	thorpej@netbsd.org	**hp300**
Tsubai Masanari	tsubai@netbsd.org	**macppc**
Tsubai Masanari	tsubai@netbsd.org	**newsmips**
Izumi Tsutsui	tsutsui@netbsd.org	**news68k**
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 1.5.3 Release Engineering team:

Chris G. Demetriou	cgd@netbsd.org
Håvard Eidnes	he@netbsd.org
Ted Lemon	mellon@netbsd.org
Perry Metzger	perry@netbsd.org
Curt Sampson	cjs@netbsd.org
Jason Thorpe	thorpej@netbsd.org
Todd Vierling	tv@netbsd.org

Developers and other contributors:

Nathan Ahlstrom	nra@NetBSD.org
Steve Allen	wormey@netbsd.org
Julian Assange	proff@netbsd.org
Lennart Augustsson	augustss@netbsd.org
Christoph Badura	bad@netbsd.org
Bang Jun-Young	junyoung@netbsd.org
Dieter Baron	dillo@netbsd.org
Robert V. Baron	rvb@netbsd.org
Jason Beegan	jtb@netbsd.org
Erik Berls	cyber@netbsd.org
John Birrell	jb@netbsd.org
Mason Loring Bliss	mason@netbsd.org
Rafal Boni	rafal@netbsd.org
Manuel Bouyer	bouyer@netbsd.org
John Brezak	brezak@netbsd.org
Allen Briggs	briggs@netbsd.org
Aaron Brown	abrown@netbsd.org
Andrew Brown	atatat@netbsd.org
David Brownlee	abs@netbsd.org
Frederick Bruckman	fredb@netbsd.org
Jon Buller	jonb@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
James Chacon	jmc@netbsd.org
Bill Coldwell	billc@netbsd.org
Julian Coleman	jdc@netbsd.org
Chuck Cranor	chuck@netbsd.org
Alistair Crooks	agc@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
Tracy Di Marco White	gendalia@netbsd.org
Jaromír Dolecek	jdolecek@netbsd.org
Andy Doran	ad@netbsd.org
Roland Dowdeswell	elric@netbsd.org
Emmanuel Dreyfus	manu@netbsd.org
Matthias Drochner	drochner@netbsd.org
Jun Ebihara	jun@netbsd.org
Håvard Eidnes	he@netbsd.org
Stoned Elipot	seb@netbsd.org
Enami Tsugutomo	enami@netbsd.org
Bernd Ernesti	veego@netbsd.org
Erik Fair	fair@netbsd.org
Hubert Feyrer	hubertf@netbsd.org
Jason R. Fink	jrf@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
Brad Grantham	grantham@tenon.com
Brian C. Grayson	bgrayson@netbsd.org
Matthew Green	mrg@netbsd.org
Juergen Hannken-Illjes	hannken@netbsd.org
Charles M. Hannum	mycroft@netbsd.org
Eric Haszlakiewicz	erh@netbsd.org
John Hawkinson	jhawk@netbsd.org
HAYAKAWA Koichi	haya@netbsd.org
René 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
Nick Hudson	skrll@netbsd.org
Martin Husemann	martin@netbsd.org
Dean Huxley	dean@netbsd.org
Bernardo Innocenti	bernie@netbsd.org
Tetsuya Isaki	isaki@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
Takahiro Kambe	taca@netbsd.org

Antti Kantee	pooka@netbsd.org
Masanori Kanaoka	kanaoka@netbsd.org
KAWAMOTO Yosihisa	kawamoto@netbsd.org
Mario Kemper	magick@netbsd.org
Lawrence Kesteloot	kesteloo@cs.unc.edu
Thomas Klausner	wiz@netbsd.org
Klaus Klein	kleink@netbsd.org
Wayne Knowles	wdk@netbsd.org
John Kohl	jtk@netbsd.org
Martti Kuparinen	martti@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
Tomasz Luchowski	zuntum@netbsd.org
Federico Lupi	federico@netbsd.org
Brett Lymn	blymn@netbsd.org
Paul Mackerras	paulus@netbsd.org
MAEKAWA Masahide	gehenna@netbsd.org
David Maxwell	david@netbsd.org
Dan McMahill	dmcmahill@netbsd.org
Gregory McGarry	gmcgarry@netbsd.org
Jared D. McNeill	jmcneill@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
NISHIMURA Takeshi	nsmrtks@netbsd.org
NONAKA Kimihiro	nonaka@netbsd.org
Jesse Off	joff@netbsd.org
Tatoku Ogaito	tacha@netbsd.org
Masaru Oki	oki@netbsd.org
Atsushi Onoe	onoe@netbsd.org
Greg Oster	oster@netbsd.org
Herb Peyerl	hpeyerl@netbsd.org
Matthias Pfaller	matthias@netbsd.org
Chris Pinnock	cjep@netbsd.org
Dante Profeta	dante@netbsd.org
Chris Provenzano	proven@netbsd.org
Michael Rauch	mrauch@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
Reed Shadgett	dent@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
Jeff Smith	jeffs@netbsd.org
SOMEYA Yoshihiko	someya@netbsd.org
Bill Sommerfeld	sommerfeld@netbsd.org
Bill Squier	groo@netbsd.org
Bill Studenmund	wrstuden@netbsd.org
Kevin Sullivan	sullivan@netbsd.org
SUNAGAWA Keiki	kei@netbsd.org
Kimmo Suominen	kim@netbsd.org
TAMURA Kent	kent@netbsd.org
Shin'ichiro TAYA	taya@netbsd.org
Matt Thomas	matt@netbsd.org
Christoph Toshok	toshok@netbsd.org
UCHIYAMA Yasushi	uch@netbsd.org
Shuichiro URATA	ur@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
Todd Whitesel	toddpw@netbsd.org
Rob Windsor	windsor@netbsd.org
Dan Winship	danw@netbsd.org
Jim Wise	jwise@netbsd.org
Michael Wolfson	mbw@netbsd.org
Colin Wood	ender@netbsd.org
YAMAMOTO Takashi	yamt@netbsd.org
Yuji Yamano	yyamano@netbsd.org
Reinoud Zandijk	reinoud@netbsd.org

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:

er 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 Eric Young (eay@mincom.oz.au). 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 University of Oregon. This product includes software developed by the University of Southern California and/or Information Sciences Institute. This product includes software developed by Internet Initiative Japan Inc. This product includes software developed by Reinoud Zandijk.

This product includes software developed by David Jones.

This product includes software developed by Steve C. Woodford.

The End