NAME

INSTALL — Installation procedure for NetBSD/mvme68k.

CONTENTS

DESCRIPTION

About this Document

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

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

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

Changes Between The NetBSD 9 and 10 Releases

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

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

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

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

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

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

CHANGES:
https://cdn.NetBSD.org/pub/NetBSD/NetBSD-10.2/CHANGES

CHANGES-10.1:
https://cdn.NetBSD.org/pub/NetBSD/NetBSD-10.2/CHANGES-10.1

CHANGES-10.2:
https://cdn.NetBSD.org/pub/NetBSD/NetBSD-10.2/CHANGES-10.2

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

Features to be removed in a later release

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

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

The NetBSD Foundation

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

Sources of NetBSD

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

NetBSD 10.2 Release Contents

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

`.../NetBSD-10.2/`

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

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

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

gnusrc This set contains the "gnu" sources, including the source for the compiler, assembler, groff, and the other GNU utilities in the binary distribution sets.

sharesrc This set contains the "share" sources, which include the sources for the man pages not associated with any particular program; the sources for the typesettable document set; the dictionaries; and more.

src This set contains all of the base NetBSD 10.2 sources which are not in **gnusrc**, **sharesrc**, or **syssrc**.

syssrc This set contains the sources to the NetBSD 10.2 kernel for all architectures as well as the `config`(1) utility.

xsrc This set contains the sources to the X Window System.

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

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

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

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

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

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

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

NetBSD/mvme68k subdirectory structure

The mvme68k-specific portion of the NetBSD 10.2 release is found in the `mvme68k` subdirectory of the distribution: `.../NetBSD-10.2/mvme68k/`. It contains the following files and directories:

`INSTALL.html`
`INSTALL.ps`
`INSTALL.txt`
`INSTALL.more` Installation notes in various file formats, including this file. The `.more` file contains underlined text using the `more`(1) conventions for indicating italic and bold display.

`binary/`
- `kernel/`
  - `netbsd-GENERIC.gz`
    A gzipped NetBSD kernel containing code for everything supported in this release.
  - `netbsd-VME147.gz`
    A kernel for MVME147 boards.
  - `netbsd-VME162.gz`
    A kernel for MVME162 boards.
  - `netbsd-VME167.gz`
    A kernel for MVME167 boards.
  - `netbsd-VME172.gz`
    A kernel for MVME172 boards.

`netbsd-VME177.gz`
A kernel for MVME177 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 10.2 release for mvme68k. The binary distribution sets can be found in the `mvme68k/binary/sets` subdirectory of the NetBSD 10.2 distribution tree, and are as follows:

base The NetBSD 10.2 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.

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

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

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

games This set includes the games and their manual pages.

kern-GENERIC
This set contains a NetBSD/mvme68k 10.2 `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 system dictionaries, the typesettable document set, and other files from `/usr/share`.

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

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

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

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

xbase The basic files needed for a complete X client environment. This does not include the X servers.

xcomp The extra libraries and include files needed to compile X source code.

xdebug This distribution set contains debug information for all X11 binaries. It is useful when reporting issues with these binaries or during development. This set is huge, if the target disk is small, do not install it.

xfont Fonts needed by the X server and by X clients.

xetc Configuration files for X which could be locally modified.

xserver The X server.

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 filenames stored in the sets are relative and therefore the files are extracted *below the current directory*. Therefore, if you want to extract the binaries into your system, i.e. replace the system binaries with them, you have to run the **`tar -xzpf`** command from the root directory (`/`) of your system.

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

`miniroot/`

Note: The **`sysinst`** method of installation is the preferred method for installing NetBSD/mvme68k. *The Traditional method of installation is deprecated and will be removed in a future release.* Some other important differences between the two installation methods are:

- The Installer and Mkfs utilities are not used if the installation is done using the **`sysinst`** method via booting an Installation Kernel.
- Mkfs creates a 4.3BSD "old" format file system. This is the only file system format understood by the Installer. The **`sysinst`** process creates "new" file systems which can't be processed by Mkfs or the Installer.
- The Installer (and probably Mkfs) is known to have problems with BSD file systems that are larger than 1 GB or extend beyond the 1 GB physical limit. Keep this in mind if doing a Traditional method installation with these utilities.
- The Booter is known to have problems booting from a NetBSD file system if the inode of the kernel file is too large. For large disks it is advisable to have a small root partition and one or more usr-type partitions for other files. This ensures the inode of the kernel in the root file system will be small.
- The Booter is designed to work with 4.3BSD "old" file systems, but there are enough similarities between the "old" and "new" file system formats in the first few hundred blocks that the Booter can usually boot a Kernel from a 4.3BSD new file system. If you choose to use "new" file system formats, it is advisable to use a small root partition and one or more usr-type partitions for other files. This is the default layout used by **`sysinst`**.

`miniroot.gz` A copy of the miniroot file system. This file system 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-RAMDISK.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 10.2 runs on Motorola MVME147 , MVME162 , MVME167 , MVME172 , and MVME177 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. 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 10.2 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.

A swap partition of 2-3*RAM is recommended.

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/VME172 hardware

- Serial ports (RS232)
  - built-in console
  - ttyC1
  - ttyC2
  - ttyC3
- 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/VME177 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-10.2/mvme68k/installation
.../NetBSD-10.2/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-10.2/mvme68k/installation
# set T = /dev/nrst0
# mt -f $T rewind
# dd if=tapeimage/stboot of=$T obs=8k conv=osync
# dd if=tapeimage/bootst of=$T obs=8k conv=osync
# gzip -dc tapeimage/netbsd-RAMDISK.gz | dd of=$T obs=8k conv=osync
# gzip -dc miniroot/miniroot.fs.gz  | dd of=$T obs=8k conv=osync
# mt -f $T rewind
```

The installation set tape is created as follows:

```
# cd .../NetBSD-10.2/mvme68k/binary/sets
# set T = /dev/nrst0
# mt -f $T rewind
# for f in base etc comp games man misc rescue text; do
    gzip -d < $f.tgz | dd of=$T bs=8k conv=osync
# 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 , MVME167 , MVME172 , and MVME177 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-RAMDISK 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-RAMDISK` 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.fs.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-RAMDISK 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-RAMDISK (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-RAMDISK (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 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-10.2/mvme68k/binary
```

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 , MVME167 , MVME172 and MVME177 boards require a jumper to be removed or DIP switch changed before NetBSD can be installed. On MVME162-LX and MVME172-LX pins 1-2 of jumper J11 *must* be removed. On MVME162-P2/P4 and MVME172-P2/P4 switch S4, position 8 *must* be set to OFF. On MVME167 and MVME177 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 MVME162/172 or MVME167/177 board (the following boot message is from MVME167; the others are 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.
NetBSD does not affect

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 other board types. On the MVME147, enter the last five digits of the address using the **lsad** command. On the MVME162/172 and MVME167/177, you should use the **cnfg** command.

The NetBSD kernel reads the first two long words of the onboard NVRAM to determine the starting and ending address of any VMEbus RAM that should be used by the system. You should verify that this area is set properly for your configuration.

If you have no VMEbus RAM boards, the values should be set to zero (0).

For an MVME162, MVME167, MVME172 or MVME177 board, at the 1xx-Bug> prompt:

```
1xx-Bug>mm fffc0000 ;l
fffc0000: xxxxxxxx?0
fffc0004: xxxxxxxx?0
fffc0008: xxxxxxxx?.
1xx-Bug>
```

For an MVME147 board, at the 147Bug prompt:

```
147Bug>mm fffe0764 ;l
fffe0764: xxxxxxxx?0
fffe0768: xxxxxxxx?0
fffe076c: xxxxxxxx?.
```

If you do have VMEbus RAM available and want NetBSD to use it, the first long word should be set to the starting address of this RAM and the second long word should be set to the ending address.

If you have more than one VMEbus RAM board installed, the starting and ending addresses must be contiguous from one board to the next. Also note that, for various reasons beyond the scope of this document, VMEbus RAM should be configured in A32 address space.

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 issuing 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 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 MVME147 machine has a tape drive the easiest way is **Installing from tape** (details below). All other machines can be installed easily over the network from a suitable NFS server. See **Installing from NFS** for details. 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 10.2 boot tape as described in the section entitled **Creating boot/install tapes**. 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 fictitious 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:  $003F0000
>> BSD MVME147 tapeboot [$Revision: 1.25 $]
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:  Partition your disk (use the command:  disklabel -i -I /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 if=/dev/nrst0 of=/dev/rsd0b bs=8k conv=sync
(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
#
```

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 disklabel program, passing it the name of the Raw Partition of your root/boot disk.

```
# disklabel -i -I /dev/rsd0c
important disklabel -i options:
? - short help on all commands
P - print current partition table
W - write the current partition table
Q - quit
[a-p] - define named partition
```

To start creating the basic partitions, you should enter **a** at the partition> prompt. disklabel will then prompt for the partition information in a form similar to:

```
partition> a
Filesystem type [?] [unused]: 4.2BSD
Start offset ('x' to start after part 'x') [0c, 0s, 0M]:
Partition size ('$' for all remaining) [0c, 0s, 0M]: 32m
```

When you enter the start and length of a partition, you can append s, c, m, or g to specify sectors, cylinders, megabytes or gigabytes respectively.

The above example creates partition 'a', starting at sector zero and with a size of 32MB.

The `type` of the partition should be `4.2BSD`, which is the recommended NetBSD file system type.

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 miniroot to complete the NetBSD installation!

```
partition> b
Filesystem type [?] [unused]: swap
Start offset ('x' to start after part 'x') [0c, 0s, 0M]: a
Partition size ('$' for all remaining) [0c, 0s, 0M]: 16m
```

Here, we specify that the swap partition starts immediately after the 'a' partition and is of size 16MB. The length of the swap partition should be at least as large as the amount of RAM in your system and ideally 64MB or over. The remainder of the disk can be allocated to the 'd' partition for the `/usr` file system.

```
partition> d
Filesystem type [?] [unused]: 4.2BSD
Start offset ('x' to start after part 'x') [0c, 0s, 0M]: b
Partition size ('$' for all remaining) [0c, 0s, 0M]: $
```

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 the shell using '**Q**'.

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 shell prompt, use the following commands to copy the miniroot image from tape to swap (b).

```
# mt -f /dev/nrst0 rewind
# mt -f /dev/nrst0 fsf 3
# dd if=/dev/nrst0 of=/dev/rsd0b bs=8k conv=osync
```

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

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

You can now shutdown the system.

```
# halt
signal 15
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] Y?
Cold/Warm Reset flag [C,W] = C?
Execute Soft Reset [Y,N] N? y
```

Resetting the other types of MVME 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, MVME167, MVME172 and MVME177 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 MVME162/172 or MVME167/177, all you need to type is:

```
1xx-Bug> nbo
```

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

```
Start @ 0x8000 ...
>> BSD MVME147 netboot (via sboot) [$Revision: 1.25 $]
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:

```
# ifconfig le0 inet 192.168.1.4 up
```

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

```
# ifconfig ie0 inet 192.168.1.4 up
```

will bring up the MVME162/172 or MVME167/177 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:

```
# route add default 192.168.1.2 1
```

You can look at the route table using:

```
# route show
```

Now mount the NFS file system containing the miniroot image:

```
# 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:

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

Otherwise, you will need to use **gzcat** to expand the miniroot image while copying.

```
# gzcat miniroot.fs.gz | dd of=/dev/rsd0b obs=8k conv=osync
```

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 boot command is:

```
1xx-Bug> bo 0,,b:
```

The command line parameters above are:

0 controller (usually zero)

,, 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.

All recent releases of NetBSD provide the option of using **sysinst** instead of the old installation and upgrade shell scripts. At this time, **sysinst** is not capable of installing distribution sets from tape. In such a case, you should fallback to the shell script installation or upgrade procedure. In all other cases, **sysinst** should be used. Note that as soon as **sysinst** can deal with tapes, the shell script tools will be removed.

The miniroot's install program (both **sysinst** and the shell script version) 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.

Running the sysinst installation program

1. *Introduction*

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

2. *General*

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

3. *Quick install*

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

   - What you need.
     - The distribution sets (in this example, they are on the CD or DVD).
     - One 1.44 MB 3.5" floppy.
     - A minimum of 4 MB of memory installed.
     - An optical drive.
     - A hard drive with at least 500 MB of free space for a complete base install, not including room for swap. If you wish to install the X Window System as well, you will need at least 225 MB more.
   - The Quick Installation
     - Insert the first boot floppy you just created and boot the computer. After language selection, the main menu will be displayed.

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

     - If you wish, you can configure some network settings immediately by choosing the **Utility menu** and then **Configure network**. It isn't actually required at this point, but it may be more convenient. Go back to the main menu.
     - Choose **Install**.
     - You will be guided through the setup of your disk.
     - You will be asked to choose which distribution sets to install.
     - When prompted, choose **CD-ROM** as the install medium if booted from CD-ROM. The default values for the path and device should be OK.
     - After the installation process has completed, you will be brought back to the main menu, where you should select **Reboot**, after you have removed the bootfloppy from the drive.

– NetBSD will now boot. If you didn't set a password for the `root` user when prompted by **`sysinst`**, logging in as `root` and setting a password should be your first task. You are also advised to read `afterboot`(8).

4. *Booting NetBSD*

   Boot your machine. The boot loader will start, which will print a countdown and begin booting.

   If the boot loader messages do not appear in a reasonable amount of time, you either have a bad boot floppy or a hardware problem. Try writing the install floppy image to a different disk, and using that.

   It will take a while to load the kernel from the floppy, probably around a minute or so; then the kernel boot messages will be displayed. This may take a little while also, as NetBSD will be probing your system to discover which hardware devices are installed. The most important thing to know is that `wd0` is NetBSD's name for your first SATA/PATA disk, `wd1` the second, etc. `sd0` is your first SCSI disk, `sd1` the second, etc.

   Note that once the system has finished booting, you need not leave the floppy in the disk drive.

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

5. *Network configuration*

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

6. *Installation drive selection and parameters*

   To start the installation, select **Install NetBSD to hard disk** from the main menu.

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

7. *Selecting which sets to install*

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

8. *Partitioning the disk*

   • Choosing which portion of the disk to use.

     You will be asked if you want to use the entire disk or only part of the disk. If you decide to use the entire disk for NetBSD, **`sysinst`** will check for the presence of other operating systems and you will be asked to confirm that you want to overwrite these.

9. *Editing the NetBSD disklabel*

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

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

`a` Root partition (/)

`b` Swap partition.

`c` The entire disk.

`d-h` Available for other use. Traditionally, `d` is the partition mounted on `/usr`, but this is historical practice and not a fixed value.

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

10. *Preparing your hard disk*

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

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

11. *Getting the distribution sets*

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

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

12. *Installation from CD-ROM*

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

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

13. *Installation using FTP*

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

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

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

14. *Installation using NFS*

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

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

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

15. *Installation from an unmounted file system*

    In order to install from a local file system, you will need to specify the device that the file system resides on (for example `wd1e`), the type of the file system, and the directory on the specified file system where the sets are located. **sysinst** will then check if it can indeed access the sets at that location.

16. *Installation from a local directory*

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

17. *Extracting the distribution sets*

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

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

18. *Configure additional items*

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

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

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

19. *Finalizing your installation*

    Congratulations, you have successfully installed NetBSD 10.2. You can now reboot the machine and boot NetBSD from hard disk.

Post installation steps

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

1. Before all else, read `postinstall`(8).

2. Configuring `/etc/rc.conf`

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

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

   and with the root file system (`/`) mounted read-only. When the system asks you to choose a shell, simply press RETURN to get to a `/bin/sh` prompt. If you are asked for a terminal type, respond with **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 file system read/write with:

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

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

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

   Other values that may need to be set in `/etc/rc.conf` for a networked environment are *hostname* and possibly *defaultroute*. You may also need to add an *ifconfig_int* for your ⟨int⟩ network interface, along the lines of

   ```
   ifconfig_le0="inet 192.0.2.123 netmask 255.255.255.0"
   ```

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

   ```
   ifconfig_le0="inet myname.my.dom netmask 255.255.255.0"
   ```

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

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

3. Logging in

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

4. Adding accounts

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

5. The X Window System

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

6. Installing third-party packages

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

   - More information on the package system is available at
     https://www.NetBSD.org/docs/software/packages.html
   - A list of available packages suitable for browsing is at
     https://cdn.NetBSD.org/pub/pkgsrc/current/pkgsrc/README.html
   - Precompiled binaries can be found at
     https://cdn.NetBSD.org/pub/pkgsrc/packages/NetBSD/
     usually in the `mvme68k/10.2/All` subdir. If you installed `pkgin`(1) in the **sysinst** post-installation configuration menu, you can use it to automatically install binary packages over the network. Assuming that `/usr/pkg/etc/pkgin/repositories.conf` is correctly configured, you can install them with the following commands:

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

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

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

     ```
     export PKG_PATH=https://cdn.netbsd.org/pub/pkgsrc/packages/NetBSD/$(uname -p)/$
     pkg_add pkgin
     ```
   - If you wish to use the `pkgsrc`(7) framework for compiling packages and did not install it from the `sysinst`(8) post-installation configuration menu, you can obtain it by retrieving the file
     https://cdn.NetBSD.org/pub/pkgsrc/stable/pkgsrc.tar.gz.
     It is typically extracted into `/usr/pkgsrc` (though other locations work fine) with the commands:

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

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

7. Misc

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

     ```
     # man 5 filename
     ```

     is likely to give you more information on these files.

Upgrading a previously-installed NetBSD System

It is possible to easily upgrade your existing NetBSD/mvme68k system using the upgrade program in the miniroot or by manually performing the same steps as the miniroot upgrade program.

Upgrading using 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 10.2 executables.
7. Install a new boot block.
8. Check your file systems for integrity.
9. You'll have to reboot your system manually

Manual upgrade

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 10.2 kernel may be incompatible with your old binaries.
2. Back up your pre-existing kernel and copy the 10.2 kernel into your root partition (/).
3. Reboot with the 10.2 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 10.2.

Note that **sysinst** will automatically invoke

    postinstall fix

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

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

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

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

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

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

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

Using online NetBSD documentation

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

- intro(1),
- man(1),

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

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

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

```
# man passwd
```

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

```
# man 5 passwd
```

instead.

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

```
# apropos subject-word
```

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

Administrivia

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

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

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

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

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

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

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

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

Thanks go to

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

  Keith Bostic
  Ralph Campbell
  Mike Karels
  Marshall Kirk McKusick

for their work on BSD systems, support, and encouragement.

- The Internet Systems Consortium, Inc. for hosting the NetBSD FTP, CVS, AnonCVS, mail, mail archive, GNATS, SUP, Rsync and WWW servers.
- The Internet Research Institute in Japan for hosting the server which runs the CVSweb interface to the NetBSD source tree.
- The Columbia University Computer Science Department for hosting the build cluster.
- The many organizations that provide NetBSD mirror sites.
- Without CVS, this project would be impossible to manage, so our hats go off to Brian Berliner, Jeff Polk, and the various other people who've had a hand in making CVS a useful tool.
- We list the individuals and organizations that have made donations or loans of hardware and/or money, to support NetBSD development, and deserve credit for it at **https://www.NetBSD.org/donations/** (If you're not on that list and should be, tell us! We probably were not able to get in touch with you, to verify that you wanted to be listed.)
- Finally, we thank all of the people who've put sweat and tears into developing NetBSD since its inception in January, 1993. (Obviously, there are a lot more people who deserve thanks here. If you're one of them, and would like to be mentioned, tell us!)

Legal Mumbo-Jumbo

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

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

NetBSD is a registered trademark of The NetBSD Foundation, Inc.
This product includes software developed by the University of California, Berkeley and its contributors.
This product includes software developed by the NetBSD Foundation.
This product includes software developed by The NetBSD Foundation, Inc. and its contributors.
This product includes software developed for the NetBSD Project. See https://www.NetBSD.org/ for information about NetBSD.
This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)
This product includes cryptographic software written by Eric Young (eay@mincom.oz.au)
This product includes software designed by William Allen Simpson.
This product includes software developed at Ludd, University of Luleå.
This product includes software developed at Ludd, University of Luleå, Sweden and its contributors.
This product includes software developed at the Information Technology Division, US Naval Research Laboratory.
This product includes software developed by Aaron Brown and Harvard University.
This product includes software developed by Adam Ciarcinski for the NetBSD project.
This product includes software developed by Adam Glass.
This product includes software developed by Adam Glass and Charles M. Hannum.
This product includes software developed by Alex Zepeda.
This product includes software developed by Alex Zepeda, and Colin Wood for the NetBSD Project.
This product includes software developed by Allen Briggs.
This product includes software developed by Amancio Hasty and Roger Hardiman
This product includes software developed by Ben Gray.
This product includes software developed by Berkeley Software Design, Inc.
This product includes software developed by Bill Paul.
This product includes software developed by Bodo Möller.
This product includes software developed by Boris Popov.
This product includes software developed by Brini.
This product includes software developed by Bruce M. Simpson.
This product includes software developed by Causality Limited.
This product includes software developed by Charles Hannum.
This product includes software developed by Charles M. Hannum.
This product includes software developed by Charles M. Hannum, by the University of Vermont and State Agricultural College and Garrett A. Wollman, by William F. Jolitz, and by the University of California, Berkeley, Lawrence Berkeley Laboratory, and its contributors.

This product includes software developed by Christian E. Hopps.
This product includes software developed by Christian E. Hopps, Ezra Story, Kari Mettinen, Markus Wild, Lutz Vieweg and Michael Teske.
This product includes software developed by Christopher G. Demetriou.
This product includes software developed by Christopher G. Demetriou for the NetBSD Project.
This product includes software developed by Chuck Silvers.
This product includes software developed by Cisco Systems, Inc.
This product includes software developed by Colin Wood.
This product includes software developed by Colin Wood for the NetBSD Project.
This product includes software developed by Computing Services at Carnegie Mellon University (http://www.cmu.edu/computing/).
This product includes software developed by Daan Vreeken.
This product includes software developed by Daishi Kato
This product includes software developed by Daniel Widenfalk and Michael L. Hitch.
This product includes software developed by Daniel Widenfalk for the NetBSD Project.
This product includes software developed by David Jones and Gordon Ross
This product includes software developed by David Miller.
This product includes software developed by Dean Huxley.
This product includes software developed by Emmanuel Dreyfus
This product includes software developed by Eric S. Hvozda.
This product includes software developed by Eric S. Raymond
This product includes software developed by Eric Young (eay@cryptsoft.com)
This product includes software developed by Eric Young (eay@mincom.oz.au)
This product includes software developed by Ezra Story.
This product includes software developed by Ezra Story and by Kari Mettinen.
This product includes software developed by Ezra Story, by Kari Mettinen and by Bernd Ernesti.
This product includes software developed by Ezra Story, by Kari Mettinen, and Michael Teske.
This product includes software developed by Ezra Story, by Kari Mettinen, Michael Teske and by Bernd Ernesti.
This product includes software developed by Frank van der Linden for the NetBSD Project.
This product includes software developed by Gardner Buchanan.
This product includes software developed by Garrett D'Amore.
This product includes software developed by Gary Thomas.
This product includes software developed by Gordon Ross
This product includes software developed by Harvard University.
This product includes software developed by Harvard University and its contributors.
This product includes software developed by Hellmuth Michaelis and Joerg Wunsch
This product includes software developed by Henrik Vestergaard Draboel.
This product includes software developed by Herb Peyerl.
This product includes software developed by Hidetoshi Shimokawa.
This product includes software developed by Hubert Feyrer for the NetBSD Project.
This product includes software developed by Ian W. Dall.
This product includes software developed by Intel Corporation and its contributors.
This product includes software developed by Internet Initiative Japan Inc.
This product includes software developed by Internet Research Institute, Inc.
This product includes software developed by James R. Maynard III.
This product includes software developed by Jared D. McNeill.
This product includes software developed by Jason L. Wright
This product includes software developed by Jason R. Thorpe for And Communications, http://www.and.com/
This product includes software developed by Joachim Koenig-Baltes.
This product includes software developed by Jochen Pohl for The NetBSD Project.
This product includes software developed by Joerg Wunsch
This product includes software developed by John Birrell.
This product includes software developed by John P. Wittkoski.
This product includes software developed by John Polstra.
This product includes software developed by Jonathan R. Stone for the NetBSD Project.
This product includes software developed by Jonathan Stone.
This product includes software developed by Jonathan Stone and Jason R. Thorpe for the NetBSD Project.
This product includes software developed by Jonathan Stone for the NetBSD Project.
This product includes software developed by Julian Highfield.
This product includes software developed by K. Kobayashi
This product includes software developed by K. Kobayashi and H. Shimokawa
This product includes software developed by Kazuhisa Shimizu.
This product includes software developed by Kazuki Sakamoto.
This product includes software developed by Kenneth Stailey.
This product includes software developed by Kiyoshi Ikehara.
This product includes software developed by Klaus Burkert,by Bernd Ernesti, by Michael van Elst, and by the University of California, Berkeley and its contributors.

This product includes software developed by Kyma Systems.
This product includes software developed by Leo Weppelman and Waldi Ravens.
This product includes software developed by Lloyd Parkes.
This product includes software developed by Lutz Vieweg.
This product includes software developed by Marc Horowitz.
This product includes software developed by Marcus Comstedt.
This product includes software developed by Mark Brinicombe.
This product includes software developed by Mark Brinicombe for the NetBSD Project.
This product includes software developed by Mark Tinguely and Jim Lowe
This product includes software developed by Markus Wild.
This product includes software developed by Marshall M. Midden.
This product includes software developed by Masanobu Saitoh.
This product includes software developed by Masaru Oki.
This product includes software developed by Matt DeBergalis
This product includes software developed by Matthew Fredette.
This product includes software developed by Michael Smith.
This product includes software developed by Microsoft
This product includes software developed by Mika Kortelainen
This product includes software developed by Mike Pritchard.
This product includes software developed by Mike Pritchard and contributors.
This product includes software developed by Minoura Makoto.
This product includes software developed by MINOURA Makoto, Takuya Harakawa.
This product includes software developed by Niels Provos.
This product includes software developed by Niklas Hallqvist.
This product includes software developed by Niklas Hallqvist, Brandon Creighton and Job de Haas.
This product includes software developed by Paolo Abeni.
This product includes software developed by Paul Kranenburg.
This product includes software developed by Paul Mackerras.
This product includes software developed by Paul Mackerras <paulus@samba.org>.
This product includes software developed by Pedro Roque Marques <pedro_m@yahoo.com>
This product includes software developed by Per Fogelstrom.
This product includes software developed by Peter Galbavy.
This product includes software developed by Phase One, Inc.
This product includes software developed by Philip A. Nelson.
This product includes software developed by QUALCOMM Incorporated.
This product includes software developed by Ravikanth.
This product includes software developed by RiscBSD.
This product includes software developed by Roar Thronæs.
This product includes software developed by Rodney W. Grimes.
This product includes software developed by Roger Hardiman
This product includes software developed by Rolf Grossmann.
This product includes software developed by Ross Harvey.
This product includes software developed by Ross Harvey for the NetBSD Project.
This product includes software developed by Scott Bartram.
This product includes software developed by Scott Stevens.
This product includes software developed by Shingo WATANABE.
This product includes software developed by Softweyr LLC, the University of California, Berkeley, and its contributors.
This product includes software developed by Stephan Thesing.
This product includes software developed by Steven M. Bellovin
This product includes software developed by Takashi Hamada.
This product includes software developed by Takumi Nakamura.
This product includes software developed by Tatoku Ogaito for the NetBSD Project.
This product includes software developed by Tommi Komulainen <Tommi.Komulainen@iki.fi>.
This product includes software developed by TooLs GmbH.
This product includes software developed by Trimble Navigation, Ltd.
This product includes software developed by Waldi Ravens.
This product includes software developed by WIDE Project and its contributors.
This product includes software developed by Winning Strategies, Inc.
This product includes software developed by Yen Yen Lim and North Dakota State University
This product includes software developed by Zembu Labs, Inc.
This product includes software developed by the Alice Group.
This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory.
This product includes software developed by the Computer Systems Laboratory at the University of Utah.
This product includes software developed by the Harvard University and its contributors.
This product includes software developed by the Kungliga Tekniska Högskolan and its contributors.
This product includes software developed by the Network Research Group at Lawrence Berkeley Laboratory.

any improvements or extensions that they make and grant Carnegie the rights to redistribute these changes.

Some files have the following copyright:

Some files have the following copyright:

The End