LynxOS 2.5.0 Release Notes

LynxOS Version 2.5.0

DOC-0183 039707

Printed in the United States of America.

Contents

CHAPTER 1

New Features in LynxOS 2.5.0

Improved C and C++ Tools

We now have integrated the GNU C/C++ compiler and tools, based on Cygnus Support's Cygnus Developer's Kit, as the default compilation environment. This change is the next step in our ongoing program (initiated in LynxOS 2.3) to upgrade the LynxOS development environment. Please see Chapter 3, "Compilers in LynxOS 2.5.0" for detailed information.

Upgraded Networking Facilities

LynxOS networking support has been upgraded based on the 4.4BSD-Lite distribution. The new features include:

- Thread-safe BSD library (`libbsd.a`)
- Device independent Ethernet processing absorbed in TCP/IP code
- Upgraded networking utilities
- Network snooping using binary packet filter and `tcpdump`.

Protocol enhancements include:

- TCP header prediction
- Serial line IP (SLIP) header compression
- Long fat pipe support (for high bandwidth, long delay paths)
- IP level multicasting support
- Updated routing algorithms

This upgrade provides many benefits to networking support in LynxOS, including:

- The new protocol implementations benefit from further years of protocol research at UC Berkeley, yielding substantial overall robustness and performance benefits. For instance, in preliminary performance measurements over a serial line, we have seen two to three times improvements in throughput in some environments (benefiting from the SLIP header compression and other optimizations).
- The upgrade builds a foundation for future performance improvements.
- The new BSD library provides better support of threaded applications, allowing them to safely call networking library functions without the burden of serializing those calls.
- Raw Ethernet support is available by default as part of the TCP/IP protocol stack.

ISO9660 File System and CD-ROM Drive Support

LynxOS 2.5.0 offers full support for ISO9660 file systems with Rock Ridge Interchange Protocol (RRIP) on all four platforms, providing access to the ubiquitous distribution medium of CD-ROMs. New LynxOS utilities (on x86 systems only) allow you to create CD-ROMs that are readable by most other computer systems. We also think you'll find the CD-ROM distribution of LynxOS much more convenient to use than floppies or tapes.

NOTE: *By default, LynxOS products are shipped on CD-ROM media. Please see the LynxOS Installation Guide for more information.*

Booting from CD-ROM

LynxOS supports booting directly from CD-ROM media on PowerPC, 68k, and SPARC systems. CD-ROM media are treated as non-partitioned, read-only disks by LynxOS I/O disk drivers.

Driver Support

With the exception of the Adaptec 1522 SCSI device driver (`sim1522`) for x86, all LynxOS SCSI drivers (`sim*`, `sd`, `ncr810`) have been enhanced to support SCSI CD-ROM drives on all platforms. Also, the IDE driver (`hd`) now supports the ATA Packet Interface to IDE CD-ROM drives on the x86.

Utility Support

Standard file-oriented facilities (`cd`, `close`, `ioctl`, `mount`, `open`, `pwd`, `read`, `select`, `write`, `umount`) support ISO9660 file systems as read-only. For example, to mount a CD-ROM device on `/cdrom`, type the following:

mount -o ro /dev/*device* **/cdrom**

where **ro** specifies read-only and *device* is the device name of a CD-ROM drive on your system.

In addition, two new utilities are provided: `mkisofs` and `cdwrite`. The `mkisofs` utility makes an ISO9660 or ISO9660+RRIP file system. The `cdwrite` utility, supported only on the x86, writes an ISO9660 file system image to a CD Recorder. Please refer to the online manual pages for details about these utilities.

PosixWorks® Desk

PosixWorks Desk is a graphical user interface (GUI) designed to provide easy access to LynxOS facilities and tools. Used in conjunction with the X & Motif Development Package, PosixWorks Desk aids in software development and debugging. Currently, PosixWorks Desk is supported only on LynxOS for x86; it will be available on LynxOS for PowerPC in the future.

PosixWorks Desk is an installation option with the X & Motif Development Package. You can choose to use either the Motif Window Manager (`mwm`) or Tom's Window Manager (`twm`) with PosixWorks Desk. You can also use the online help facility to learn more about PosixWorks Desk.

Overview of the Task Bar

The primary feature, the Task Bar at the bottom of the screen, provides button and menu access to the underlying features of PosixWorks Desk. Selecting (pressing the left mouse button) an icon on the Task Bar results in the default action. Pressing the right mouse button allows you to select alternate options.

Task Bar Features

Clock	Displays analog or digital time; updates the display every minute.
Calendar	Displays current date and month.
Files	Folder-style file manager that shows a graphical view of files and directories, allowing easy file copy and file move operations. Drag-and-drop feature

	supports dragging a file icon to an editor or other utility to execute, or dragging it to the trash can to delete.
`Editors`	Provides access to several text editors, including `vi` (default), `nedit`, `emacs`, and `xedit`. Also, `bitmap`, `editres`, and `pixmap` editors are available.
`Shells`	Provides access to several UNIX shells, including `bash` (default), `tcsh`, `dlsh`, and `tcl` shell.
`Network`	Menu access to networking functions, including `ftp`, `telnet`, `rlogin`, and `ping`.
`Virtual Screens`	Allows four virtual screens.
`Printer`	Menu access to printer setup and control functions.
`Applications`	Menu access to the calculator, X Windows tools and games, and demonstration programs.
`Programs`	Toolbar for TotalView, TimeScan, LynxInsure++, X Tools, and Examples. If one of these products is not on your system, the button for it will be gray.
`Administration`	Menus to change the color and background attributes of your desktop.
`Trash`	Allows drag-and-drop deletion of files. Selecting this icon displays the contents of the `~/.trash` directory.
`Help`	Provides detailed help information about PosixWorks Desk.

New Text Editor

The new `nedit` text editor provides graphical menus, dialog boxes, editing facilities, and mouse support. It also can highlight C, C++ and HTML language syntax. Use the Preferences->Highlight Syntax menu to enable this feature.

Improved ROMing Support

ROMing facilities have been improved by replacing the `mkromk` utility. The new utility, called `mkimage`, supports a common interface across all native and cross LynxOS environments. It takes input from a user-defined specification file that sets the necessary parameters for a given ROM application.

The new mkimage improves many steps in the process of making kernel images, such as:

- You no longer use cryptic command line options; the ASCII specification file fully controls the attributes of the image.
- The specification file can create a root file system with arbitrary files and directories.
- You no longer need “root” privileges to create ROMable images.
- Images can be tested by simply loading from preboot.

We have already experienced many of these benefits here at Lynx, using mkimage to build the bootable images for LynxOS installation.

See the *LynxOS User's Guide* and the mkimage manual page for more information about this utility.

Upgraded X Windows

X Windows support has been upgraded to the X Window System Release 6 (X11R6) based on the Metro Link 3.1.5 general release. It also includes an upgrade to OSF/Motif 2.0. These releases fix hundreds of specific bugs and provide many new features, including:

- Improved install/uninstall scripts
- Support for the XKB keyboard extension
- Additional Motif widgets such as ComboBox and Notebook
- Improved Motif Window Manager (mwm) with virtual screen support

Improved Netboot and SCMP

Network booting (netboot) and Snugly Coupled Multiprocessing (SCMP) support have been improved in this release. LynxOS provides a new build environment with easy to use scripts that will help you implement these facilities quickly in your environment. See the *LynxOS User's Guide* for more information.

New and Improved Facilities

LynxOS Library Functions

Table 1-1 lists new library functions in this release. For more information, use the man command to display the online manual page.

Table 1-1: New and Upgraded Library Functions in LynxOS 2.5.0

Library Function	Description
fpgetround, fpsetround, fpgetmask, fpsetmask, fpgetsticky, fpsetsticky	IEEE floating-point environment control.

LynxOS System Calls

Table 1-2 lists new and improved system calls. For more information, use the man command to display the online manual page.

Table 1-2: New and Updated System Calls in LynxOS 2.5.0

System Call	Description
sysctl[1]	Get or set system information.
getdents	Read directory entries into a file system-independent format; formerly SVR3-only system call but now implemented in general context.
getsockopt, setsockopt	Get and set options on sockets.

1. New system call for LynxOS 2.5.0.

LynxOS Utilities

Table 1-3 lists new and improved utilities in this release. For more information, use the `man` command to display the online manual page.

Table 1-3: New and Updated Utilities in LynxOS 2.5.0

Utilities	Description
`as`	GNU assembler.
`apropos, man, whatis`	GNU utilities to format and display the online manual pages.
`ar`	Create, modify, and extract from archives.
`autoconf,`[1] `autoheader, autoreconf`	GNU utility to create the `configure` file from `configure.in`.
`byacc`	LALR parser generator.
`c++filt`	Demangle C++ symbols.
`cdwrite`[1]	Write a file or partition to a CD-R (x86).
`cmp`	Compare two files.
`comm`[1]	Compare two sorted files line by line.
`diff, diff3, sdiff`	File comparison utilities.
`ex, vi, view`	Text editors based on nvi -1.71.
`expect`	Programmed dialogue with interactive programs, Version 5.
`expectk`	Expect with Tk support.
`flex, flex++`	Fast lexical analyzer generator.
`gasp`	GNU assembler preprocessor.
`gcc, g++`	GNU C and C++ compiler.
`gdb`	GNU debugger.
`ld`	GNU linker.
`m4`	General-purpose macro processor.
`make`	GNU make utility to maintain groups of programs.
`makeinfo`	Build GNU info files.
`mkimage`[1]	Make LynxOS kernel images. Replaces `mkromk`.
`mkisofs`[1]	Create an ISO9660 or ISO9660 + RRIP file system.
`netstat`[1]	Show network status.
`nm`	List symbols from object files.

Table 1-3: New and Updated Utilities in LynxOS 2.5.0

Utilities	Description
`objcopy`	Copy and translate object files.
`objdump`	Display information from object files.
`patch`	Apply a diff file to an original.
`ping`	Send ICMP `ECHO_REQUEST` packets to network hosts.
`query`[1]	Query the route daemon to get route information.
`ranlib`	Generate index to archive.
`size`	List section sizes and total size (in bytes).
`strings`	Print the strings of printable characters in files.
`strip`	Discard symbols from object files.
`route`	Manually manipulate the routing tables.
`routed`	Network routing daemon.
`tclsh`	Simple shell containing Tcl interpreter.
`tr`	Translate or delete characters.
`trace_net`[1]	Trace on/off message to routed daemon.

1. New utility for LynxOS 2.5.0.

In addition, the `mkshlib` utility for shared libraries is now supported from both native and cross development systems on LynxOS for x86, 68k, and SPARC.

CHAPTER 2

Compatibility with LynxOS 2.5.0

This chapter discusses the compatibility issues with LynxOS 2.5.0 and previous releases.

General Source Compatibility

LynxOS 2.5.0 is source compatible with LynxOS 2.4.0 in most instances. However, Lynx continues to improve support for ANSI C, making some prototype and header file changes that may require source code changes to user applications. See “Header File Changes” on page 11 for more information. You should also review Chapter 3, “Compilers in LynxOS 2.5.0” for details about the compiler changes in this release.

We recommend that you recompile and relink applications from source code where possible.

POSIX Support

If your applications continue to use POSIX.4 Draft 9 (referred to as “4D9”) facilities, refer to the LynxOS 2.4.0 Development OS Release Notes and POSIX.1b Migration Guide for details on how to convert to POSIX.1b.

CAUTION! *Please note that LynxOS 2.5.0 is the last release in which* POSIX.4 *Draft 9 facilities will be supported.*

As a reminder, Table 2-1 summarizes the POSIX command line options to GNU C/C++.

Table 2-1: GNU C/C++ Compiler Options for POSIX Support in LynxOS 2.5.0

POSIX Specification	In LynxOS 2.5.0
POSIX 1003.1 C Language Standard	default
POSIX 1003.1b Final version of 1003.4 (Draft 14)	default
POSIX 1003.4a Draft 4 Thread Extensions	**-mthreads**
POSIX 1003.4 Draft 9 Real-Time Extensions Draft	**-mposix4d9**

Device Driver Compatibility

Device drivers should be source compatible as well, except in the following two areas:

- User-written mass storage device drivers.

  You may need to change these drivers to attach to CD-ROM drives. LynxOS drivers have typically assumed a 512 byte sector size while CD-ROM drives only support a 2048 byte sector size.
- User-written Ethernet drivers. See the *Writing Device Drivers for LynxOS Guide* for more information.

General Binary Compatibility

Applications from LynxOS 2.4.0 generally are binary compatible with the LynxOS 2.5.0 release. But there are important exceptions that may cause inconsistencies in broad classes of applications.

File System Changes

Several file-oriented library functions were modified to support ISO9660 file systems:

```
opendir()
readdir()
telldir()
seekdir()
rewinddir()
closedir()
```

Because of these changes, LynxOS 2.4.0 utilities that contain these functions may not work with LynxOS 2.5.0, especially across NFS-mounted file systems. If you have a LynxOS 2.4.0 application that contains one of these functions, you will need to relink this application with LynxOS 2.5.0.

ROMing Changes

The improvements to ROMing facilities has resulted in the following incompatibilities:

- The LynxOS 2.5.0 `preboot` cannot be used to boot ROM images from earlier releases of LynxOS.
- The LynxOS 2.5.0 `mkimage` utility cannot be used to create ROM images for LynxOS 2.4.0 or earlier releases.
- Existing netboot ROMs for x86 systems (which worked under both LynxOS 2.3 and LynxOS 2.4.0) will not work with LynxOS 2.5.0.

Header File Changes

The following header file changes were made in LynxOS 2.5.0 and may impact compatibility.

ANSI Prototypes

We continue to improve coverage of prototypes to include non-ANSI functions. However, if you encounter missing prototypes, you can:

- Add your own prototypes for the missing functions or contact Lynx Technical Support for more information.
- Turn off prototype checking in the GNU C/C++ compiler by using options such as **`-fno-strict-prototype`**. See the *Using GNU C* manual for more information on compiler options.

Structure Field Renames for Galaxy

The kernel partitioning for the recently announced Galaxy project required the renaming of several structure fields and one global variable as shown in Table 2-2.

Table 2-2: Structure Fields and Global Variable Renamed for Galaxy Technology

File	Structure	Old Name	New Name
<fa.h>	fa_info	preempt	fa_preempt
<proc.h>	st_entry	link	st_link
<proc.h>	st_entry	quantum	st_quantum
<proc.h>	pssentry	umask	pss_umask
<proc.h>	pssentry	vtimer	pss_vtimer
<proc.h>	pssentry	ptimer	pss_ptimer

Table 2-2: Structure Fields and Global Variable Renamed for Galaxy Technology

File	Structure	Old Name	New Name
`<proc.h>`	`pssentry`	`utime`	`pss_utime`
`<time.h>`	`clock_descr`	`rtimer_list`	`cd_rtimer_list`
x86 `<kernel.h>`	N/A	`job`	`Job`
68k `<kernel.h>`	N/A	`job`	`Job`
`<mem.h>`	N/A	`job`	`Job`
`<mem.h>`	`cram_header`	`root_device`	`cram_root_device`

NGROUPS and MAXSYMLINKS Now Configurable

The following changes resulted from making `NGROUPS` and `MAXSYMLINKS` configurable parameters:

- The #define for the `NGROUPS` and `MAXSYMLINKS` constants were moved from `<conf.h>` to `uparam.h` in `/sys/lynx.os`. This allows a user to modify the default value of `NGROUPS` or `MAXSYMLINKS` for the system then rebuild a kernel. These constants should *not* be used in application programs. Instead, application programs should obtain the value of `NGROUPS` or `MAXSYMLINKS` using the `info()` system call.
- The “groups” field in the following structures has been changed from an array to a pointer:

```
<proc.h>:              struct pssentry
<nfs/lockd_clnt.h>:    struct lock_msghdr
<nfs/unfs_driver.h>:   struct unfs_msghdr
```

SCSI Manager Changes

The header file `<scsimgr.h>` has been modified in order to reduce the amount of space taken up by the SCSI manager. The values of `MAXACTIO`, `MAXSGLIST`, and `MAXCHAINS` have been reduced, and a few data structures have been modified.

Miscellaneous Changes

Table 2-3: Miscellaneous Header File Changes in LynxOS 2.5.0

Header File	Changes
<inode.h>	m_vfs has been added to the end of struct mount_entry; changed argument types in prototype of new_inode().
<param.h>	Fixed definition for MAX_MSGS_PER_QUEUE.
<resolv.h>	Changed the type of _res.
<rk.h>	Removed the #defines for RKINFO_LOC, RK_MEMGAP, and RK_OSCOMPRESS.
<ttymgr.h>	Changed the trans field of struct ttystatics from type int (*)() to type void (*)().
<kernel.h>	Moved typedef for recobuf_t to <proc.h>.
<gmon.h>	Removed the macros ROUNDDOWN() and ROUNDUP().
<ioctl.h>	Changed the prototype for ioctl().
<pwd.h>	Fixed the return types for getpwuid_r(), getpwnam_r(), and getpwent_r().
<string.h>	Modified the prototypes for strdup(), bcmp(), bcopy(), and bzero().
<wait.h>	Removed the non-POSIX versions of the wait() and waitpid() prototypes (i.e. the ones that used union wait).

LynxOS Kernel Directory Reorganization

In LynxOS 2.5.0, the overall structure of the /sys directory (location of kernel and related components) has been changed in two ways. First, the files have been partitioned into several functional groups, each in a separate directory. Second, kernel components are maintained in archives, rather than as individual .o files (and driver files continue to be maintained that way). If you are an end user, these differences will require just a few changes in your development work, even though the implementations of Makefiles, install/uninstall scripts and other support components have been updated.

The immediate benefits of this new organization include:

- Simplification of the `lynx.os` directory to contain mainly the user-modifiable files. This directory now has only a few files versus dozens in previous releases.
- Elimination of `ar`/`libr`/`ranlib` operations during kernel reconfiguration, yielding improved robustness and speed. We've seen the time for installing all the optional "packages" (e.g., TCP/IP, NFS, etc.) into the kernel improve by a factor of five on some systems.
- The install and uninstall scripts, in addition to being faster, are now much simpler, improving Lynx's ability to ensure that they are robust and usable across all the native and cross platforms we support.

The following sections brieflt discuss the new structure of `/sys`, the implications for end users, and the new development functionality that has resulted. These topics are discussed in more detail in the *LynxOS User's Guide*. You can also view the file `/sys/README`.

Directory Structure of /sys

The LynxOS `/sys` directory contains all the scripts, library archives, and platform-specific files needed to customize and rebuild a LynxOS kernel. It is organized as follows:

`Makefile`	Top level Makefile
`OBJ_RULES`	Makefile rules
`README`	File describing the `/sys` structure
`bsp.xxx/`	Default Board Support Package (BSP) for the main hardware platforms currently supported by LynxOS (`x86`, `ppc`, `usparc`, `030_M147`, `040_MVME`). This directory contains the parts of the LynxOS kernel that are particular to a given target board. Also, the actual kernel image is modified and linked in this directory.
`cfg/`	Specifies driver entry points for each device driver
`devices/`	Device driver information files
`dheaders/`	Device driver header files
`drivers/`	Device driver sources
`lib/`	Contains the following kernel and driver library archives:

`aux.a`	Auxiliary support routines
`devices.a`	Devices

`drivers.a`	Device drivers
`hybrid.a`	Hybrid driver
`kernel.a`	Kernel routines
`nfs_client.a`	NFS Client code
`nfs_server.a`	NFS server code
`nullnfs.a`	Null NFS driver
`streams.a`	STREAMS code
`syscalls.a`	System call interface
`tcpip.a`	TCP/IP code

`lynx.os@`	A symbolic link to the supported BSP
`romkit/`	Examples for building bootable kernel images
`stubs/`	"Stub files" for particular functionality modules that can override objects in `kernel.a`

Of these directories, you will most likely modify only the files in the `lynx.os` directory (a symbolic link to `bsp.xxx`). You can also modify the files in:

```
cfg/
devices/
dheaders/
drivers/
```

However, these files primarily are modified only when new drivers are added, or when the functionality of a specific driver needs to be changed. See the *LynxOS User's Guide* for more information.

End User Impacts

The notes below explain how to do common kernel development tasks within the new structure compared to how it was done in previous releases. The next section, "New Support for Team Development" on page 16, describes additional methods for kernel development that are new in LynxOS 2.5.0.

Rebuilding Kernels

Rebuilding the kernel in the new structure requires no changes in the steps that you take. You still invoke the following:

```
cd /sys/lynx.os
make install
```

Adding Device Drivers

Now drivlib.a, renamed to drivers.a, and devlib.a, renamed to devices.a, live in the /sys/lib directory. All the standard LynxOS Makefiles have been modified to deal with this change. See the *LynxOS User's Guide* for examples on how to add device drivers.

Making Custom Kernels

In previous versions of LynxOS, you would modify <param.h> to build custom kernels. In LynxOS 2.5.0, the editable parameters of <param.h> have been extracted into a file called uparam.h in the lynx.os directory. Modify this file (uparam.h) and rebuild the kernel. Only source code customers can modify <param.h>.

Configuring Stubs

The KERNEL.CFG facility has been removed. Now you just need to define an environment variable called STUBS within /sys/lynx.os/Makefile that specifies the set of stub files to be included in the kernel you're building. Remember that this Makefile executes in the /sys/lynx.os directory while the stub files are in /sys/stubs.

New Support for Team Development

In the new paradigm, necessary libraries and other sources within /sys are referenced via the $ENV_PREFIX environment variable when building the kernel. This variable is / by default on native development systems and is set to /usr/lynx/2.5.0/*target* by the SETUP.*shell* script on cross development systems. The structure changes and use of $ENV_PREFIX provides better support for team development by allowing engineers to build and test customized kernels in their own personal directories while leveraging off common libraries and sources.

Root privileges were required for kernel building in previous LynxOS releases, and kernel building was restricted to the /sys directory. In LynxOS 2.5.0, you can copy $ENV_PREFIX/sys/lynx.os to some other directory and make kernels within that directory. Other members of your team also can make their own copies of $ENV_PREFIX/sys/lynx.os. The team can develop in parallel yet continue to use the same fundamental kernel building components designated by $ENV_PREFIX.

There are three primary files, located in $ENV_PREFIX/sys/lynx.os, that you can edit to "customize" a LynxOS kernel: CONFIG.TBL, Makefile, and uparam.h. You can modify CONFIG.TBL to add or remove (by commenting in or out with the # sign) device drivers in the kernel. You can modify uparam.h to customize kernel parameters such as number of system threads (NTHREADS). And, you can modify Makefile to include custom libraries or other components.

Once you have changed some parameters and want to build a new kernel, invoke `make all` or `make install` from within `lynx.os`. The first option, `make all`, makes new kernel (`a.out`) and `nodetab` files in the working directory.

The second option, `make install`, executes `make install`, moves the “old” kernel `$ENV_PREFIX/lynx.os` to `$ENV_PREFIX/lynx.os.old`, copies the new kernel (`a.out`) to `$ENV_PREFIX/lynx.os`, copies the new `nodetab` to `$ENV_PREFIX/etc`. Both options work from a `lynx.os` directory located anywhere on the system.

You can also easily configure LynxOS “package” support for TCP/IP, NFS, STREAMS, and SKDB using `/lynx.os/Makefile`. For example, to make a local kernel copy with TCP/IP in a personal build directory:

```
cp -r /sys/lynx.os /tmp
cd /tmp/lynx.os
make all.tcp
```

Refer to the chapter on customizing kernels in the *LynxOS User's Guide* for more examples.

Networking Upgrade

The upgrade to BSD 4.4 compatibility involved several changes to LynxOS that are discussed in the following sections.

Data Structure Changes

The data structures `sockaddr` and `msghdr` in `<socket.h>` were changed in LynxOS 2.5.0. We have retained BSD 4.3-compatible definitions of these structures in `<socket.h>`, called `osockaddr` and `omsghdr`. If your source code uses `sockaddr` and/or `msghdr`, you have two options for compatibility with LynxOS 2.5.0:

1. Modify your source code to use the new structures and their parameters.
2. Rename calls in the source code from `sockaddr` to `osocksddr` and/or `msghdr` to `omsghdr`.

See the *Writing Device Drivers for LynxOS Guide* for more information on networking device drivers.

Header File Changes

- In <ioctl.h>, all the socket I/O controls have been deleted. These ioctl requests are now defined in the new header <sockio.h>. Source code that uses the following ioctl calls can either redefine them as shown in Table 2-4 or modify the source code to use the appropriate modified structures defined in <sockio.h>.

Table 2-4: How to Use Previously Available Ioctl Calls in LynxOS 2.5.0

LynxOS 2.4.0 and Earlier	LynxOS 2.5.0	Description
SIOCGIFADDR	OSIOCGIFADDR	get ifnet address
SIOCGIFDSTADDR	OSIOCGIFDSTADDR	get p-p address
SIOCGIFBRDADDR	OSIOCGIFBRDADDR	get broadcast addr
SIOCGIFCONF	OSIOCGIFCONF	get ifnet list
SIOCGIFNETMASK	OSIOCGIFNETMASK	get net addr mask

- Replaced <socket.h> with the one from BSD 4.4.
- In <st.h>, two more thread-specific indices have been added (THREADSPECIFIC_BSD and THREADSPECIFIC_H_ERRNO), and the value of THREADSPECIFIC_FIRSTAVAIL has been raised from five to eight.
- In /usr/include/arpa, /usr/include/bsd, and /usr/include/protocols, several headers have been replaced by BSD 4.4 versions. The changes to /usr/include/bsd are shown in Table 2-5.

Table 2-5: Header File Changes in /usr/include/bsd

LynxOS 2.4.0 and Earlier	LynxOS 2.5.0
bsd_domain.h	domain.h
bsd_mbuf.h	mbuf.h
bsd_protosw.h	protosw.h
bsd_socketvar.h	socketvar.h

Impacts on Existing Binaries

Improvements to the LynxOS networking facilities has resulted in the following binary incompatibilities:

- Due to the header file changes needed for the upgrade, networking applications must be recompiled and linked using the new libraries.
- Networking-oriented libraries built with LynxOS 2.4.0 or previous releases will not work properly in LynxOS 2.5.0.
- Existing binaries that attempt to access NFS-mounted directories will fail. Prior to LynxOS 2.5.0, there was an inappropriate work-around to the lack of a "get directory entries" (`getdents(2)`) service. Now the old work-around has been removed, affecting existing binaries that depend on it.

Ethernet Driver Name Changes

The Ethernet device names were modified to be more industry standard by removing numerics from the names. Table 2-6 shows the new names of the Ethernet devices in LynxOS 2.5.0. See the `Intro(4)` manual page for more information.

Table 2-6: New Ethernet Device Names in LynxOS 2.5.0

New Device Name (LynxOS 2.5.0)	Previous Device Name (LynxOS 2.4.0)
wd	wd3e0
el_isa	3c509
el_eisa	3c579
oblan	oblan
smc	smc_dec21040
amd	amd970
dlpi_dec	dlpi_21040
ultra	N/A

SMC Ethernet Adapters on x86 Systems

The `wd` driver should be used only with the SMC EtherElite or older adapters. The newly named `ultra` driver should be used with SMC EtherUltra and SMC EtherEZ Ethernet adapters.

System V Support

In our efforts to further strengthen LynxOS's compliance to standards, we have removed SVR3 functions that conflict with more recently adopted standards. We have absorbed some non-conflicting System V facilities into the Lynx API: `getdents`, `times`, `statfs`, `fstatfs`, `uname`, `sysfs`, `ulimit`, and `ustat`.

CAUTION! *All special System V libraries, compilation support for establishing, and mechanisms for entering System V execution mode have been removed from this release.*

If you have been developing with the 94Q4/95Q2 or 96Q1 GNU C/C++ compiler tools, this change should not affect your applications.

Other Compatibility Notes

Library Changes

The library `libcyg.a` has been removed, and the object modules that it contained have been moved into `libc.a`. This change should be transparent to users.

LynxOS 2.5.0 and Previous Releases on the Same System (X86)

There are several compatibility issues to consider if you plan to install and use LynxOS 2.5.0 on an x86 system that also has previous versions of LynxOS. The most important thing to remember is that previous versions of `preboot` cannot boot LynxOS 2.5.0 kernels. Similarly, due to enhancements for this release, the LynxOS 2.5.0 `preboot` cannot boot kernels from older releases.

LynxOS 2.5.0 and Older Releases on Different Disks

The easiest way to implement this scenario is to have the most current release (LynxOS 2.5.0) installed on disk 0 (the bootable disk according to most BIOS types). This ensures that LynxOS 2.5.0 is the default version.

To boot the other disks on the system, first boot to a floppy that contains the correct version of `preboot`, then use preboot commands to boot the older release from disk.

LynxOS 2.5.0 and Older Releases on the Same Disk, Different Partitions

This is similar to the previous scenario. Use a boot floppy with the correct version of `preboot` to load a previous release.

New Key Combination Entry into SKDB

To break into SKDB, the default key combination is now **Shift-Ctl-Minus**. Some keyboards may support using **Ctl-Minus**.

rc.network and CONFIG.TBL

The `rc.network` and `CONFIG.TBL` files from previous LynxOS releases will not work on LynxOS 2.5.0. You should use the versions provided in this release.

Using l-strip on Kernels (X86 and 68K)

You must use the Lynx strip utility, `l-strip`, to remove symbol information from an `a.out` format file. This includes *all* LynxOS kernels on x86 and 68k systems. See Chapter 3, "Compilers in LynxOS 2.5.0" for more information about the legacy Lynx tools in this release.

Using ranlib on Kernel Libraries (X86 and 68K)

In Beta releases of LynxOS 2.5.0 for x86 and 68k, the `/bin/ranlib` utility could not be used on kernel library files (`.a` files in `/sys/lib`) due to incompatibilities between the legacy Lynx library archive utility `libr` and the new GNU `ar` and `ranlib` utilities.

This problem has been fixed in this release. We no longer use the GNU `ar` utility (`/bin/ar`) and `/bin/ranlib` to create the `/sys/lib/*.a` libraries.

NOTE: *The* `ranlib` *utility is called automatically with GNU* `ar`*.*

To use GNU `ar` requires that all `.o` files that are archived in `/sys/lib/*.a` libraries must be COFF objects, the default object format for `/bin/gcc` and `/bin/g++`. The legacy Lynx C compiler, `l-cc`, produces `a.out` format; therefore, the resulting `a.out` must be to be converted to COFF format *before* archiving into a `/sys/lib` archive.

To convert an `a.out` format file to COFF, execute a partial link as shown below:

`l-ld -k -r -o` *coff_file*.**o** *file*.**o**

CAUTION! *Failure to convert an* `a.out` *object file into a COFF file before archival into the* `sys/lib/*.a` *library will result in errors when linking the kernel. The linker error will state:*

```
Bad COFF object in archive: /sys/lib/drivers.a
```

X Windows Online Manual Pages

The manual pages for the X & Motif Development Package are in `/usr/man/X11` and `/usr/man/Motif`. To resolve manual page paths, the `xman` utility now reads the `MANPATH` environment variable and the `/usr/lib/man.config` file.

PowerPC-Specific Compatibility Issues

Dynamic Driver Installation on PowerPC

LynxOS for PowerPC supports dynamic driver installation with the GNU C compiler but requires a special *import file* for the linker. This import file contains a list of driver service calls used by the device driver.

To determine what services need to be specified in the import file, examine the sources for your driver, noting all functions that are not defined in your application. Alternatively, you can compile the object file (`.o`) for your driver and use the `nm` utility to display the list of unresolved symbols in that object. This approach is shown in the example below.

1. Compile the driver with GNU C.

   ```
   gcc -c -o driver.o driver.c
   ```

2. Determine the driver services for the import file. See the `nm(1)` manual page for more information about this utility.

   ```
   nm driver.o | grep U
   U .fclear
   U .free1page
   U .get1page
   U .iointclr
   U .iointset
   U .sysbrk
   U currpid
   ```

3. Create an import file with these services. For example, the file `driver.import` contains:

```
fclear
free1page
get1page
iointclr
iointset
sysbrk
currpid
```

4. Now create a dynamically loadable object module by entering the following:

```
ld -bM:SRE -bimport:driver.import -o driver.obj \
driver.o
```

The option **-BM:SRE** tells the linker that this is a shared reusable module. The option **-bimport** tells the linker the name of the import file.

This will create a object module which has a loader section for dynamic linking and stub functions for kernel callbacks.

CAUTION! *If there are several source files for your driver, make sure that the unresolved symbols displayed by* `nm` *for one file are not defined in another source file. The list for the import file should contain only those services that are external to the driver sources.*

The `entry_points` structure in dynamic drivers needs to be global, not static. If the structure is defined as a `static`, the dynamic loader (`drinstall`) will fail to get `entry_points` and report the error message “Bad Exec Format.”

Logical Partition A for Booting

In LynxOS 2.4.0 for PowerPC, each disk must reserve a 10 MB partition A to contain the bootable kernel. With LynxOS 2.5.0, this is no longer a requirement.

In LynxOS 2.5.0, partition A is a logical partition that points to a contiguous file containing the bootable image plus the necessary hardware information required for booting by `pboot`. This means that partition A is really a subset of the partition that contains the contiguous file. If you make new partitions with `mkpart` on a LynxOS 2.5.0 system, you should always specify **0** to be the size of partition A.

NOTE: *You should not remove the contiguous file. If removed, partition A will load the blocks that comprised the contiguous file and may get bad data if those blocks have been used for something else.*

LynxOS preboot Utility Now Supported

The `preboot` utility is a new feature on PowerPC platforms. As such, you cannot use `preboot` to boot a pre-2.5 kernel. Instead, you must use the facilities of the PPCbug. For example, assume that a SCSI disk (on one machine) at ID 0 is configured as follows:

Partition B LynxOS 2.4.0	Partition C LynxOS 2.5.0 (with preboot)

In this example, `preboot` on Partition C has been made bootable with the `makeboot` utility (see `makeboot(1)` for more information).

To boot the LynxOS 2.5.0 kernel `/lynx.os` on partition C:

1. Boot preboot from the PPC Bug.

   ```
   Bug> pboot 0 0
   ```

2. Press **Enter** at the `preboot` prompt.

To boot the LynxOS 2.4.0 kernel `/lynx.os` on partition B of a SCSI disk at ID 0:

1. From PPC Bug, set the major device number (in hexadecimal) of the root device.

   ```
   Bug> mm 80000074; B
   80000074 10 ? 15
   80000075 10 ? 00
   80000076 10 ?
   80000077 02 ? 0        major number in hex
   80000078 10 ? .
   ```

2. Set the minor device number (in hexadecimal) of the root device.

   ```
   Bug> mm 80000074; B
   80000074 10 ? 14
   80000075 10 ? 00
   80000076 10 ?
   80000077 20 ? 20       minor number in hex
   80000078 10 ? .
   ```

3. Boot `preboot`.

   ```
   Bug> pboot 0 0
   ```

4. At the `preboot` prompt, boot the LynxOS 2.4 kernel.

   ```
   b s0b /lynx.os
   ```

CHAPTER 3

Compilers in LynxOS 2.5.0

This section provides an overview of the compiler environment provided in LynxOS 2.5.0.

What's New in LynxOS 2.5.0?

LynxOS 2.5.0 now supports one default compilation environment based on the Cygnus Developer's Kit 96Q1 GNU C/C++. Previous versions of GNU C/C++ (versions 1.x) have been removed from x86, 68k, and SPARC systems, allowing full integration of Cygnus GNU C/C++ as the default compilers. Lynx also has migrated most of the kernel-level code to GNU C on the x86 and 68k platforms, renaming the Lynx proprietary C compiler and tools as legacy Lynx utilities. (See "Legacy Lynx C Compiler and Tools (X86 and 68K only)" for more information.)

NOTE: *Check your scripts and Makefiles for references to the* `CYGNUS.` *shell utility. This script has been removed in LynxOS 2.5.0. If your development activities use the Lynx* `ENVIRONMENT` *file, your scripts won't need to be modified.*

New Features of 96Q1 GNU C/C++

The 96Q1 GNU C/C++ software development tools provide many new features, specifically:

- Improved conformance of the C++ compiler to the draft ANSI/ISO C++ standard.
- Additional support for `.align`, `.balign`, and `.p2align` directives in the assembler and linker. Previously, these directives were only truly effective if the requested alignment was some very small value (for example, 8 or less). Now, however, much larger values can be used.
- Support for a new (optional) third argument (used to specify required alignment) in `.bss` and `.lcomm` pseudo-ops in the assembler.
- Support for the PowerPC processor family in the GNU linker (`ld`) `ranlib`, archiver (`ar`), `size`, and `strip` programs. These GNU tools replace the proprietary counterparts that were supplied in LynxOS 2.4.0 for PowerPC.
- Improved performance of the tools and of programs generated by them.
- New processor-specific options: **`-m586`** and **`-m68060`**.

Compiler Location

The GNU C/C++ compilers and tools are now located in `/usr/bin` and `/usr/lib` with symbolic links provided in the standard release directories, `/bin` and `/lib`. You do not need to invoke any setup scripts to access these tools—they are available by default.

Legacy Lynx C Compiler and Tools (X86 and 68K only)

In this release, a limited subset of proprietary Lynx C compiler and tools has been retained to build a very small part of the LynxOS product. These “legacy” Lynx tools remain because we did not complete the conversion of all Lynx C style in-line assembly code in the kernel and libraries to GNU C style. We intend to complete this conversion and remove the renamed tools in a subsequent release of LynxOS.

The Lynx C tools have been renamed `l-`*utilname* to indicate their “legacy” status (for example, `l-cc` is the “legacy” Lynx C compiler). Lynx believes the renaming and eventual removal of these tools will not be an issue for customer applications since most users already have converted their applications to GNU C/C++. However, if you have application code that uses the Lynx C in-line assembly, you should convert that code to GNU C in-line assembly language. Please contact Lynx Technical Support if you need assistance.

CAUTION! *The Lynx C archiver (*`ar`*) is not included in this release because it cannot be used on LynxOS 2.5.0 objects. Using an* `ar` *from LynxOS 2.4.0 (or earlier) on LynxOS 2.5.0 objects will result in library corruption.*

Compatibility with 94Q4/95Q2 GNU Compilers

In general, you can continue to use existing relocatable COFF object files (and libraries containing such files) with the new tools. However, we strongly recommend that you recompile all such binaries, whenever possible, to get the performance benefits offered by the new tools.

At the source level, you may need to change many things in existing C++ source files before they can be successfully compiled with the new GNU C++ compiler. The differences and incompatibilities are largely due to the still evolving nature of the ANSI/ISO C++ standard; this is beyond the control of Lynx Real-Time Systems.

With C sources, you should see few, if any differences between this release of the GNU C compiler for LynxOS and earlier version of the GNU C compiler on LynxOS. However, there are many differences and incompatibilities between the GNU C compiler and the legacy Lynx C compiler (`l-cc`). Most importantly, the legacy Lynx C compiler is a pre-ANSI C compiler, while the GNU C compiler supports the full ANSI C standard.

Changes to Predefined Preprocessor Symbols

For both the GNU C and GNU C++ compilers, the set of compiler predefined preprocessor symbols has changed significantly in LynxOS 2.5.0. You may need to modify your source code in response to these changes.

Predefines for LynxOS

For all LynxOS supported target platforms, the GNU C and C++ compilers will predefine the symbol **`__Lynx__`** just as in earlier releases. However, the symbols **`LYNX`**, **`__LYNX`**, **`__LYNX__`**, **`Lynx`**, **`__Lynx`**, and **`__lynx__`** are no longer automatically predefined by the compilers for any LynxOS supported target platform. Nor are the symbols **`unix`**, **`__unix`**, **`__unix__`**, **`IBITS32`**, **`__IBITS32`**, and **`__IBITS32__`** predefined by the compilers when you are compiling for any LynxOS supported target platform.

Predefines for Target Architectures

LynxOS 2.4.0 and earlier compilers automatically defined certain symbols identifying the target architecture for conditional compilation of architecture-dependent source code. The chosen symbol names were not necessarily compliant with ANSI name space rules. In particular, the following previously available compiler-predefined preprocessor symbols have been retired in this release:

Table 3-1: Obsolete Compiler Predefines by Architecture

Architecture	Obsolete Compiler Predefines					
x86	`I386__`	`i386`	`__i386`	`__i386__`	`I386`	`__I386__`
68k	`mc68000`	`__mc68000`	`__mc68000__`	`M68K`	`__M68K`	`__M68K__`
SPARC	`sparc`	`__sparc`	`SPARC`	`__SPARC`	`__SPARC__`	

NOTE: *A backward compatibility switch,* **`-mbackward`**, *can be used to cause the compiler to throw the retired predefines. We strongly urge you to begin migrating towards the new definitions as soon as possible.*

The standardized processor family symbols that are automatically defined by the 96Q1 GNU C compilers are shown below. Note that the symbol __ represents two underscore characters together.

x86	__x86__
68k	__68k__
SPARC	__sparc__
PowerPC	__powerpc__

In addition to predefining one of the above processor family symbols (depending upon the type of the target system), the compilers will also predefine one additional symbol to indicate the exact model of CPU being compiled for. Table 3-2 indicates the correspondence between compiler options and predefined symbols for specific CPU models:

Table 3-2: Compiler Predefines for Individual Processors in an Architecture Family

Architecture Family	Processor Option	Compiler Directive
x86	none	`__i386__`
	`-m386`	`__i386__`
	`-m486`	`__i486__`
	`-m586`	`__i586__`

Table 3-2: Compiler Predefines for Individual Processors in an Architecture Family

Architecture Family	Processor Option	Compiler Directive
m68k	none	`__m68020__`
	-m68020	`__m68020__`
	-m68020-40	`__m68020__`
	-m68030	`__m68030__`
	-m68040	`__m68040__`
	-m68060	`__m68060__`
SPARC	none	`__sparc_v7__`
	-msparclite	`__sparclite__`
	-mv8	`__sparc_v8__`
PowerPC	none	`__ppc601__`
	-m603	`__ppc603__`
	-m604	`__ppc604__`

Global Declarations

Another important difference in this release that may affect (and require changes in) C source code relates to the precise semantics of file-scope (i.e. "global") data object declarations that do not have the explicit storage class specifiers `extern` or `static` and do not include explicit initializers. In previous releases of GNU C for LynxOS, all such data objects were allocated by the compiler to "common" storage. Also, all such declarations of objects that had the same name (from all of the translation units comprising a complete C program) were, by default, effectively merged at link-time (by the linker) into a single allocated data object. (This behavior could always be overridden by using the **-fno-common** option.)

Beginning with 96Q1 GNU C for LynxOS, the default behavior of the compiler has been changed to **-fno-common**. This causes each such data object declaration to be allocated storage *within the current translation unit* (i.e. within the `.o` file resulting from the current compilation). The net effect of this change is that you may now occasionally get "multiply defined" errors at link time for C programs that previously compiled and linked without errors. In such cases, we recommend that you locate all of the declarations of the offending data object and make all but one of them `extern`. Alternatively, you may use the **-fcommon** option to the GNU C compiler to restore the old compiler behavior.

Part of the rationale for this change relates to a problem with linking that can best be illustrated by the following short code example:

```
int sync;     int main () { sync = 1; return 0; }
```

This code would, when compiled with 94Q4 or 95Q2 GNU C, fail at run-time with a somewhat mysterious `SIGSEGV` (segmentation fault) because the variable `sync` was allocated (by the C compiler) to "common" and was later commonized (inappropriately) by the linker with the C library function of the same name. Thus, the assignment to the `sync` variable, in fact ended up as an attempt to write into code space—a protected operation. In order to minimize such potential problems, you should minimize the allocation of C language data objects to common.

This change also addresses another situation—the contribution of "common" data objects to overall program size is not well reported by the `size` program. In fact, although the `size` program will correctly report the exact sizes for the `.text`, `.data`, and `.bss` sections of relocatable (i.e. unlinked `.o`) object files, `size` does not report the amount of space that a given `.o` file has requested in "common" at all. Thus, it is more difficult to make programming decisions about storage allocation when "common" allocation of uninitialized data objects is used. So for this reason also, you should avoid common allocation of uninitialized data objects.

Assembly Language Impacts

Most existing assembly language source files that were previously assembled with the GNU assembler should require no changes. There are changes, however, in the assembly language `.align`, `.balign`, and `.p2align` directives. In previous LynxOS releases, these directives were not truly effective if the requested alignment was greater than some particular small value (for example, 8). Now however, Lynx has extended the definitions of the COFF object file format somewhat (and has made relevant changes to the assembler and linker), so that the `.align`, `.balign`, and `.p2align` directives should now be effective even when large power-of-two alignments are requested. Please note that the LynxOS extensions to the COFF object file format only support section alignments that are exact powers of two. Thus, directives such as:

```
.align     13
.balign    13
```

are not allowed or supported, whereas directives such as:

```
.align     16
.balign    16
```

are now supported by the GNU assembler and linker on LynxOS.

Assembly language sources that use the `.bss` and/or `.lcomm` directives may now take advantage of new support for an (optional) third argument. This argument can be supplied in these directives to specify required alignment of the allocated space. We recommend that you supply this new alignment argument to insure that the alignments of all relevant data objects are neither smaller nor larger than what is required for the data objects in question.

NOTE: *The default alignments used in cases where no alignment factor is explicitly specified may now be different in some cases and for some objects.*

In cases where no explicit alignment is specified in a `.bss` or `.lcomm` directive, the following default alignment rules will be applied:

- If the object being allocated is larger than 8 bytes and if its size is also a multiple of 8 bytes, then the object will be aligned to an 8 byte boundary.
- If the size of the object is larger than 4 and is also a perfect multiple of 4, then the object will be aligned to a 4 byte boundary.
- If the size of the object is larger than 2 and a perfect multiple of 2, then the object will be aligned to a 2 byte boundary.
- If the size of the object does not fit any of the above criteria, then the object will be unaligned.

Changes to .align on PowerPC and 68K

On the PowerPC and 68k platforms, previous releases of the GNU assembler (`gas`) interpreted the values given in `.align` directives as powers of two. Thus, when assembling an assembly language source file for the PowerPC or 68k platforms, a directive such as `.align 3` was taken to mean that alignment to an 8 byte boundary was being requested. The interpretation of the parameter value given in `.align` directives has now been standardized across all LynxOS supported platforms, and values given in `.align` directives are now taken to represent themselves, rather than their corresponding powers of two. Thus, to request alignment to a 8 byte boundary, you should now write `.align 8`, regardless of LynxOS supported platform you are coding for.

This change of semantics may require some manual modifications for some existing assembly language source files which have previously been developed for LynxOS for PowerPC or LynxOS for 68k. This change of assembly-level semantics should not have any effect upon existing higher level language programs (e.g. those written in C or C++) because the compilers for those languages have already been modified to take account of this change.

CAUTION! *Be aware that C and/or C++ source files with embedded* `asm()` *constructs may need to be updated if they contain obsolete* `.align` *directives.*

LynxOS -D__NO_AP Compiler Option (All Platforms)

NOTE: *We strongly recommend that you do not use this option when compiling application code.*

The command line option **`-D__NO_AP`** is an internal Lynx option that has the intended effect of preprocessing specially declared function prototypes by removing their formal argument lists and permitting the functions to appear as traditional-C prototypes. Ultimately, by removing the formal argument list from a prototype, the compiler no longer performs type-checking nor type-conversions between the actual and formal arguments of a function call. This can lead to serious errors in an application which relies on the compiler to generate implicit type-conversions for parameter passing, since a type-conversion may result in a physical restructuring of the bit-representation of a parameter.

CHAPTER 4

Miscellaneous Notes

This section describes miscellaneous notes about this release, subdivided by platform where applicable.

All Platforms

Simple Kernel Debugger (SKDB) Characteristics

The Lynx SKDB has the following characteristics by design:

- The *default* SKDB port specifies the port to which SKDB reports a crash by default. Other SKDB-ready ports still accept their respective hot keys to invoke unless the entire SKDB is disabled (uninstalled).
- To break into SKDB with a hot key from a port, the port must be explicitly opened by a process.

Using a PS/2 Mouse with X (X86 and PowerPC)

If you plan to use a PS/2 mouse connected to a mouse port, you should install the PS/2 mouse driver in LynxOS before installing X. To install the PS/2 mouse driver on a LynxOS 2.5.0 system, login as root and run the utility `Install.ps2mouse`.

```
/usr/bin/Install.ps2mouse
```

X & Motif Development Package

Motif Demo Programs

Motif 2.0 demo program sources are available as a gzipped `tar` file in *installdir*`/usr/lib/X11/demos`.

Motif Documentation

Motif 2.0 documentation has been incorporated into the manual page distribution. You can use the `xman` utility to view the available documents.

LynxOS for X86

Configuring the SMC EtherEZ Card for LynxOS

For the SMC 8416 (EtherEZ) card to work with LynxOS, the Plug-and-Play software on the card must be disabled. This can be done by using the EZstart utility that is provided by SMC. Also, you must enable the "memory-mapped" feature.

- On some types of PCI machines, the settings within the SMC card are lost between cold boots. To resolve this, disable the Plug-and-Play boot feature by running the LynxOS `ezsetup` utility with the **`-nopnp`** option.
- The **`pnpboot`** option in the LynxOS `ezsetup` utility must be set to "enabled" for the card to work in a non-PNP machine.

NFS with 3C509A Ethernet Controller

The 3C509 Rev. A Ethernet controller has only 4 KB of on-board DRAM. By default, the LynxOS NFS client driver uses 8 KB for transmit and receive buffers. This difference in access size results in some unpredictable behavior of NFS-mounted file systems.

To specify a smaller transmit and receive buffer size for the NFS client driver, use the following options to the `mount` command.

`mount -o rsize=1536,wsize=1536` *filesystem directory*

LynxOS for PowerPC

Default Console

A PowerPC system needs either a VGA terminal connected to a video port or a terminal connected to the serial (COM1) port to server as the console. By default, LynxOS performs an auto-detection test for the VGA terminal at boot-up. If a VGA terminal is not found, the COM1 port will become the console.

By default, the console terminal type is “dumb” when it comes up. However, it can be set to the appropriate terminal type using an `export` command or by modifying the `/etc/ttys` file.

For example, to use `export`:

```
export TERM=vt100at
```

Or, to modify `/etc/ttys` for the same terminal type as the console:

```
vi /etc/ttys
# default for console
# /dev/con:1:default96:dumb:/bin/login
# change console to have no logins and set correct type
/dev/con:0:default96:vt100at:/bin/login
/dev/com1:0:default96:wyse50:/bin/login
/dev/com2:0:default96:wyse50:/bin/login
# enable logins on the VGA port
/dev/atc0:1:default:vt100at:/bin/login
```

Enforce In Order Execution of I/O

Applications doing I/O should use the instruction `eieio` to prevent the processor from re-ordering reads and writes to memory mapped I/O. This mnemonic expands to “enforce in order execution of I/O”.

The `eieio` instruction can be added using an in-line assembly statement in the application source code. For example:

```
asm("eieio");
```

Device drivers doing I/O also should have the `eieio` instruction in their source code.

IDE Support

The IDE device driver (`hd`) is included but not supported in LynxOS for PowerPC.

Network Booting

When using the netboot facilities of LynxOS, you must disable the PReP-style netboot in the PPCbug. This setting is a query in the `env` command.

LynxOS for 68K

Floating Point Emulator

A floating point emulator handles CPUs that lack an FPU, such as the 68LC040. On CPUs with an FPU, such as the 68040 and 68060, the emulator, although present, does not get used. For correct operation of the emulator on the 68LC040, you must be using a recent version of the 68LC040 chip that includes a critical fix in the processor to support the “F-Line” traps on which the emulator fundamentally depends. Silicon mask versions 3E23G and later are confirmed to include this fix. If you have a MVME162 board without this mask version and you need floating point emulation, Lynx suggests that you contact your board supplier to obtain an upgrade of the processor on the board.

Interphase Hawk Ethernet

On the newer Interphase Hawk Ethernet boards currently being shipped by Motorola, there is a new jumper (J16) that *must* be removed for proper operation of the hardware. You must also set the following parameters in the bug monitor:

- Short I/O (VMEbus A16) Enable must be set to:

  ```
  Yes
  ```

- Short I/O (VMEbus A16) Control must be set to:

  ```
  0x5 (D16 transfers | supervisor mode)
  ```

Configurable DMA Parameters

The burst length of a DMA cycle and the choice to do cache line bursts are configurable parameters of the MVME 162/167/177 on-board SCSI interface. These parameters are selected via the low-level SCSI device structure initialized in `/sys/devices/llscsiinfo.c`.

When performing DMA, the NCR chip always releases the memory bus for four cycles after each burst. The chip will then do a burst DMA of one cycle, two cycles, four cycles, or eight cycles. The corresponding burst length values are 0, 1, 2, 3. Setting the DMA for a short burst length will improve the real-time performance of the system by reducing memory cycle stealing from the CPU over a given period of time. In contrast, increasing the burst rate may improve SCSI device throughput for very fast devices. In many cases, however, no throughput change will be noticed with a shorter burst cycle.

The other selectable feature of SCSI DMA is the “cache line fill” mode. The cache line fill field is initialized in `/sys/devices/llscsiinfo.c` and should be 1 to enable or 0 to disable cache line fill. This mode may provide better memory bus utilization for a given burst length. However, when cache line fill is enabled, the only allowable burst length is four cycles (burst length value of 2). If you wish to use a burst length other than four cycles (value 2), you must initialize the cache line fill field to 0. Please contact Lynx Technical Support for assistance with building the `/sys/devices` directory.

Troubleshooting

These tips may help you troubleshoot problems with your LynxOS system.

1. *Whenever creating new device nodes or rebuilding the kernel, you should immediately reboot the system with the appropriate options.*

   Device file entries must match the executing kernel. Building a new kernel modifies device files immediately, though the system will be executing the previous kernel. You must reboot the system to have it function properly.

2. *Check your SCSI termination*

   We have encountered situations in which disks that have seemed to fail under LynxOS were actually incorrectly terminated. The final device on a SCSI cable must be appropriately terminated. On x86 systems, this is especially critical for the Adaptec 1542C controller which seems to be highly susceptible to line noise.

CHAPTER 5

Known Problems

This section lists known outstanding issues with this release.

Installation Issues

Correct System Date Must Be Set Before Installing

Before installing LynxOS with the `installit` utility, make sure that the system date is correct. If you are running a previous version of LynxOS, use the date command to do this:

`date` *YYMMDDhhmm*

Booting from CD-ROM on MVME177

We have encountered a problem with booting from CD-ROM on the MVME177. Use the Installation Boot Tape to boot LynxOS and begin the `installit` utility. Select CD-ROM as the distribution media for installation.

Preboot Problems on MVME2600 Series

LynxOS `preboot` does not disable the CPU cache on MVME2600 systems, resulting in errors during booting. On these systems, `preboot` can load itself from CD-ROM or disk, but then it fails to boot LynxOS.

To work around this problem, select an unused SCSI ID and set the boot device to this device at the `preboot` prompt. (This step seems to flush the CPU cache.)

```
Command? b s3b
```

Error messages will be displayed to say that this is invalid device. Now, select the correct SCSI ID and partition of the disk you want to boot. For example, to boot the CD-ROM distribution during initial installation, enter the following at the `preboot` prompt:

```
Command? b s6
```

After installing LynxOS on the disk, you will need to follow the same procedure to boot from it. For example, if you have installed LynxOS on partition A of SCSI disk 0, specify the following boot device *after* first setting the invalid device that flushes the cache.

```
Command? b s0a
```

If you have any difficulties with these steps, please contact Lynx Technical Support.

Setting Boot and Root Devices with preboot on 68K Systems

The LynxOS `preboot` utility "remembers" the previous root and boot device and will boot that device during installation. To complete the installation, you need to use `preboot` commands to set the root and boot devices after the system reboots. To illustrate these steps, this section describes installing LynxOS from a CD-ROM drive at SCSI ID 6 to partition A of a SCSI disk at ID 0. The example assumes that you have a previous version of LynxOS on disk 0.

1. Reboot the system and insert the LynxOS CD-ROM into the drive.

   ```
   reboot -a
   ```

2. Enter the VMEbug prompt and boot the LynxOS CD-ROM.

   ```
   xxx-Bug> bo 0 60
   ```

3. At the preboot prompt, set the root and boot device for LynxOS.

   ```
   Command? R 0 6
   Command? b s6
   ```

4. Proceed through the product installation. When the system reboots, you'll need to do similar steps to specify the disk. These are summarized below:

   ```
   xxx-Bug> bo 0 0
   Command? R 0 16
   Command? b s0a
   ```

See the *LynxOS Installation Guide* and the `preboot(1)` manual page for more information about booting LynxOS.

Benign Error Message

When choosing products to install from the `installit` main menu, please ignore the following error message that you may see on your screen:

```
[: too many arguments
```

Install.XM

If, by mistake, you choose option 3) Install X to run from a mounted CD, the script will install this option even if you have returned to the main menu and choose another option. You must exit and run `Install.XM` again to change your selection to another option.

Memory Limitations on 4 MB Systems

The automated installation utility provided in LynxOS 2.5.0 requires a minimum of 4 MB of physical memory. However, this is not enough memory to install all of the menu options. You *must* choose to install Virtual Memory to have `installit` correctly install LynxOS on a 4 MB system.

All LynxOS Systems

File System Check (fsck) Needs /lost+found Directory

After installing LynxOS, create a `/lost+found` directory. This directory is needed by `fsck(1)`.

```
mkdir /lost+found
```

GNU Debugger (gdb)

Unable to Read Core Files from Cross

In cross debugging environments (SunOS, Solaris, and AIX), `gdb` is unable to recognize the format of the core file. Please contact Lynx Technical Support for more information.

Tcl/Tk Graphical Interface for GDB

The Tcl/Tk graphical interface to `gdb` is not working properly in this release. We recommend that you do not use it. To disable this feature, you must unset the `DISPLAY` environment variable before running `gdb`.

If you are using the `bash` shell:

```
unset DISPLAY
```

If you are using the `csh` shell:

```
unsetenv DISPLAY
```

SCMP Script Problem (PowerPC and 68K)

When executing `make config` in `/sys/romkit/scmp_boot`, the resultant configuration file, `netboot.config-1`, sets the variable `clientip_hex` incorrectly. This is due to an error on line 69 in the `Config-Netboot` script. To work around this problem, you can either modify the script or calculate the hexadecimal conversion by hand.

To modify the script, edit the file `/sys/romkit/scmp_boot`. Go the function `decip2hexip()`. Change the variable `hex_ip` to `_hex_ip` as shown below.

```
decip2hexip()
{
     _hex_ip=""
     _decip=`echo $1 | tr '.' ' '`
     for _digit in $_decip; do
             decimal_to_hex $_digit _retval
# The following line is incorrect so comment out
#            hex_ip="$_hex_ip$_retval"
# Copy previous line and change hex_ip to _hex_ip
             _hex_ip="$_hex_ip$_retval"
     done
```

Alternatively, calculate the hexadecimal value for the decimal IP address and enter it in by hand into the configuration file.

Shared Libraries

LynxOS provides the `mkshlib` utility for making shareable libraries on x86, 68k, and SPARC systems.

LynxOS for x86 and 68k ships with X11R6 shared libraries. However, the X11R6 shared libraries for SPARC currently are not available. Please contact Lynx Technical Support for more information.

STREAMS Implementation of TCP Endpoints

The implementation of TCP-based STREAMS endpoint has the following limitation—if an endpoint is put in asynchronous mode and a connect is tried to a non-existing peer endpoint, the TCP Protocol Control Block is released. You need to close this endpoint and retry the connection later.

Swap Files for Virtual Memory

If you are using virtual memory (VM) with a swapfile, do not change the size of this file or remove it while VM is enabled. To change the size of the swapfile, follow these steps (assuming that the current swapfile is called `/.swap`):

1. Create a new contiguous file.

   ```
   mkcontig /.swap64 64m
   ```

2. Edit `/bin/rc` to use this new swapfile.

   ```
   vi /bin/rc
   # old swapfile
   # prio 25 vmstart /.swap
   # new swapfile
   prio 25 vmstart /.swap64
   ```

3. Reboot LynxOS.

   ```
   reboot -a
   ```

4. Now the new swapfile will be active. At this time, you can remove the old swapfile `/.swap`.

   ```
   rm ./swap
   ```

C Shell Limitation

The C shell (`tcsh`) command `limit cputime number` does not work because LynxOS does not support limiting a process' CPU time or memory usage.

Terminal Output with the Bash Shell

Different from other shells with which you may be more familiar, the bash shell, if run in the background, will display job output without carriage returns. For instance, if you run an `ls` command in the background:

```
ls &
```

you will get output that looks as if it were indiscriminately spewed across the screen. To work around this problem, use one of the other shells: `csh` or `dlsh`. Alternatively, do not run jobs with output in the background. This problem is cosmetic only and does not result if you redirect output to a file.

LynxOS for PowerPC

Virtual Memory

CAUTION! *A late-breaking problem with Virtual Memory (VM) was discovered in this release. If you need to use VM, please contact Lynx Technical Support for more information.*

Compiler Issues

Symbol Resolution by the Linker for "end"

The following code fragment exposes a problem with the 96Q1 linker on the PowerPC:

```
int a = 1;
main(){
}

end(){
}
```

The following error is printed:

```
collect2: ld returned 1 exit status
```

The problem is that the linker does not correctly handle the symbol resolution for procedure end. To work around this problem, rename the procedure _end.

GNU Profiler (gprof)

The gprof profiler does not properly handle the tracking of time samples outside the program being profiled. As a consequence, the statistics for <locore> (addresses below the bounds of the program) and <hicore> (addresses above the bounds of the program) should be ignored.

Compiling X and Motif Programs

If the linker fails when compiling X and Motif programs, increase the Data limit value in /etc/starttab. Reboot the system and relink the program.

Legacy Linker

When creating executables on the PowerPC, the legacy linker (`l-ld`) does not honor the user's `umask`. Instead all executables are created with the permissions 0777. If you would like the permissions to be more restrictive, use the standard LynxOS `chmod` command to change the permissions. See the `chmod` manual page for more information on this command.

CONFIG.TBL Instructions

Comments within the file `/sys/lynx.os/CONFIG.TBL` provide instructions for archiving driver `.o` files into `drivlib.a`. This is incorrect. Because of the new `/sys` structure and the use of `.a` archive libraries that already contain driver `.o` files, no special processing outside of `CONFIG.TBL` is needed to install or remove a driver and relink a kernel.

Kernel Linking Problem

NOTE: *If you are adding new device drivers to LynxOS for PowerPC, we recommend that you modify the kernel Makefile as described in this section.*

When linking the kernel, the **`-r`** option of the linker is used. This can cause problems if an unresolved reference is encountered during the linking process. Because the **`-r`** flag specifies a relocatable object, the linker will not report the unresolved reference. When attempting to boot a kernel that has an unresolved reference, it is most likely that a kernel crash will occur when the system starts up (a reference to this unresolved symbol will be unpredictable).

To fix this problem, modify `/sys/lynx.os/Makefile.common` as shown below.

```
cd /sys/lynx.os
cp Makefile.common Makefile.common.orig
vi Makefile.common

# Original link lines as shipped
#$(KERN_LINK) -o a.out $(DEPENDENCIES) $(TIMESTAMP_OFILE) \
#               $(KERNEL) $(SYSCALLS) $(AUX) $(OTHER)
#
# Replacement link lines to fix problem
ld -r -o tmp.a.out $(FIRST_OFILE_PATCHES) $(OFILES)
$(PATCHES) \
$(STUBS) $(TIMESTAMP_OFILE)

$(filter-out -r, $(KERN_LINK)) -o a.out tmp.a.out $(SYSCALLS)
\
$(KERNEL) $(DEVICES) $(CUSTOMER_DRIVERS) $(DRIVERS) $(TCPIP)
\
```

```
$(NFS_SERVER) $(NFS_CLIENT) $(STREAMS) $(HYBRID) $(AUX)
$(OTHER) \
$(SYSCALLS) $(KERNEL) $(AUX)
```

These changes ensure that the linker will report undefined references when building LynxOS kernels.

LynxOS and AIX

- LynxOS cannot coexist with AIX on the same hard disk.
- The LynxOS kernel may crash if an AIX executable is run under LynxOS. Incompatible link line options between LynxOS and AIX executables can cause this situation.
- The AIX cross development system is supported only on PowerPC-based AIX systems. That is, AIX systems running on Power or Power2 architecture processors are not supported.

MVME2600 Series

Ethernet Driver Autonegotiation Mode

To enable the **`AUTONEGOTIATION`** mode for the controller chip on a MVME2600-01 with the 761-001 transition module, update the `dec` Ethernet driver as described in this section.

1. Edit the file `/sys/drivers/if_dec21040/if_dec21040.c`.
   ```
   cd /sys/drivers/if_dec21040
   vi if_dec21040.c

   # add following line above dcreset() in dcinstall()
   sc->flags |= DCAUTONEGOTIATE;
   dcreset();
   ```
2. Rebuild the driver.
   ```
   make install
   ```
3. Rebuild the kernel.
   ```
   cd /sys/lynx.os
   make install
   ```
4. Reboot LynxOS.
   ```
   reboot -a
   ```

NOTE: *The function* `dcreset()` *is called from other places in this driver; the fix must be made to the* `dcreset()` *that is inside the* `dcinstall()` *function.*

Ethernet Driver Automatic Station ID Detection

If the station ID is not automatically detected by the `dec` Ethernet driver, you will need to modify it as described in this section.

1. Edit the file `/sys/devices/if_dec21040info.c`.

   ```
   cd /sys/devices
   vi if_dec21040info.c
   ```

2. In this file, change the `dcinfo` structure as follows:

   ```
   struct dcinfo dc0_info = {
     0, /* 0=AUI;1=TP;2=100 Mbit (DEC21140);4=BNC */
     1,        /* manual ethernet address search */
     {aa,bb,cc,dd,ee,ff}        /* input station ID */
      0                                /* pci bus */
   };
   ```

   where `{`*aa*`,`*bb*`,`*cc*`,`*dd*`,`*ee*`,`*ff*`}` is the Ethernet address (in hexadecimal notation) of the station ID from the label on the 2600 board.

3. Rebuild the device.

   ```
   make all
   ```

4. Rebuild the kernel.

   ```
   cd /sys/lynx.os
   make install
   ```

5. Reboot LynxOS.

   ```
   reboot -a
   ```

Booting from Disk

If you have difficulty booting, please see "Preboot Problems on MVME2600 Series" on page 39 for more information.

Level 2 Cache

LynxOS currently does not enable the Level 2 (L2) cache on MVME260x VME processor boards. Please contact Lynx Technical Support for more information.

PPCbug Firmware Required

If you have a MVME2600 board with Open Firmware, you should contact Motorola to get PPCbug firmware.

System May Hang After Reboot

You may encounter a system hang at the PPCbug prompt after rebooting LynxOS. To work around this problem, use the `reset` command of the PPCbug firmware.

```
PPC1-Bug> reset
Cold/Warm Reset [C,W] = C?
Execute Local SCSI Bus Reset [Y,N] = N? Y
Execute Local (CPU) Reset [Y,N] = N? Y
```

Now boot LynxOS with the `pboot` command. Refer to the Motorola PPCbug Firmware documentation for more information about these commands.

LynxOS for 68K

Virtual Memory

CAUTION! *A late-breaking problem with Virtual Memory (VM) was discovered in this release. If you need to use VM, please contact Lynx Technical Support for more information.*

Booting from CD-ROM on MVME177

We have encountered a problem with booting from CD-ROM on the MVME177. This problem appears to be specific to booting only; all other CD-ROM functionality works as expected on the MVME177.

Creating Boot Media

When making boot media with the scripts in `/sys/romkit/installation`, be aware that if SKDB is installed, the resultant boot media will be specific to the hardware target on which you have made the media and will not boot on other 68k variants. To avoid this problem, we recommend that you uninstall SKDB before creating boot media.

GNU Debugger (gdb)

Unable to Read Core Files

On native 68k systems, when `gdb` is used to read a core file, a segmentation fault will occur. Please contact Lynx Technical Support for more information.

Unexpected Behavior on Applications Using recvfrom

Lynx recommends that you do not use `gdb` to debug applications that use the `recvfrom` system call on the 68k. When using `gdb` to debug an application that calls `recvfrom`, the application's behavior may change. Data from the socket may never be received when run under `gdb`.

LynxOS for SPARC

Shared Library Utility Needs $ENV_PREFIX

When building shared libraries with the `mkshlib` utility, you may encounter an invocation failure of `mkshlib` if the environment variable `$ENV_PREFIX` is not set. To work around this problem, set `$ENV_PREFIX` before running `mkshlib`.

At the `bash` shell prompt:

```
export ENV_PREFIX=
```

Or, at the `csh` shell prompt:

```
setenv ENV_PREFIX
```

X Windows Issues

Adapters with Fixed Frequency Clocks

If the `Xerrors` file contains the message "no defined matching dot clock for 1152x864@70Hz", your adapter has a fixed frequency clock chip that does not support that setting. However, another revision of the adapter might support it.

Display Power Management System Support

Only the following chipsets support Display Power Management System (DPMS):

- S3 864
- S3 964
- S3 trio32
- S3 trio64
- S3 trio 64V
- I128
- STORM
- MACH64

Dual Servers with Millennium Adapters

This feature is not working properly. However, you can use dual servers with a combination of Millennium and Diamond Stealth adapters.

Dynamically Loading Server Extensions

Most server extensions are built into the server. However, the following extensions can be loaded dynamically:

- XPRIO
- X3D-PEX
- DOUBLE-BUFFER

To dynamically load one of these extensions, edit the file `/usr/lib/X11/Metro/Extensions` and change the setting OFF to DYN for the extension.

Error Messages When Exiting the X Server

You can ignore the following error message that may appear when you exit the X server:

```
Couldn't load XKB keymap, falling back to pre-XKB keymap
```

This message will disappear if you select another keyboard mapping (other than the default) in `configX`.

Keyboard LEDs

Keyboard LEDs are not supported in this release. Please contact Lynx Technical Support for more information.

Motif Window Manager

- The `mwm` window manager may appear to lock up the screen. If this happens, press the **Esc** key to unlock the screen.
- The multiscreen option to `mwm` is not working.

  `mwm -multiscreen`

  You must start separate `mwm` clients for each screen in a multiple screen configuration.

Mouse Systems Compatible and configX

We have encountered a situation in which a Mouse Systems compatible mouse was not recognized by the `configX` utility. To work around this problem, first exit `configX` by pressing **Ctrl**-**Alt**-**Backspace**.

Now, run the `Install.XM` utility to re-install the Mouse Systems compatible mouse as shown in the steps below.

1. In the Main Menu, select 2) Choose Products to Install
2. In the Software Configuration Menu, set options 1-3 to **`n`**. Then select option 4) Advanced Installation Options.
3. In Advanced Installation Menu, set options 1-5 to **`n`** and set option 6) Run the X Server Configuration Utility to **`y`**.
4. Return to the main menu.
5. In the Main Menu, select 1) Modify Default Values.
6. In the hardware configuration Menu, select 3) Select Mouse Type for `configX`. Specify the type and connection for your mouse.
7. Return to the main menu.
8. Begin the "installation."

PosixWorks Desk Priority Conflict

The PosixWorks Desk may freeze due to a priority conflict with the X Server and the Motif window manager. If this happens, you must reboot the system. Please contact Lynx Technical Support for more information.

Recognizing Amount of RAM on Adapter

The X server may not recognize the exact amount of RAM on the video adapter. This may cause the display on the screen to appear distorted. To fix this problem, specify the amount of available video RAM in the file `/usr/lib/X11/Metro/Xconfig`. For example:

```
Driver       SVGA
VideoRam     1024
```

Switching Between PosixWorks Desk and the Default X Environment

To switch between PosixWorks Desk and the default X environment, run `Install.XM` and follow the steps below.

1. In the Main Menu, select option 2) Products to Install.
2. In the Software Configuration Menu, select **n** for options 1 and 2. If you want to enable the PosixWorks Desk, select **y** for option 3; otherwise, select **n** for option 3.

Video Adapters on X86 Systems

ATI Mach Adapters

- Some ATI Mach64 and S3 adapters have hardware limitations which prevent the use of certain modes you may have selected.
- Resolutions with colors higher than 256 currently do not work with Mach 32 adapters.

Diamond Adapters

The Diamond Viper P9000 adapter at a resolution of 800x600 with 256 colors at a resolution 72 Hz may display some jitter.

ET4000 Adapters

If you have trouble with text restore on ET4000 adapters, add the following “Option” line after “Driver” in `/usr/lib/X11/Metro/Xconfig`.

```
Driver      SVGA
Option      ET4000Text
```

I128 Adapters

The latest versions of the I128 adapters are not yet supported. Please contact Lynx Technical Support for more information.

MGA Impression

MGA Impression VLB adapters do not support the 16-bpp color depth. Selecting this configuration will result in a screen with a green hue.

Number Nine Adapters

- To obtain 16 Million colors from a Number Nine GXE 64 PRO graphics adapter, there must be at least 4 MB of memory on the adapter.
- The Number Nine GXE Level 14 sets the screen color to green on exit.

OAK Adapters

OAK OTI87 has a known problem where the root weave is drawn incorrectly the first time the adapter is run in VGA mode. This does not adversely affect operation and will be corrected in a future update.

Video 7 Adapters

Video7 ERGO II adapters may not refresh the screen when switching display resolutions dynamically. Please run the `xrefresh` utility to manually refresh the screen.

Virtual Resolution and Switching

Virtual Resolution and Resolution Switching are not supported on some adapters. Although `configX` allows you to select multiple resolutions with these adapters, only the highest resolution selected will be implemented. To change to a different resolution, go back to `configX` and select the desired resolution; toggle off all others. Some adapters affected by this are:

- Diamond Viper
- Matrox Impression Plus
- Matrox Marvel I
- Western Digital

X Server Fails to Execute (PowerPC)

If the X server fails to execute, see if the following lines are in the file `/usr/lib/X11/Metro/Xconfig`:

```
Chipset       STDVGA
Clocks        25     28
```

If so, comment them out and try running the X server again.

```
#Chipset       STDVGA
#Clocks        25     28
```

Documentation

ftruncate Manual Page

`ftruncate(2)` cannot extend a disk file larger than its current size. The present LynxOS implementation of POSIX.1b shared memory does not allow a shared memory region to be resized with `ftruncate()` after it has been mapped by a process. Attempts to do this will be silently ignored.

mmap Manual Page

You must set either `MAP_PRIVATE` or `MAP_SHARED`; otherwise, `EINVAL` will be returned.

Network Snooping with tcpdump

To use the `tcpdump` utility, you must first configure the LynxOS kernel with the binary packet filter driver (`bpf`).

1. Edit the `CONFIG.TBL` file.

   ```
   cd /sys/lynx.os
   vi CONFIG.TBL
   ```

   At the end of this file, add the following line for the `bpf` driver:

   ```
   I:bpf.cfg
   ```

2. Rebuild the kernel.

   ```
   make install
   ```

3. Reboot LynxOS and build new device nodes.

   ```
   reboot -aN
   ```

See `tcpdump(1)` and `bpf(4)` for more information.

TCP/IP Protocol Stack Manual Page

The manual page for the TCP/IP protocol stack, `hbtcpip(4)`, was not included in this release. If you plan to tune the TCP/IP protocol stack, please contact Lynx Technical Support for more information.

CHAPTER 6

Defect Fixes in LynxOS 2.5.0

The following table lists known user-visible product defects that have been fixed since the LynxOS 2.4.0 release.

Platform	Subcomponent	ID	Summary
All	.exrc	9075	The .exrc file is missing from product.
All	.exrc	9096	vi prints out some annoying messages when started.
All	/usr/include	9273	symlinks cause problems in /usr/include.
All	Cross Development	8932	Missing windex files for manpages.
All	Device Driver Development Guide	8800	Rephrase from “have restricted access to OS structs”
All	Device Driver Development Guide	8828	Page 23 (2.3.0) refers to getphys instead of get_phys.
All	Device Driver Development Guide	8862	Missing device access information (Sect 5.7) from former PPC Supplement
All	ENVIRONMENT	4487	COFF_INIT is wrong for linking COFF programs.
All	Ethernet (All ethernet drivers)	9005	Problem in the Promiscuous mode
All	Install.nfs	7589	Installing with out server comments out portmapper.
All	Install.nfs	8541	System hangs if client support is retained by Uninstall.nfs
All	Install.nfs	9687	Adding nfs client doesn’t update rc_network correctly.
All	Install.skdb	9463	Wrong com? numbers displayed in choice menu
All	Installation Guide - LynxOS	9119	Chapter 6 inappropriately refers to a “Cross Development” manual.
All	Installation Guide - LynxOS	9136	Guide doesn’t say that virtual memory must be installed on systems with < 4 MB RAM.
All	Installation Guide - LynxOS	9173	Appendix A is inaccurate on ethernet support.
All	Installation Scripts	8459	installit creates / (root) with 0777 permissions
All	Kernel Configuration Guide	8573	Errors in commands for netboot support on SPARC, section 3.2
All	Kernel Configuration Guide	8633	SCMP chapter p87, wrong path name for bplane driver.

Platform	Subcomponent	ID	Summary
All	Kernel Configuration Guide	8942	param.h manpage should reflect all the modifications.
All	Kernel Configuration Guide	9250	Netboot instructions for PPC should state “enable real-time clock.”
All	Kernel Configuration Guide	9275	Update “Reconfiguring the Kernel” chapters to reflect /sys changes.
All	Kernel Configuration Guide	9572	Kernel Conf Guide should warn against using ranlib on kernel libs
All	Kernel Configuration Guide	9640	Re-write SCMP chapter based on changes to /sys and scripts.
All	OBJ_RULES	7783	App Guide c++ compiles for uSparc cant use -D__NO_AP.
All	Porting Kit Guide	8588	Update PKIT Guide
All	Programmers Guide	9682	Intro.3 manpage refers to man/cate. Should be /src/examples.
All	Reference (See Description)	9278	Remove references that say “Not supported on SPARC.”
All	SCMP	9566	Update scmp-related manpages with changes to framework.
All	Source Code Build Guide	8929	Adjustments required if source not loaded in “/”
All	_getpgrp()	6429	_getpgrp.9 manpage incorrectly says can be used in interrupt routine.
All	_kill	9126	The diagnostics are not documented in the _kill.9 manpage.
All	a2ps	9483	a2ps gives a random exit status upon success.
All	all	9158	Installation from a mounted CD-ROM is not supported.
All	bash	8692	bash.1 manpage has problems.
All	chread.c	8821	Spurious parameter in call to sismine().
All	chroot	7931	chroot() is not secure.
All	clock.c	8445	A bad VME address can cause fault handler to loop infinitely.
All	csh	8941	No manpage for csh.1 (tcsh.1 manpage only shows additional features over csh).
All	df	8627	In the “df” utility, check_mount() returns address of local variable.
All	df	9698	df should show the same info for monuted CD’s on all platforms.
All	diff	7153	GNU diff sometimes thinks two identical binary files are different.
All	dr_install()	8930	dr_install() doesn’t install raw interface along with block driver
All	ekill()	6876	ekill.2 manpage needs work on how signal is delivered.
All	enable_write_through	9579	When write_through is enabled, boot process hangs after “Mounting all filesystems”.
All	ethernet	8021	ether.4 manpage contains bogus header files.
All	evtpoll	8869	nano seconds are treated like micro seconds in kernel, evtpoll.

Platform	Subcomponent	ID	Summary
All	exit.c	8584	Some processes get killed randomly with lots of process creation and deletion.
All	ezsetup	8628	In the ezsetup utility: unsigned comparison for less than zero.
All	fdopen()	6360	fdopen() and fdclose() have memory leaks.
All	file locking	9608	NFS file locking locks up the inode in the server
All	file system	8322	NFS exporting a FLFS, massive file reads may lose data; if NFS readonly mode, ok!
All	floating-point	8700	Add support for user-changeable and user-readable FPU status/control.
All	flook	9339	flook() is not consistent between the platforms.
All	floppy	7489	Open of raw floppy (x86) succeeds even when the door is open.
All	fsck	8959	fsck must make sure directory entries are 4-byte aligned.
All	fstab	9664	Improve fstab warning message on system startup
All	ftime()	9257	In time.3 manpage, there is a typo for “millitm.”
All	gcc	9568	Add information about $ENV_PREFIX to gcc.1 manpage.
All	gdb (96q1 cygnus)	9726	native gdb doesn’t read core files on m68k, sparc, x86
All	gdb (cygnus)	8590	Forwarding a signal to thread in evtpoll() runs signal handler.
All	gdb (cygnus)	8600	Attached process requires “continue” in gdb twice to actually continue.
All	gdb (cygnus)	9496	Example program runs differently under gdb
All	getblk()	9440	Stress conditions appear to cause wheels to spin
All	getdents()	8856	getdents() displays bogus entries.
All	getdirpath()	8695	getdirpath() needs to depend on block size instead of hardcoded 128.
All	getgrent()	4043	Incorrect synopsis for getgrent()/getpwent.()
All	getpriority()	8503	_getpriority.9 manpage says that there are only 32 possible priorities
All	getstid()	7962	Rename manpage getstid.9 to st_getstid.9.
All	gettimeofday()	9326	gettimeofday() manpage has bogus note.
All	gmon.h	8912	gmon.h redefines ROUNDUP/ROUNDDOWN from st.h.
All	gnutar	8475	gnutar.1 manpage missing.
All	gprof (GNU)	9399	Profiler is not comptible with muti-threaded programs
All	hbtcpip	4981	socket() call does not return proper error code.
All	hbtcpip	5123	System crashes when UNIX domain socket is left from the previous crash.
All	hbtcpip	8965	tcp/ip has infinite ACK problem.
All	hbtcpip	9459	more tunable parameters for tcp/ip protocol stack
All	hbtcpip	9490	Raw ethernet packets have ether type in host order (on recv)
All	hbtcpip	9495	tcpip does not generate SIGIO

Platform	Subcomponent	ID	Summary
All	hbtcpip	9648	TCP/IP packet corruption
All	if_dec21040	9124	Calls to mem_map and mem_unmap with PHYSBASE addresses do not work.
All	in.h	9307	Unterminated strings in in.h causes compilation errors
All	inode_open()	9336	permissions are NOT being checked in 2.5.0 Beta
All	installation	9461	Make network installation scalable (rc.network and sysconfig)
All	installationScript	8461	X11R5 and Motif tar tapes have some files with absolute path names.
All	iso9660	9040	iso9660_read_inode() interprets time incorrectly.
All	iso9660	9734	iso9660 filesystem goes into infinite loop on a CD-ROM
All	kill	9238	/bin/kill does not recognize SIGUSR1 or SIGUSR2.
All	ld (epc linker)	8526	Missing ld.1 manpage for PowerPC in the 2.4.0 FCS build.
All	ld (gnu/cygnus)	9445	Change ld.1 manpage now that ld defaults to ‘-no-keep-memory’
All	ld (gnu/cygnus)	9569	Add description of no_init_section option in ld.1 manpage.
All	libr (or ar)	6212	libr truncate archive entry names to 16 characters. Document this in manpage.
All	local_param.h	9123	Add local_param.h and local_conf.h to sys/lynx.os.
All	ls	9271	“ls -s” does not show 512-byte blocks - as per man page.
All	makeboot	8572	Non-root users should not be able to run makeboot.
All	makeboot	8803	makeboot.1 manpage needs a platform-independent file pointing to the four platform-specific variants.
All	makeboot	9489	Make changes to manpages to support common makeboot.
All	makeboot	9650	makeboot on the ppc was not able to make a romkernel bootable.
All	makekdi	8804	makekdi.1 manpage is incorrectly named as makekdi_ppc.
All	man	9286	Man doesn’t work for secondary names like sbrk, getppid, etc
All	man	9478	Incorrect links in cat1 directory for 121396 build D.
All	message queues	8545	mq_selective_receive() can trash queue prio pointers.
All	migrate to gcc	9110	Rename manpages for legacy Lynx compiler tools.
All	mkfs	7622	mkfs does not check that device is a block device.
All	mkimage	9580	mkimage is creating incorrect free block bitmaps
All	mkimage	9587	mkimage symlink and link parameters are not honoring character case
All	mkisofs	8621	Check in the “mkisofs” utility.
All	mkisofs	8758	mkisofs does not write a symlink correctly.
All	mkisofs	9391	mkisofs prepends a _ to iso9660 directory names.

Platform	Subcomponent	ID	Summary
All	mkpart	9247	When running mkpart using rsh, input is requested before output is done.
All	mkromk	7477	Macro above_gap incorrect in mkromk (makefs.c).
All	mmcon()	9141	mmcon() interface is faulty.
All	monitor()	5357	monitor(3) manpage refers to vprof(1).
All	mount	7710	Usage message is incorrect for mount, ‘n’ option should be ‘q’ option.
All	mq_send	9329	mq_send() fails with invalid args on ppc, works fine on x86.
All	mutex calls	9388	killing proc that is blocked on a mutex can crash the system
All	netstat	9628	“netstat -i” does not lookup host in /etc/hosts, with active resolv.conf
All	nfs	7078	“SETATTR” on NFS locked files should not be cached.
All	nfs	9350	Writes from SunOS to a NFS mounted Lynx file truncates it to 0
All	nfssvc	9507	directory reading (of ‘magic’ size) fails over NFS
All	open()	5797	open() has a problem with FIFO opened in non blocking mode.
All	p1b_aio.c	8538	POSIX.1b asynchronous I/O has cancellation problems.
All	permissions	8635	setrootfs and makeboot need r-x------ permissions.
All	ping	8867	ping doesn’t work in the uSPARC kernel.
All	pipes	9292	SIGPIPE should be sent to the thread doing the write
All	pmsg.c	8957	heavy traffic in msgpackets with MSG_COPY can panic with spagefault.
All	posix.1	6451	streams/t_bind returns non-defined error coded causes application crash.
All	preboot	8618	Enhancement # 8237 - preboot.1 manpage needs to be changed.
All	preboot	9341	user cannot build preboot, Makefile problem
All	preboot	9449	Fix typo in booting example of preboot.1 manpage.
All	preboot	9575	Addition of preboot to PPC requires documentation review and changes.
All	process stack size	7686	The stack size can exceed limits defined in /etc/starttab for threads.
All	proto.h	8842	The proto for sem_wait() collides with the proto in semaphore.h.
All	ps	9521	‘ps ax’ reports the same number for ‘text’ as for ‘data’.
All	pselect()	9112	Select on FIFO behaves incorrectly when there is no writer.
All	ptrace()	6168	ptrace() does not validate address space is not supervisor stack.
All	ptrace()	8605	Single stepping thru sigaction() crashes kernel.
All	ptrace()	8696	ptrace() can leave scheduling off.
All	read()	8409	Block and raw drivers are not posix compliant for read().

Platform	Subcomponent	ID	Summary
All	reboot	8477	Missing “-” option for reboot.1 manpage.
All	reboot	8620	Enhancement # 8237 - reboot.1 manpage should be updated
All	recoset()	9679	GNU optimizer doesnt know dangers of recoset()
All	reltimer()	7801	Event handler receives wrong args from POSIX interval timer.
All	rlogind	8899	rlogind and telnetd fails tcsetpgrp().
All	rlogind	9009	rlogind no longer honors .rhosts
All	routed	9560	routed removes certain routing entries (incl slip and scmp)
All	routed	9561	routed broadcasts RIP packets to syslog port.
All	rpc	8808	rpc.3 manpage name is named incorrectly as rpclib.
All	scanf()	7585	scanf.3 manpage refers to unlocked_scanf(), which is not provided.
All	scanf()	8717	Improve accuracy of scanf() floating-point numbers.
All	scheduling	9511	Bug in the FIFO Scheduler
All	sem_wait()	7148	Program which receives signal in sem_wait cannot be killed.
All	sem_wait()	8763	sem_wait.2 (Lynx) manpage is incorrect.
All	semaphore.h	8547	semaphore.h and pthread.h define sem_wait() prototype differently.
All	semaphore.h	9420	semaphore.h needs to be compilable by g++
All	sems	8946	sems is not working.
All	send()	7496	send() can be interrupted by a signal after part of data transmitted.
All	services	9576	WWW and DNS not listed in /etc/services
All	setrootfs	8805	setrootfs.1 manpage is incorrectly named as setrootfs_ppc.
All	setscmpid	8806	setscmpid.1 manpage is incorrectly named as setscmpid_ppc.
All	setup_inode()	8780	Detect errors in reading inodes from disk.
All	sh (obsolete)	5853	sh returns incorrect value with command like variable=`cmd`.
All	shm_open()	8513	shm_open.2 manpage misleads regarding the return value.
All	shm_open()	8845	shm_open.2 manpage reports incorrect return values.
All	sigaction()	7118	On the 68K, sigaction() manpage with sa_flags = SV_SYSCONTEXT fails.
All	sigaction()	8770	sigaction.2 manpage states the wrong parameters of the signal handling function.
All	sigwait()	9164	sigwait() behaves incorrectly when event handlers are present.
All	slattach	9555	Running SLIP at 115200 can cause: panic mfree
All	slip	9549	When “slip” is selected, NFS hangs during bootup; ^C interrupts

Platform	Subcomponent	ID	Summary
All	snmp	9540	List of defects in the manpage of SNMP
All	socket()	8486	socket.2 manpage doesn't document AF_UNIX.
All	sqrt()	8494	For sqrt(), improve performance on the SPARC and accuracy on the 68k.
All	ssignal()	7135	The ssignal.9 manpage is incorrect.
All	st.h	8200	There are warnings when when getstid() is compiled.
All	st_switch()	9398	st_switch.2 manpage has bogus reference to fast_*().
All	strcmp()	8405	strcmp(), strncmp(), and memcmp() must do unsigned comparison
All	streams	8570	There is potential corruption of streams scheduler list when freeing a Queue.
All	streams	8721	Incorrect STREAMS buffer usage percentage calculations.
All	strncpy()	8403	If s2 is shorter than s1 an extra byte is written.
All	support for posix1b	8984	POSIX.1b objects don't {f}chmod, {f}chown
All	synch_invalidate()	8794	synch_invalidate() calls mem_unlock() with only 3 arguments.
All	syslog()	8718	syslog() seems to ignore setlogmask().
All	sysppc	9500	sysppc.2 manpage uses SIG_EXC_SIGNALLING instead of FP_EXC_SIGNALLING
All	t_alloc()	8817	Improper behavior if T_ALL is used and service provider errs
All	t_connect	9310	t_connect call hangs indefinitely
All	t_snd()	9006	t_snd() on non-blocking fd causes loss of data
All	t_sndudata()	8809	t_sndudata.3 manpage is named incorrectly as t_sndudada.
All	timeout()	8689	timeout() should return SYSERR when out of timeouts.
All	torom	8807	torom.1 manpage needs a platform-independent file pointing to the two specific variants.
All	tr	8240	tr will not replace '-' if '-' is not the first in the from list.
All	tty	7081	tty drivers manpage incorrectly documents setting async flag.
All	ttyp	6628	Master pseudo tty loses data when slave closes.
All	uparam.h	9386	uparam.h needs things added to match param.h.
All	uparam.h	9629	INSTALL_SKDB parameter does not have any effect in uparam.h
All	usleep()	8188	POSIX usleep() fails to return for larger intervals.
All	utils	9632	tr test fails
All	uvmedrvr	9570	Add manpages for uvmedrvr and vmedrvr device drivers (PPC).
All	vi	8954	Berkely nvi needs to be ported to LynxOS
All	vi	9282	"new" vi -t doesn't work, nor does :t function within vi.
All	vi	9315	"new" vi is really annoying about arrow keys over the network

Platform	Subcomponent	ID	Summary
All	vi.lynx (obsolete)	9007	vi goes run-bound in substitute command freezing the system
All	vm.c	7327	rml_count should be unsigned.
All	vmount	7053	Rename /etc/mount to /etc/vmount for System V usage.
All	wait()	8760	wait.2 manpage incorrectly orders the arguments.
All	xdm	8703	When the last client from xdm exits, xdm will crash the system.
PowerPC	Cross Development	8678	Link fails in cross dev if -pg (profile) is used for the gcc.
PowerPC	Cross Development	8679	gmon.out not created when prog is compiled with -pg options.
PowerPC	Install.skdb	8682	Install.skdb should provide for VGA console on powerpc as well.
PowerPC	Installation Scripts	8558	makeboot needed at the end of the installation scripts.
PowerPC	Mouse Support	8691	The middle button on the PS2 mouse does nothing.
PowerPC	PCI Probe	9593	Dec driver fails to install because of PCI Bus fault
PowerPC	PowerPC	9499	User program can hang PowerPC kernel
PowerPC	atc	8639	shell “read” in .profile on VGA console will hang system in single user mode.
PowerPC	atc	9042	Video controller grabs 16MB mem space reserved for ISA on ppc.
PowerPC	atc	9381	console keyboard hangs frequently on Powerstacks.
PowerPC	bdlan	9591	Amd driver for CES platform fails when user IP address blank
PowerPC	compiler	9633	cc1 on Powerpc has a 5mb BSS section.
PowerPC	conf.c	8556	There are warning messages when conf.c (devc_table array) is compiled.
PowerPC	devcon	9068	I didn’t make the devcon requirements clear enough.
PowerPC	dr_install()	9432	Dynamic driver installion fails
PowerPC	drinstall	8925	Installing Dynamic drivers with globals will crash PPC.
PowerPC	dynamic loading	8444	Dynamic driver loading fails on 2.4.0 PowerPC.
PowerPC	file	8916	/etc/magic on PPC needs an entry for lib*.a.
PowerPC	g++ (cygnus)	8455	PowerPC g++ problems: _g_config.h, libg++.a, “-liostream”.
PowerPC	g++ (gnu)	9240	Constructors are not called when the program starts.
PowerPC	gcc	8106	Sunos x and PPC need a full pathname to as.
PowerPC	gcc	9457	gcc produces FP reg access for structure passing
PowerPC	hd (ide driver)	7911	The ide driver does not work on all PPC boards.
PowerPC	hd (ide driver)	9324	hdinfo.c and hdinfos.c missing for ppc in /sys/devices/Makefile
PowerPC	if_dec21040	8664	Network performance degrades if a machine let idle for 2 days.
PowerPC	if_dec21040	8892	endian problem on PPC version of 21041 driver
PowerPC	install scripts	8564	installit modifies <param.h> but not <sys/param.h>.
PowerPC	make	8675	make unnecessarily recompiles source files

Platform	Subcomponent	ID	Summary
PowerPC	mmu	7473	Bad code in mmuinit.c; end is not assigned properly.
PowerPC	mount	8833	mount /usr/dl (without 2nd param) dumped core.
PowerPC	ptr (printer driver)	8827	ioctl for PPC printer is short an arg.
PowerPC	ptrace()	8143	gdb step mode crashes the system on user buserror.
PowerPC	ptrace()	9159	There are bad kernel memory accesses when in gdb.
PowerPC	rlog	8372	rlog dumps core
PowerPC	signal()	8779	Sample program which handles SIGFPE does not allow new network connections
PowerPC	skdb	9434	skdb is not called on a crash for ppc 2.5.0 Beta2
PowerPC	stdarg.h	8549	Simple C program with variable arguments fails to compile on PPC.
PowerPC	sysbrk()	8154	kernel sysbrk memory can be written by user appl.
PowerPC	syscall.c	9095	The check against num_syscalls should be >=, not >.
PowerPC	t_shm.c	9438	mkshm() crashed the system in power PC
PowerPC	tty	9047	tmgr_close() call missing second argument in ttydrvr.c
PowerPC	vme_getintr_vec()	8066	vme_getintr_vec returns bogus values for vectors > 0x79.
SPARC	MMU handling stuff	9054	Virtual memory on uSPARC is very unpredictable/unstable.
SPARC	MMU handling stuff	9055	Running Byte Benchmark shows memory leak with VM
SPARC	boot process	9603	usparc reboot fails every other time
SPARC	execve.c	8773	execve() for the SPARC needs to be more discriminating.
SPARC	floppy	8634	Floppy driver on uSparc can lock up the system.
SPARC	gcc	7871	cdk gcc for the uSparc lacks support for udiv & sdiv.
SPARC	gdb (cygnus)	7823	gdb can't find automatic variable defined in a block inside a function.
SPARC	gdb (cygnus)	8180	gdb function evaluation makes messes on uSparc.
SPARC	general	9602	rcp of large file to usparc machine causes kernel panic
SPARC	gethostent()	9053	gethostent() core dumps on sparc.
SPARC	hbtcpip	8866	uSPARC hbtcpip thread hangs when xntpd started.
SPARC	in.h	8837	in_makeaddr() returns the wrong type for uSPARC.
SPARC	load.c	8716	Loading does not insure the break address is properly aligned.
SPARC	mkshlib	9298	mkshlib needs to pull mulsparc.o, etc, from libc.
SPARC	romkit examples	9644	romkit example makefile and spec files on usparc does not work
SPARC	stdlib.h	9004	There is white space (indented) on line 43 that should not be there
SPARC	stubs	8171	The uSPARC distribution is missing some stub files.
x86	3COM 3C509	8518	Ethernet board delay, “100000 loop” insufficient for faster Pentiums.
x86	3COM 3C509	8579	New (3Com) 509B cards with PnP options do not work
x86	Installation Guide - X/Motif	6163	Install instructions fail for floppy install.

Platform	Subcomponent	ID	Summary
x86	Installation Guide - X/Motif	7192	removal of Xm, uil and Mrm not mentioned in documentation.
x86	Installation Scripts	8528	Virtual memory configuration menu needs to validate argument.
x86	Mouse Support	6971	Mouse Systems compatible trackball mouse does not work.
x86	Mouse Support	8659	Touchcalibrate is broken.
x86	NCR77C22E	6806	Problems with configuration of NCR driver R2.2.7.
x86	NCR77C22E	6807	Graphics Screen is corrupted with version 2.2.7 of NCR drvier.
x86	Xman	6107	Xman does not read the mandesc file.
x86	all (all/most utilities effected)	8263	All /bin utilities are unreadable by others than owner (root).
x86	as	5406	There is no /bin/as.
x86	atc	8952	num lock doesn't function.
x86	cdwrite (cd-r utility)	8811	cdwrite has a bug in determining first writable address.
x86	cdwrite (cd-r utility)	8814	cdwrite does not restore full speed to the CD-R when it's done.
x86	diff	7613	Cygnus diff was built with wrong header files (loses first 2 chars of name of file).
x86	disk cache	7769	Increasing cache blocks corrupts file system when tcp/NFS is installed.
x86	ezsetup	8229	ezsetup hangs the system if run on an EISA system.
x86	fast ada	5308	Fast ada reliability crashes the system.
x86	file locking	7879	NFS file locking fails.
x86	floating point	8550	pow() sends SIGFPE on overflow.
x86	floppy	7643	Cannot umount a write protected floppy after a write.
x86	hbtcpip	9597	IP-forwarding and routing do not work
x86	hd (ide driver)	8082	Faulty IDE drive hangs Lynx, DOS says bad disk.
x86	hd (ide driver)	8374	Add LBA and phoneix BIOS translation to ide driver.
x86	hd (ide driver)	8824	ide driver does not work with two models of ATAPI drives.
x86	hd (ide driver)	8826	ide driver is not processing blocks in priority order.
x86	hd (ide driver)	8851	If "intolerant_drive" is = 1, then page faults can occur at slevel 3.
x86	if_dec21040	9022	More than 3 etherpower boards on a system hangs NFS
x86	if_dec21040	9482	AUI port does not work on a SMC etherpower COMBO
x86	if_dec21040	9564	if_dec21040 does only looks at pci bus 0 for any netword cards
x86	if_wd3e	9716	if_wd3e crashes running raw ethernet test
x86	integration build	8580	missing files from dheaders in product
x86	ll-scsi	8787	scsi53c7x0.c rev 1.16 - references eieio() on x86.
x86	ll-scsi	9098	scsi53c7x0.c prints out error message when doing warm boot.

Platform	Subcomponent	ID	Summary
x86	makeboot	8788	makeboot depended on something opendir did to work correctly.
x86	missing drivers	9536	ps2 mousedrvr.o is missing and Install.ps2 script fails
x86	mkpart	7582	mkpart will create a type 1 partition if used as documented.
x86	pcxe (Digi-board driver)	6632	Last 5 channels fail on input for a 16 port card .
x86	pcxe (Digi-board driver)	7743	Code to set port in raw mode fails to clear ixon/ixoff for Digiboard.
x86	pcxe (Digi-board driver)	7896	The Digiboard has H/W flow control problems.
x86	pipes	8449	Quitting out of program | more can hang both programs.
x86	preboot	8617	2.4.0 preboot fails to boot kernel on Intel Zappa M/B with 2940.
x86	printf()	9224	printf() fails for long doubles.
x86	ptr (printer driver)	1667	Printer loses data when placed off line.
x86	ptr (printer driver)	8076	HP inkjet 660 does not work with LynxOS.
x86	scsimgr.c	8185	An inquiry issued before the device may be ready can cause open() to fail.
x86	scsimgr.c	8408	scsimgr_get_blocks() fails with out-of-limit blocks.
x86	scsimgr.c	8426	scsimgr tape handling at Filemark/Blank Check is wrong.
x86	scsimgr.c	9355	1542 crashes when accessing an external hard disk
x86	skdb	9245	x86 skdb doesn't trace individual threads.
x86	strip	9598	The utility /bin/strip does not work with a.out files.
x86	syslog()	9614	calls to openlog() cause core dump
x86	ttymgr.c	8195	X1_MODEM_OK is not set if LNOMDM is clear.
x86	types.h	7105	There is a K&R style prototype of malloc() in rpc/types.h; it should be ANSI.
x86	z(Obsolete) gdb (gnu)	6626	Multi-threaded program run under gdb will cause it to go run-bound.
68k	Mintape	6321	mintape is not documented.
68k	bash	9051	bash logs out when running command in the background
68k	floating point	8216	FP emulation on the 68060 has problems - gives core dumps
68k	floating point	9182	vme162 68k boards will not boot 2.5.0 beta.
68k	gdb (96q1 cygnus)	9139	96q1-m68k-GDB gives core dump when it is invoked.
68k	gdb (cygnus)	7334	gdb print command doesn't work correctly with global variables; manpage needs update.
68k	gnutar	9643	Downloading a huge tar file to MVME162 makes the system down.
68k	interrupt.h	7478	TIME_LEVEL and SCSCI_LEVEL declared inconsistently in interrupt.h.
68k	main.c	9081	Kernel does not come up if the NPROC is set to 300.

Platform	Subcomponent	ID	Summary
68k	mtscsi.c	8156	Tape I/O for larger blocking factor fails after a while with errno 22.
68k	porting kit	9601	Romtool and rkboot do not support DRAMBASE != 0
68k	preboot	8485	preboot.1 manpage missing info about the “S” command to preboot.
68k	profiler	8474	simple prog with profiling cores with segment violation.
68k	real-time performance	9735	Real time performance problems on 2.5.0 (68k, vme177)
68k	sh (obsolete)	6281	sh does incorrect command substitution within double quotes.
68k	tty1x7	8115	Need Minimal modem control support.
68k	ttymgr.c	5851	Upon receiving XOFF the tty manager does not stop transmission.