Tru64 UNIX

Technical Overview for Version 5.1A

Part Number: AA-RH8XD-TE

June 2001

Product Version: Tru64 UNIX Version 5.1A or higher

This document describes components of the Compaq Tru64 UNIX operating system.

Compaq Computer Corporation
Houston, Texas

Contents

3 Networking

4 File Systems

5 Kernel, Symmetric Multiprocessing, NUMA, Virtual Memory, and Device Support

Tables

About This Manual

This manual provides a brief overview of the Compaq Tru64™ UNIX operating system and its components.

Audience

This manual is for anyone who is interested in the components that make up Tru64 UNIX.

Organization

This manual is organized as follows:

Chapter 1	Provides an overview of the Tru64 UNIX operating system.
Chapter 2	Describes the various features for system management.
Chapter 3	Describes the available networking protocols and applications.
Chapter 4	Describes the various file systems supported by Tru64 UNIX.
Chapter 5	Describes issues regarding the kernel, focusing on symmetric multiprocessing, virtual memory, and device support.
Chapter 6	Highlights the major features provided by the Tru64 UNIX development environment.
Chapter 7	Describes the Tru64 UNIX applications that will help you achieve seamless Windows NT interaction with your UNIX system.
Chapter 8	Describes the Common Desktop Environment (CDE) and the X Window System Motif components of the Tru64 UNIX windowing environment.
Chapter 9	Describes how Tru64 UNIX addresses the requirements of the C2 evaluation class and additional security features.
Chapter 10	Describes internationalization features.
Appendix A	Lists RFC and non-RFC standards to which Tru64 UNIX complies.
Glossary	Provides definitions for common UNIX and general computing terms that are used in the Tru64 UNIX environment.

Related Information

- QuickSpecs (formerly called *Software Product Descriptions* or SPDs) provide technical information about Compaq products. The *Tru64 UNIX*

Operating System QuickSpec and QuickSpecs for several separately licensed components are located on the Operating System CD–ROM in the *mount_point*/DOCUMENTATION directory, where *mount_point* is the directory in which the CD–ROM is mounted.

- The Tru64 UNIX documentation set is available on the Tru64 UNIX Documentation CD–ROM and on the Web. For information about the documentation, see the *Documentation Overview*. To view the documentation on the Web, go to the following site:

  http://www.tru64unix.compaq.com/docs

- This Technical Overview refers to various Request for Comments (RFCs). RFCs define the Internet protocols and policies; they are available from numerous sites including the following:

  http://www.ietf.cnri.reston.va.us/home.html

  http://www.cis.ohio-state.edu/hypertext/information/rfc.html

  When viewing this *Technical Overview* on line on a system connected to the Internet, you can click on an RFC to bring up the text of that RFC in your Web browser.

- Compaq ActiveAnswers offers a Web-based, comprehensive solution program, which includes several utilities that can be used for planning, deploying, operating, and maintaining key applications. These utilities assist applications in many areas including, but not limited to, management, database and business applications, high availability, and Internet solutions.

  The main purpose of ActiveAnswers is to accelerate the time to solution while also positioning Compaq as a key enterprise solution provider. Partnerships have been made with key Independent Software Vendors (ISVs) such as Microsoft, SAP, Novell, Oracle, and Genesys. Web-based solutions for customers and system integrators have also been developed. You can visit the ActiveAnswers Web page at the following Web site:

  http://www.compaq.com/activeanswers/

- The Compaq homepage is located at the following site:

  http://www.compaq.com/

- The Compaq Tru64 UNIX homepage is located at the following site:

  http://www.tru64unix.compaq.com/

Reader's Comments

Compaq welcomes any comments and suggestions you have on this and other Tru64 UNIX manuals.

You can send your comments in the following ways:

- Fax: 603-884-0120 Attn: UBPG Publications, ZKO3-3/Y32
- Internet electronic mail: `readers_comment@zk3.dec.com`

  A Reader's Comment form is located on your system in the following location:

  `/usr/doc/readers_comment.txt`

Please include the following information along with your comments:

- The full title of the manual and the order number. (The order number appears on the title page of printed and PDF versions of a manual.)
- The section numbers and page numbers of the information on which you are commenting.
- The version of Tru64 UNIX that you are using.
- If known, the type of processor that is running the Tru64 UNIX software.

The Tru64 UNIX Publications group cannot respond to system problems or technical support inquiries. Please address technical questions to your local system vendor or to the appropriate Compaq technical support office. Information provided with the software media explains how to send problem reports to Compaq.

Conventions

This manual uses the following conventions:

`file`	Italic (slanted) type indicates variable values, placeholders, and function argument names.
`cat`(1)	A cross-reference to a reference page includes the appropriate section number in parentheses. For example, `cat`(1) indicates that you can find information on the `cat` command in Section 1 of the reference pages.
MB/s	This symbol indicates megabytes per second.
Mb/s	This symbol indicates megabits per second.
Glossary Terms	In the online version of this document, various terms are linked to the *Glossary of Common UNIX and General Computer Terms*. By clicking on the term, you will be taken to its definition. You can easily return to the place you were reading by clicking on your browser's Back button.

1

Introduction to Tru64 UNIX

This chapter introduces the Tru64 UNIX operating system and provides a description of its features. Many of the components listed in this chapter are described in more detail in later chapters. The following topics are discussed:

- Major features of the operating system, its history from its OSF/1 genesis, and a summary of the standards to which it conforms (Section 1.1)
- Product features and enhancements that support enterprise applications, high performance technical computing, and Internet communications (Section 1.2)
- Optional components that can help meet your computing needs (Section 1.3)
- Components of the operating system and how they are packaged (Section 1.4)

1.1 Overview

The Tru64 UNIX operating system delivers many features that raise it to the highest level of performance, scalability, and availability, yet make it simple to manage and operate. For users seeking to expand their information technology capabilities while increasing performance and maintaining the highest level of availability, Compaq offers the best industry UNIX solution.

Compaq Tru64 UNIX is the most time-tested 64-bit UNIX operating system you can buy. It brings new strengths, with features that:

- Dramatically simplify system management by offering a choice of management interfaces
- Substantially reduce the complexity of installation, setup, and management
- Reduce the total cost of ownership by offering familiar interfaces, utilities with a common look and feel, and automation — thereby minimizing the need for training
- Grow to multiple terabyte configurations, giving users the flexibility to satisfy their growing business needs
- Increase performance in file system, storage management, and system networking

- Deliver high integration between UNIX and Windows 2000 (see Section 7.3)
- Fill users' highest availability needs with the addition of Compaq TruCluster Server Software.

Tru64 UNIX is supported on all Alpha servers and workstations with 64 MB of memory.

The operating system incorporates several performance enhancements either developed or extended by Compaq, including wired memory, virtual memory, and Unified Buffer Cache; UNIX File System (UFS) file block clustering and cached writes over the Network File System (NFS); IPv4 multicasting, path MTU discovery, and optimized TCP/IP; and quick-started shared libraries.

With Tru64 UNIX you have a clear and concise system administration environment (including graphics, Web-based, and character-cell) that does the following:

- Simplifies your system administration tasks
- Enables an update installation that does not overwrite system files, or performs a full installation that quickly gets your system up and running
- Supports loadable drivers and other kernel subsystems, including loadable boot-path support for third-party disks and graphics cards
- Provides virtually automatic management of peripheral SCSI storage devices, such as disks, tapes, and CD–ROMs

Tru64 UNIX provides real-time support and symmetric multiprocessing (SMP), dataless servers and clients, and numerous features to assist application programmers in developing applications that use shared libraries, threads, and memory-mapped files. It is fully compliant to the Single UNIX Specification, to the X/Open UNIX brand, to POSIX 1003.1b (real time) and to POSIX 1003.1c (with POSIX Threads — formerly called DECthreads).

The Common Desktop Environment (CDE) is the default user interface.

1.1.1 Product History

Tru64 UNIX is the Compaq implementation of the Open Software Foundation Version 1.0 and Version 1.2 technology and the Motif Version 1.2.5 graphical user interface and programming environment. In addition, Tru64 UNIX supports the full features of the X Window System, Version 11, Release 6.3 (X11R6.3).

The Tru64 UNIX operating system is a multiuser/multitasking, 64-bit, advanced kernel architecture based on Carnegie Mellon University's Mach Version 2.5 kernel design with components from Berkeley Software

Distribution (BSD) Versions 4.3 and 4.4, UNIX System Laboratories System V Release 4.0, other software sources, the public domain, and Compaq Computer Corporation.

Earlier versions of Tru64 UNIX were known as DIGITAL UNIX. It began its existence with the name DEC OSF/1.

1.1.2 Standards

With the Version 5.1A release, Tru64 UNIX complies with the UNIX 98 Product Standard, a significantly enhanced version of the UNIX 95 product standard.

UNIX 98 enhancements include the following:

- Threads interfaces
- Multibyte support extension (MSE)
- Large file support
- Dynamic linking
- Changes to remove hardware data-length dependencies and restrictions
- Year 2000 changes

UNIX 98 is made up of the following product standards:

- Internationalized System Calls and Libraries Extended V2
- Commands and Utilities V3
- C Language
- Sockets V2
- Transport Service (XTI) V2
- Internationalized Terminal Interfaces (XCurses)

The UNIX 98 Conformance Statement Questionnaire for Tru64 UNIX is provided on The Open Group Web site at the following site:

`http://www.opengroup.org/csq/`

Tru64 UNIX does not support the optional enhancements to the UNIX 98 Product Standard, such as software administration facilities and a set of APIs for real-time support.

This version of the operating system also complies with the UNIX 98 Workstation Product Standard, which is the same as the UNIX 98 Product Standard, but with the additional requirement of conforming to the Common Desktop Environment Product Standard.

The CDE standard defines the X/Open Common Desktop Environment, a common graphical user interface environment for use on systems supporting the X Window System. This standard defines a standard set of functional capabilities and supporting infrastructure, as well as the associated standard application programming interfaces, command-line actions, data interchange formats, and protocols that must be supported by a conformant system. It provides standard forms of the facilities normally found in a graphical user interface environment, including windowing and window management, session management, file management, electronic mail, text editing, calendar and appointments management, calculator, application building and integration services, print job services, and a help service.

The operating system is compatible with the Berkeley 4.3 and System V programming interfaces and, by complying with the System V Interface Definition (SVID3 Base and Kernel Extensions), supports System V applications as well.

See Appendix A and the *Tru64 UNIX Operating System QuickSpec* (formerly called *Software Product Description*) for a list of standards that Tru64 UNIX supports.

1.2 Product Features and Enhancements

Tru64 UNIX offers an array of significant features and enhancements to support enterprise applications, high performance technical computing, business intelligence, and Internet communications. Most of these features are incorporated into the base operating system, while others are incorporated into the optional components and separately licensed products packaged on the Tru64 UNIX Associated Products CD–ROMs. (See Section 1.4 for information about the Tru64 UNIX media kit.)

The following sections describe features of the operating system, as well as some new features and enhancements provided in the Version 5.1A release.

1.2.1 Scalability

The easy replacement and maintenance of CPUs and the ability to configure mixed-speed CPUs are among the scalability features supported in the Version 5.1A release:

- CPU online addition and removal (OLAR)

  Version 5.1A introduces OLAR technology, which allows you to replace or upgrade CPUs on your AlphaServer GS160 and GS320 systems without needing to bring your system down. With this new feature, also referred to as “CPU hot swap,” you can replace CPUs with faster CPUs or replace a broken or malfunctioning CPU even as your system remains operational.

The Compaq Analyze utility now includes the ability to notify the operating system that a specific CPU should be replaced. You can configure the operating system to automatically request that the CPU go off line, though allowing you to define when to honor these automatic off-line requests. This feature keeps the system running longer because it can disable faulty CPUs before they cause the entire system to stop. The OLAR indictment capacity requires the installation of Compaq Analyze from the WEBES (Web-Based Enterprises Services) kit.

OLAR management is integrated with the SysMan (see Section 2.4) suite of system management applications, which provides the ability to manage all aspects of the system from a centralized location.

For information about OLAR, see the new *Managing Online Addition and Removal* manual.

- Support for AlphaServer GS mixed speed CPUs

  With the introduction of mixed-speed CPU support you can purchase additional processors for your AlphaServer GS80, GS160, and GS320 systems with limited regard for the speed the processors currently have.

  The AlphaServer GS series of computers enables you to quickly scale from 1 to 32 CPUs with up to 256 GB of memory and to run multiple instances of different operating systems.

- NUMA (non-uniform memory access)

  The GS80, GS160, and GS320 AlphaServer platforms use non-uniform memory access (NUMA) architecture to enable a multiprocessor environment that can scale to large numbers of CPUs. Starting with Tru64 UNIX Version 5.1A, the operating system software is NUMA aware, to ensure that legacy applications run with as close to optimal performance as possible without source-code modification.

  In addition, NUMA APIs are available for use in large and complex applications that require more direct management of CPU and memory access on a NUMA platform.

  For more information on NUMA, see Section 5.3.

- Compaq Capacity on Demand

  Introduced in the Version 5.1 release, the Compaq Capacity on Demand (CCoD) program lets you order a system with added CPUs for expansion, then pay for the extra capacity later when you need it. Because the CPUs are already loaded, you can activate them without rebooting your system.

  Version 5.1A supports CCoD, which is available to AlphaServer GS140E, GS160, and GS320 customers who are purchasing new systems, adding new Alpha EV67 CPUs, or upgrading to Alpha EV67 (E2067-DA) 6/700 CPUs.

The CCoD program is also available to the installed base 8200/8400 and GS60E customers who are planning to upgrade their systems to EV67 technology.

CCoD kits are available on a CD–ROM and from the World Wide Web. You can find additional information and download the CCoD kit from the following Web site:

http://www.compaq.com/alphaserver/cod

- Very large memory (VLM) and large storage capacities

  Tru64 UNIX supports memory utilization to 28 GB, thereby providing more efficient use of memory in VLM configurations. It supports file and storage systems of 4 TB or larger.

- Device support
  - The SCSI-3 standard supported by the operating system allows up to 256 target and LUN addresses per SCSI bus and supports dual pathing (alternate path to a device).
  - The device naming model provides a flexible framework that allows more than 256 device names per SCSI bus. This model supports SCSI-3 and Fibre Channel.
  - Tru64 UNIX supports Fibre Channel-switched connections and multiple concurrent paths (64) with adaptive load balancing. Both SCSI-2 and SCSI-3 commands are supported.

- Dynamic Tuning

  Most tuning can be done at run time without rebooting the system.

1.2.2 Features of the Installation Processes

You can install Tru64 UNIX in several ways, using various utilities to help make the process as simple as possible. The following list describes the installation processes available to you; each is described in more detail in Section 2.1.

- Full Installation

  Installs a new operating system. A Full Installation creates new file systems and swap space and overwrites existing system and user-created files on the disk partitions where the file systems and swap spaces are to be installed. After the installation, you configure the system for general use. (See Section 2.1.1.)

- Update Installation

  Updates a system running Version 5.0A or Version 5.1 to Version 5.1A. This type of installation preserves disk partitions, file systems, file customizations, network and print environment, user accounts, user

created files, and any other system setup you may have done on the system. (See Section 2.1.2.)

- Installation Cloning

  Lets you replicate the Full Installation from a model system that is already installed onto one or more systems with the same or similar hardware configuration. Installation Cloning is ideal for environments in which there are many systems of the same type to be installed. Cloning produces identical system installations and eliminates the need to go through the Full Installation tasks at each system. (See Section 2.1.3.)

- Configuration Cloning

  Lets you replicate the network and print environment from an already configured system during a Full Installation. (See Section 2.1.4.)

- Rolling Upgrade

  In cluster environments, the TruCluster Server software supports the Rolling Upgrade procedure, which is a software upgrade of a cluster that is performed while the cluster is in operation. Clients accessing services are not aware that a rolling upgrade is in progress. (See Section 2.1.5.)

1.2.3 System and Network Management Features

Tru64 UNIX has a rich collection of features and provides many applications to help you easily set up, configure, and tune your system, and to simplify your daily maintenance, and administration tasks.

The following list highlights system and network management features. Chapter 2 provides more detailed information on many of these features.

- Managing tasks with SysMan Menu

  The SysMan Menu provides a framework for organizing various system management tasks. Each task represents a small application that is launched from the SysMan Menu.

  SysMan Menu can be run in CDE, HTML, or ASCII text environments. Thus, all the tasks on this menu can be performed from an X11 capable display, a personal computer running Microsoft Windows products, Linux, or the Macintosh Operating System, or from a character cell terminal. (See Section 2.4.1 for more information about SysMan Menu.)

- Managing tasks with SysMan Station

  The SysMan Station provides a graphic representation of the Tru64 UNIX system and enables you to manage it remotely from any computer. This Java utility is fully integrated with the TruCluster Server software. (See Section 2.4.2 for more information about SysMan Station.)

- Configuring your system with Quick Setup

Quick Setup provides a fast, user-friendly way to set up your system with the basic system configuration. The resulting system can be used "as is" or you can customize it with settings accessible in the full-featured configuration application. Quick Setup, which is accessible with a character cell interface, is available from the System Setup menu with the `/usr/sbin/setup` command. (See Section 2.2 for more information about Quick Setup.)

- Memory trolling

  Memory trolling is a process of reading the system's memory to proactively discover and handle memory errors. Tru64 UNIX supports memory trolling on AlphaServers GS80, GS160, and GS320 systems.

  You manage memory trolling with the `vm_troll_percent` attribute, which was introduced in the Version 5.0A release. You can use the `vm_troll_percent` attribute to enable, disable, and tune the trolling rate. You can change the rate at any time. (See the *Managing Online Addition and Removal* manual and the `memory_trolling`(5) reference page for more information about memory trolling.)

- Getting on the network with the Network Setup Wizard

  The Network Setup Wizard guides you through the process of adding a system to a network. The wizard steps you through the applications in a recommended order and provides information to help you determine which applications are applicable to each situation. (See the *Network Administration: Connections* manual and the SysMan Menu online help for information on the Network Setup Wizard.)

- Link aggregation

  Version 5.1A gives system administrators the ability to combine two or more physical Ethernet Network Interface Cards (NICs) to create what is called a "link aggregation group."

  The upper-layer software sees this link aggregation group as a single logical interface. Link aggregation (also called "trunking") provides the following capabilities:

  – Increased network bandwidth, depending on the number and type of NICs

  – Load distribution across all ports of a link aggregation group

  Link aggregation is supported only on server-to-server and server-to-switch point-to-point connections. (See Section 3.3.10 for more information.)

- Analyzing and managing events

  – The `collect` utility

The `collect` utility is a system monitoring tool that records or displays specific operating system data. With `collect`, you can include or exclude any set of the subsystems, such as file systems, message queue, tty, or header in data collection. You can display data at the terminal or store it in either a compressed or uncompressed data file. Data files can be read and manipulated from the command line, through use of command scripts, or through a graphical user interface.

– Compaq Analyze

  Compaq Analyze helps you analyze events stored in the system's event log file. This graphical interface is a rules-based, hardware fault-management diagnostic utility that enables you to set program and configuration parameters and to review event information. (See Section 2.5.2 for more information about Compaq Analyze.)

– EVM event manager

  The EVM event manager provides a single point of focus for the multiple channels (such as log files) through which system components report event and status information. (See Section 2.5.8 for more information about Event Manager.)

– DECevent

  The DECevent event management utility provides error reporting and binary-to-text translation capabilities. DECevent provides system-directed diagnostic capability for various platforms. (See Section 2.5.9 for more information about DECevent.)

- Monitoring systems with Compaq Insight Manager

  Compaq Insight Manager provides a Web-based environment from which you can monitor supported systems using Web-Based Enterprise Services (WEBES). Compaq Insight Manager provides a consistent wrapper for SysMan and other Tru64 UNIX utilities. (See Section 2.5.1 for more information about Compaq Insight Manager.)

- Other utilities of note

  Most of the following utilities are available from the Sysman Station, the SysMan Menu, the command line, and the Common Desktop Environment Application Manager. See the *System Administration* manual and the appropriate reference page for information on how to use these tools.

  – Hardware management (`hwmgr`)

    The `hwmgr` utility helps you manage hardware components, including disk and tape drives, processors, and buses. (See Section 2.7 for more information about `hwmgr`.)

- Environmental Monitoring

  The `envconfig` utility monitors the thermal, fan, and redundant power supply state of AlphaServer systems with prerequisite hardware sensor support. (See Section 2.5.7 for more information on environmental monitoring.)

- Analyzing system parameters (`sys_check`)

  The `sys_check` utility outputs the configuration, hardware and software, of the running system and displays the information in HTML format. It also performs a basic analysis of operating system parameters and attributes and provides warnings if it detects problems.

- Class Scheduler

  The Class Scheduler provides a convenient way to prioritize tasks.

- Kernel Tuner

  The Kernel Tuner lets you display and change parameters of the kernel subsystem. (See Section 5.1.1 for more information.)

- Process Tuner

  The Process Tuner lets you display, monitor, and manage system processes. It provides sort and filter options to help manage the display of the information.

- Optimizing code with `spike`

  The `spike` utility lets you optimize code after linking. It is a replacement for `om` and does similar optimizations. Because it can operate on an entire program, `spike` can do optimizations that cannot be done by the compiler. See the `spike(1)` reference page for more information.

- Division of Privileges

  The Division of Privileges utility enables administrators to grant users or groups access to privileged programs without the `root` password. All privileged system management applications can be launched from the SysMan Menu, the SysMan Station, the desktop's Application Manager, the Custom Setup checklist, or the command line. For more information, see the *Security* manual.

- Bootable tape

  The bootable tape feature gives you the ability to create and recover a disk image from a system. The bootable tape utility (`btcreate`) supports the Logical Storage Manager (LSM).

1.2.4 ARMTech Resource Management Software

Version 5.1A introduces support for Aurema's Resource Management Software products, which allow for richer and easier management of the resources needed to keep mission-critical applications running at peak performance.

Tru64 UNIX includes Aurema's ARMTech suite of resource management products on the Associated Products Volume 2 CD–ROM. Aurema's ShareExpress entry-level resource management utility is available license free. The enhanced Aurema resource management products, ShareExtra and ShareEnterprise, can be enabled by purchasing a license from Aurema.

ARMTech is a powerful system utility that enables dynamic allocation and balancing of CPU system resources.

The license-free ShareExpress product provides an enhanced UNIX timeshare scheduler offering equal sharing on a per-user basis. The ShareExtra product provides differential sharing of CPU resources. The ShareEnterprise product provides group-based sharing of CPU resources, storage of historical and accounting information, and other powerful resource management features.

1.2.5 UNIX and Microsoft Windows Interoperability

Tru64 UNIX provides a number of capabilities and products to simplify the development, deployment, and management of solutions in a UNIX and Microsoft Windows environment. These capabilities help solve integration problems across a number of functional areas. See Chapter 7 for details.

Windows 2000 Single Sign-On software gives Windows 2000 users the capability of logging in to a Tru64 UNIX system using their Windows 2000 user names and passwords. See Section 7.3 for more information about this optional component.

1.2.6 Advanced Printing Software

Advanced Printing Software is a printing system for Tru64 UNIX. It was developed in collaboration with Xerox and based on the PrintXchange technology from Xerox. Advanced Printing Software is a distributed client/server printing system for workgroup and enterprise environments.

Advanced Printing Software offers full-featured print spooling functions such as job scheduling, job retention, event notification, multiple levels of access control, print queue failover, and host transparency when used in a TruCluster Server environment.

It uses the Network Information Service (NIS) or the Lightweight Directory Access Protocol (LDAP) to distribute printer information throughout your environment and uses inbound and outbound gateways to interoperate with `lpd` print subsystems.

Advanced Printing Software is based on the print system model defined in the ISO 10175 Document Printing Application standard and the command set defined in the POSIX 1384.7 draft standard.

For more information, see the Advanced Printing Software *Release Notes* (included with the software subset) and the Advanced Printing Software *User Guide*, which is available on the Tru64 UNIX Documentation CD–ROM and on the Web.

1.2.7 System V Support

The System V habitat consists of alternate versions of commands, subroutines, and system calls that support the source code interfaces and run-time behavior for all components of the base system and kernel extensions as defined in the System V Interface Definition (SVID). Using the System V habitat lets you override the default system commands and functions with corresponding System V commands and functions (system calls and subroutines). The Tru64 UNIX System V habitat supports all SVID 2 and SVID 3 functions.

For information about the System V habitat, see the *Command and Shell User's Guide*.

1.2.8 Compaq Secure Web Server

The Compaq Secure Web Server included in the Version 5.1A release is based on the industry standard Apache Software Foundation (ASF) code base. Compaq improved the base ASF product by including Secure Sockets Layer (SSL) capability that allows for encryption up to 156 bits. This server also includes support for Java Servlets, Java Server Pages, and Hypertext Preprocessor (PHP).

1.2.9 iPlanet Directory Server

Tru64 UNIX includes the iPlanet Directory Server, which you can use as a general-purpose corporate directory or as the primary directory for Tru64 UNIX applications or other applications that need access to a directory. The software license is included with the Tru64 UNIX base license and is valid for up to 200,000 users. For more than 200,000 users, you must purchase an additional license.

1.2.10 Tru64 UNIX Open Source Software Collection

Included with the Tru64 UNIX Media Kit is the Open Source Software Collection CD–ROM, which supplies you with user and administration utilities along with multimedia and graphics packages that enable you to exploit the power of Tru64 UNIX systems. This software collection saves you the time and inconvenience of searching the Internet to locate software that has been qualified and tested on Compaq Tru64 UNIX.

In addition to an array of public domain freeware and shareware programs, the Open Source Software Collection disk includes tools developed by Compaq that are not for commercial sale and demonstration versions of commercial packages. (See also Section 1.3.7.1 for information about the Internet Express for Tru64 UNIX CD–ROM.)

1.2.11 Documentation

The Tru64 UNIX documentation gives you the information you need in the format that is most convenient for you. For complete information about the Tru64 UNIX documentation, including changes made in the Version 5.1A release, see the *Documentation Overview*. The following list describes some of the features of the Tru64 UNIX documentation:

- Most of the documentation is available in HTML format for viewing with a Web browser, in PDF format for viewing and printing with the Adobe Acrobat Reader, and in printed books.

  When viewing the books with a browser, references to other documentation — including the reference pages — are linked, so that a click of your mouse will take you there.

  With your browser's print option, you can easily print the sections you want, or you can cut and paste sections to create your own documentation to better help you perform specific tasks. With Acrobat Reader, you can print sections, chapters, or entire books with a near type-set quality.

- The Documentation CD–ROM, which contains HTML and PDF versions of Tru64 UNIX books, white papers, and the complete set of operating system reference pages, can be used on a Tru64 UNIX system, as well as on a Windows PC or Macintosh — laptop or workstation — or on any other system that uses the ISO 9660 Level 1 CD-ROM standard. This CD–ROM also provides versions of Adobe Acrobat Reader for many operating systems.

  To help you locate specific information in the documentation, the CD-ROM contains a copy of the AltaVista Search CD-ROM software with a complete index of the HTML documents. The AltaVista software runs on an x86-based PC with Windows 95, Windows 98, Windows 2000, or Windows NT Version 4.0.

The *Master Index* listing of the Online Documentation Library includes a *Master Index* search utility. When you type the word or phrase you want to find, the search utility searches all *Master Index* entries for matches and then displays the results. When you click on a resulting entry, the target book opens in its own window — in most cases to the place indicated by the entry. Figure 1–1 shows the results of a search for the text string "bsd."

Figure 1–1: Master Index Search Display

bridging drivers	BSD drivers to STREAMS protocol stack	NetProgGde (Sec. 8.2)
BSD driver	bridging to STREAMS protocol stack	NetProgGde (Sec. 8.2)
BSD socket		NetProgGde (Sec. 4.4)
BSD socket	and network addresses	NetProgGde (Sec. 4.4.1)
driver	bridging BSD driver to STREAMS protocol stack	NetProgGde (Sec. 8.2)
sockets	BSD	NetProgGde (Sec. 4.4)
STREAMS protocol stack	bridging to BSD driver	NetProgGde (Sec. 8.2)

The *Master Index* search utility works on the Documentation CD–ROM and on the Web.

- Printed versions of the Tru64 UNIX core documentation are available in the separately orderable Tru64 UNIX Documentation Kit and in individual subkits. See the *Documentation Overview* for a description of those kits.

  The books in the Startup Documentation Kit are included in the Tru64 UNIX media kit. See Section 1.4.1 for a list of those books. The media kit also includes the *Cluster Technical Overview*.

- On the Web, you can find the current documentation set, as well as sets for Tru64 UNIX versions dating back to Version 4.0D. To view this documentation, point your Web browser to the following site:

  http://www.tru64unix.compaq.com/docs

  The Web site also gives you access to the Tru64 UNIX reference pages, as well as to the following documentation:

  – Tru64 UNIX Best Practices documentation

  – Tru64 UNIX Device Driver documentation library

  – TruCluster Server documentation (including the TruCluster reference pages)

  – Advanced Server for UNIX documentation

– Internet Express for Tru64 UNIX, formerly Open Source Internet Solutions (see Section 1.3.7.1)

The Version 5.1A release offers a new full-text search facility, called *isearch*, for finding information in Tru64 UNIX documentation on the Web and on local area networks, using any platform, including UNIX workstations, Macintosh computers, and PCs:

– When accessing Tru64 UNIX documentation on the Web, you simply type a word or words in the search facility to find the places in the documentation that discuss the topic.

– To use the search facility on an intranet, your system administrator must install the isearch software on your local server. Thereafter, users with access to that server can search the documentation in the same way as they would on the Web. The required software is provided free on the Tru64 UNIX Documentation CD-ROM.

- Best Practices documentation offers a recommended method for performing a task, rather than presenting all available options. The documentation walks you step-by-step through the process.
- The online help for SysMan and its associated applications is robust and task oriented to help you find the information you need to complete a given goal.

1.3 Optional Components

You can purchase a wide variety of software products from Compaq and many other companies to run on the Tru64 UNIX operating system.

Most of the separately licensed products described in the following sections are included with the Tru64 UNIX media kit. See the *Tru64 UNIX Operating System Quickspec* (formerly called *Software Product Description*) or the Tru64 UNIX Web site for additional information on these optional software packages.

1.3.1 TruCluster Server

TruCluster Server is a highly integrated synthesis of Tru64 UNIX software, AlphaServer systems, and storage devices that operate as a single system. A TruCluster Server cluster acts as a single virtual system, even though it is made up of multiple systems. Members of the cluster can share resources, data storage, and clusterwide file systems under a single security and management domain, yet they can be booted or shut down independently without disrupting the cluster.

A TruCluster Server environment can be as simple or as feature rich as you want. You can configure a cluster that fits your needs, from a two-node

cluster up to an eight-node cluster running high availability applications, such as the following:

- Transaction processing systems
- Servers for network client/server applications
- Data-sharing applications that require maximum uptime
- Distributed parallel processing applications that take full advantage of the TruCluster Server application programming interfaces (APIs)

TruCluster Server includes a cluster alias for the Internet protocol suite (TCP/IP), so that a cluster appears as a single system to its network clients and peers.

If you know how to manage a Tru64 UNIX system, you already know how to manage a TruCluster Server cluster because TruCluster Server extends single-system management capabilities to clusters. The SysMan Menu utilities provide an integrated view of the cluster environment, letting you manage a single member or the entire cluster.

For more information, see the TruCluster Server *Cluster Technical Overview*.

1.3.2 Logical Storage Manager

The Logical Storage Manager (LSM) software is an optional integrated, host-based disk storage management application that allows you to manage storage devices without disrupting users or applications accessing data on those storage devices. Among the features provided by LSM is concatenation, striping, mirroring, and RAID 5. You can perform LSM tasks using several graphical interface utilities, a menu-driven utility, or the LSM command line.

With the Version 5.1A release, the LSM graphical interface `lsmsa` has been enhanced to provide full AdvFS functionality and an extensive new help suite. Also called the LSM Storage Administrator, the `lsmsa` interface displays a hierarchical view of LSM objects, AdvFS domains, and their relationships. You use `lsmsa` to view and manage LSM objects and AdvFS domains on a local or remote (client) system.

The Version 5.1A release also removes some of the restrictions on LSM in a cluster. You can now place the cluster_root domain into an LSM volume with the new `volmigrate` command, and encapsulate cluster members' swap devices into LSM volumes with the new `swap` keyword to the `volencap` command. For more information see the *Cluster Administration* manual and the `volmigrate`(8) and `volencap`(8) reference pages.

For more information about LSM, see Section 2.3 or the *Logical Storage Manager* manual.

1.3.3 Advanced File System Utilities

The Advanced File System Utilities extend the high availability and flexibility of AdvFS. They provide a graphical user interface to help you do the following:

- Spend less time managing file systems
- Keep your data on line during routine maintenance
- Extend the capacity of your files
- Balance the percentage of space used on volumes
- Undelete files, stripe files, and clone files for hot backup

The graphical interface integrated with Performance Manager lets file system metrics pass from Advanced File System Utilities to Performance Manager. (See Section 2.5.3 for information about Performance Manager.)

1.3.4 Developer's Toolkit

The Developers' Toolkit is a prerequisite for all Compaq Tru64 UNIX development tools, languages, and environments. The Toolkit contains the following components:

- Compaq C for Tru64 UNIX, an ANSI conformant C compiler
- Debuggers (Ladebug, dbx)
- The Atom API
- Program analysis tools (profiling and performance analysis)
- Visual Threads (thread-related profiling and debugging)
- Porting Assistant

See Chapter 6 for more information.

1.3.5 Advanced Server for UNIX

The Advanced Server for UNIX software provides seamless interoperability between Tru64 UNIX servers, Windows NT and Windows 2000 servers, and Microsoft Windows clients. The Advanced Server enables a Tru64 UNIX system to run the services that make it appear as a Microsoft Advanced Server. See Section 7.2 for more information.

1.3.6 Multimedia Services

Multimedia Services software brings audio and video capabilities to Compaq Tru64 UNIX workstations and provides a full multimedia programming

library for developers. The Multimedia Services Run-Time license is included with the base operating system.

1.3.7 Other Software

The following sections describe software that is not included with the Tru64 UNIX media kit, but can be separately obtained:

1.3.7.1 Internet Express for Tru64 UNIX

The software on the Internet Express for Tru64 UNIX CD–ROM provides commonly used Internet software, which has been compiled and configured to run on Tru64 UNIX AlphaServer systems. The Internet Express (formerly Open Source Internet Solutions) CD–ROM is packaged with AlphaServer systems and is separately orderable from Compaq.

The software package also provides administrative tools developed by Compaq to configure and manage Internet components. The following list describes some of the software included in the package:

- A collection of Open Source Internet software (binaries and sources), tested and qualified on Tru64 UNIX servers (Web, mail, news, directory, proxy, and streaming media), a database management system, security services, and tools for providing dynamic Web content.
- Automatic installation and configuration of Open Source Internet software.
- A collection of commercial Internet products, either full-function or evaluation copies.
- Compaq Secure Web Server (which is also included with the Tru64 UNIX kit. See Section 1.2.8.)
- Compaq Internet Monitor (Web-based quality-of-service tools).
- A Web-based administration utility to manage Internet services.

All software has been tested and certified to operate on a TruCluster Server. Visit the following Web site for more information about Internet Express for Tru64 UNIX:

http://tru64unix.compaq.com/internet/osis.htm

See also Section 1.2.10 for information about the Tru64 UNIX Open Source Software Collection CD–ROM.

1.3.7.2 Enterprise Toolkit for Visual Studio

The Enterprise Toolkit is a set of extensions or add-ins to Microsoft Visual Studio that support developing C, C++, and Fortran applications for Tru64

UNIX servers. With Compaq Enterprise Toolkit for Visual Studio, developers can use the popular Microsoft Visual Studio tool to develop, edit, compile, build, and debug applications for Tru64 UNIX or Windows from a single desktop.

You can create and manage basic UNIX applications or create more powerful and complex client/server and distributed applications, harnessing the power of 64-bit Tru64 UNIX Alpha technology with a single set of PC tools.

Additionally, the Compaq Enterprise Toolkit provides developers with a rich set of performance and memory analysis tools. The Enterprise Toolkit uses Visual Studio's documentation browser, the HTML Help Viewer, to provide access to UNIX and product documentation from the same window that developers view Windows documentation.

1.3.7.3 Source Materials Options

A source kit is available for users who need to retrieve and modify selected source modules, primarily for making highly specialized modifications.

1.4 Packaging

The Tru64 UNIX media kit contains the following CD-ROMs:

- Operating System
- Associated Products Volume 1
- Associated Products Volume 2
- Software Documentation
- Alpha Systems Firmware Update
- Open Source Software Collection (see Section 1.2.10)

Many of the associated products are optional subsets of the base operating system. Others (such as the Developer's Toolkit, TruCluster Server products, Logical Storage Manager, and Advanced Server for UNIX) are separately licensed products. See Section 1.3.7 for more information.

The media kit also contains printed to help you install, set up, and become familiar with your Tru64 UNIX operating system. See Section 1.4.1 for a list of the included documents.

If you purchase an Update Contract for the media, you receive the current versions of the operating system, associated products, and documentation CD–ROMs, in addition to any of the printed manuals in the Startup Documentation Kit that have changed since the last release.

1.4.1 Documentation

The following printed documentation ships with the Tru64 UNIX media kit:

- *Release Notes for Version 5.1A*
- *Installation Guide*
- *Installation Guide — Advanced Topics*
- *Update Installation Quick Reference Card*
- *Full Installation Quick Start*
- *Technical Overview*
- TruCluster Server *Cluster Technical Overview*
- *Quick Reference Card*
- *Documentation Overview*

1.4.2 Licensing

Tru64 UNIX operating system software is furnished under the licensing of the Compaq Computer Corporation Standard Terms and Conditions.

In addition to the Operating System Base License, which is the prerequisite for all other licenses, there are four other types of operating system licenses available:

- Symmetric Multiprocessing (SMP) Extension to Base license
- Concurrent Use licenses
- Unlimited Interactive User licenses
- Hardware Partitioning license

  Tru64 UNIX provides the technology to support static hardware partitions only on certain servers. These hardware partitions allow multiple instances of the operating system, which increases flexibility in testing new versions and running multiple versions for applications. The use of Tru64 UNIX in hardware partitions requires a Tru64 UNIX Hardware Partitioning License for each additional partition. (See the *System Administration* manual for more information about hardware partitioning.)

For more information on these licenses, see the *Tru64 UNIX Operating System QuickSpec* (formerly called *Software Product Description*).

2

System Management

This chapter introduces some of the system management features of the Tru64 UNIX operating system. The following topics are discussed:

- Installing the operating system and its optional subsets (Section 2.1)
- Configuring your system (Section 2.2)
- Managing your storage devices with the Logical Storage Manager (Section 2.3)
- Managing your system using the SysMan Menu suite of graphical applications (Section 2.4)
- Monitoring and managing systems and events (Section 2.5)
- Managing your hardware with the `hwmgr` utility and the Dynamic Device Recognition framework (Section 2.7)
- Loading subsystems dynamically (Section 2.8)
- Changing kernel attributes dynamically (Section 2.9)
- Using dataless management services (Section 2.10)

2.1 Installation

Tru64 UNIX supports Full and Update Installations either from a CD-ROM or across the network from a Remote Installation Services (RIS) server. It also supports system cloning.

2.1.1 Full Installation

A Full Installation lets you install Tru64 UNIX on new and existing systems. You can use the recommended settings for the file system layout, kernel components, and software or you can make your own customized selections.

You can use either a graphical interface or a text-based interface to install the operating system quickly and easily. The graphical interface (available only on graphics-capable systems) steps you through each phase of the setup process and lets you go backward and forward at any time during the installation. The text-based interface also guides you through each setup phase and you can go back and change your answers, if necessary.

A Full Installation creates new file systems and swap space and overwrites existing system and user-created files on the disk partitions where the file systems and swap spaces are to be installed. You have the option to use default values for the disk layout and swap space allocation or to completely customize the locations of file systems and swap space.

Major features of the Full Installation process include the following:

- Both the text-based and graphical user interfaces have a task-oriented design, which steps you through each installation task and lets you go backward and forward at any time to change your answers.
- By default, the Full Installation process determines the file system layout based on your software selections. You do not need to calculate the size of the file systems, nor do you need to repartition your disks in advance of the installation process to ensure a successful installation.
- You can install and configure the Logical Storage Manager (LSM) and the Worldwide Language Support (WLS) software during a Full Installation, instead of as a post-installation task.

You can customize and extend the Full Installation by creating custom scripts or programs to run at three process points during installation: before the installation begins, after software subsets load, and after the system reboots. The files you create can be loaded on a diskette, a CD–ROM, or a RIS server for use by the installation process.

A Full Installation creates a Configuration Description File (CDF) called `install.cdf` that can be used to replicate the Full Installation on similar systems (see Section 2.1.3). You can also capture configuration data into a `config.cdf` file from a running system and replicate the following during a Full Installation:

- Network card and router configuration
- Domain Name System (DNS)
- Network Information Service (NIS)
- Network Time Protocol (NTP)
- Printer services
- Mail services

For more information on Full Installations, see the *Installation Guide* and the *Full Installation Quick Start* card.

2.1.2 Update Installation

An Update Installation updates Tru64 UNIX from an earlier version of the operating system. You can use the Version 5.1A Update Installation process

to update your system from Versions 5.0A or 5.1 to Version 5.1A. An Update Installation preserves the following:

- Disk partitions
- File systems
- File customizations
- The network, print, and mail environments
- User accounts and user-created files
- Other system customizations you may have done

Do not perform an Update Installation if you want to change the type, location, or size of file systems or if you want to install optional software.

An Update Installation updates the following software subsets:

- Any currently installed software subsets that comprise the operating system (known as base software subsets). Base software subset names start with the prefix `OSF`.
- Any mandatory base software subsets that are introduced in the new version.
- Any currently installed Worldwide Language Support (WLS) software subsets. WLS software subset names start with the prefix `IOS`.
- Any currently installed TruCluster software subsets. TruCluster subsets start with the prefix `TCR`.

When invoked with the `-u` option, the Update Installation process runs in unattended mode, which means that barring any problems with the update, no user interaction required. The only exception to this is the switching of CD–ROMs if WLS software is being updated. The `-u` option builds a kernel with all kernel components and does not provide the chance to archive obsolete files.

An analysis phase at the beginning of the Update Installation process does the following:

- Detects the presence of layered products that prevent the Update Installation from continuing
- Determines if any layered products that should be reinstalled after the update
- Finds fatal and nonfatal file system type conflicts
- Determines available disk space

If layered product or nonfatal file type conflicts are discovered, you can resolve them directly from the Update Installation user interface; there is no need to exit the installation, resolve the conflict, and restart the installation.

If your system does not have enough available disk space for new software and room for Update Installation processing, disk space recovery options are available directly from the Update Installation as well.

You cannot install additional optional software subsets during an Update Installation nor update layered products. You can, however, install additional optional software subsets by using the `setld` utility when the Update Installation is complete (see Section 2.1.6). To update layered products, it may be necessary to delete the existing version of the product and install the new version that is designed to operate with the new operating system version. The Update Installation notifies you accordingly.

The Update Installation features are classified into two types:

- Features that you control when you begin an Update Installation (shown in Table 2–1).
- Features that are built into the Update Installation process (shown in Table 2–2).

Table 2–1: User-Controlled Features of the Update Process

User Options	Description
Unattended Update Installation	If you do not need to select optional kernel components or archive obsolete files, you can invoke the Update Installation with the `-u` flag to run the update without any user intervention.
Kernel Component Options	You have the option to build either mandatory only or all kernel components into the kernel, or you have the option to interactively select optional kernel components.
Archive Obsolete Files	You have the option to archive obsolete files before they are automatically removed by the Update Installation.

Table 2–2: Built-In Features of the Update Process

Built-In Feature	Description
Notification of conflicting layered products	Notifies you when an installed layered product may not be compatible with the new version of the operating system; this layered product may need to be reinstalled later.
Remove layered products that prevent the update from continuing	Upon your confirmation, removes layered products that prevent the update from continuing.
Update base operating system, WLS, and TruCluster software to new version	Updates existing installed subsets and installs new mandatory subsets introduced in the new version.

Table 2–2: Built-In Features of the Update Process (cont.)

Built-In Feature	Description
Check for changed file types	Checks for file types that have been changed. The update might not be able to proceed if certain conflicts are found.
Disk space recovery	Provides the option to remove unnecessary software subsets and `.PreUPD`, `core`, and extra kernel files to recover disk space if there is not enough file space to complete the update.
Execute instructions provided in user-supplied files	You can customize an Update Installation by creating and moving user-supplied scripts, programs, or executable files to the appropriate location. If the update process finds files with the correct names in the appropriate locations, the files are executed.

For more information on the Update Installation, see the *Installation Guide* and the *Update Installation Quick Reference Card*.

2.1.3 Installation Cloning

Installation Cloning lets you duplicate the installation characteristics (that is, the file systems and installed software) from a running system onto one or more systems with the same or similar hardware configuration.

The use of Installation Cloning to mass-install systems has the following benefits:

- You can produce identical installations with less effort.
- You can set up the Installation Cloning process to run with minimal user intervention.
- You can save time and reduce the chance of error in environments because Installation Cloning eliminates the need to manually perform duplicate installations on all systems.
- You can centrally administer software instead of attempting concurrent installations with locally mounted removable media such as CD–ROMs.

When you install the current version of the operating system on a machine, the installation process automatically generates a configuration description file (CDF) that contains a record of the installation setup data you specified, and therefore contains all of the installation information required to perform the same installation on a target system.

Installation cloning is not supported between different releases of the operating system because CDFs created by other versions of the operating system are not compatible with the current version. Therefore, if you want

to clone Version 5.1A onto a target system, you must create the CDF by performing a Version 5.1A Full Installation.

Systems that are installed by the cloning process must have the same disk configuration as the system where the CDF was generated. This means that the disks used for the / (root), /usr, /var, /usr/i18n file systems and swap areas on both systems must have the same disk type and the same device name. It is possible, however, to accommodate slight differences in configuration.

For information about Installation Cloning, see the *Installation Guide — Advanced Topics*.

2.1.4 Configuration Cloning

Configuration Cloning lets you duplicate the network, printer, and mail services and other configuration items from an already configured system onto one or more systems. Configuration Cloning is practical when more than one system has to be configured in a similar fashion.

To achieve a fully automated installation and configuration of another system, Configuration Cloning can be combined with Installation Cloning to completely eliminate the need to manually perform configuration tasks after the system is installed.

Configuration Cloning is not supported between different releases of the operating system.

For more information see the *Installation Guide* and the *Installation Guide — Advanced Topics*.

2.1.5 Rolling Upgrade

A rolling upgrade is a software upgrade of a cluster that is performed while the cluster is in operation. One member at a time is rolled and returned to operation while the cluster transparently maintains a mixed-version environment for the base operating system, cluster, and Worldwide Language Support (WSL) software. Clients accessing services are not aware that a rolling upgrade is in progress.

Rolling in a patch uses the same procedure as rolling in a new release of the base operating system and cluster software. The only difference is whether you run `installupdate` or `dupatch` during the install stage.

TruCluster Server Version 5.0A was the first TruCluster Server Version 5 release to contain the infrastructure that makes a rolling upgrade possible.

For more information, see the TruCluster Server *Installation Guide*.

2.1.6 The setld Utility

The `setld` utility allows system administrators to install software subsets, list installed subsets, and delete subsets that are formatted according to the guidelines set forth in *Guide to Preparing Product Kits*. For example, a system administrator might use the `setld` utility to install optional subsets that were not installed during a Full or Update Installation of the operating system.

Application programmers should use the Compaq kitting process when packaging software subsets designed to be installed on Tru64 UNIX systems.

For more information on the `setld` utility, see the *Installation Guide* and the `setld`(8) reference page.

2.1.7 Installation Documentation

The following documentation can help you with your installations:

- *Installation Guide*
- *Installation Guide — Advanced Topics*
- *Update Installation Quick Reference Card*
- *Full Installation Quick Start* card
- TruCluster Server *Cluster Installation*
- *Sharing Software on a Local Area Network*

  This manual describes how to set up a Remote Installation Services (RIS) server and create RIS environments to serve.

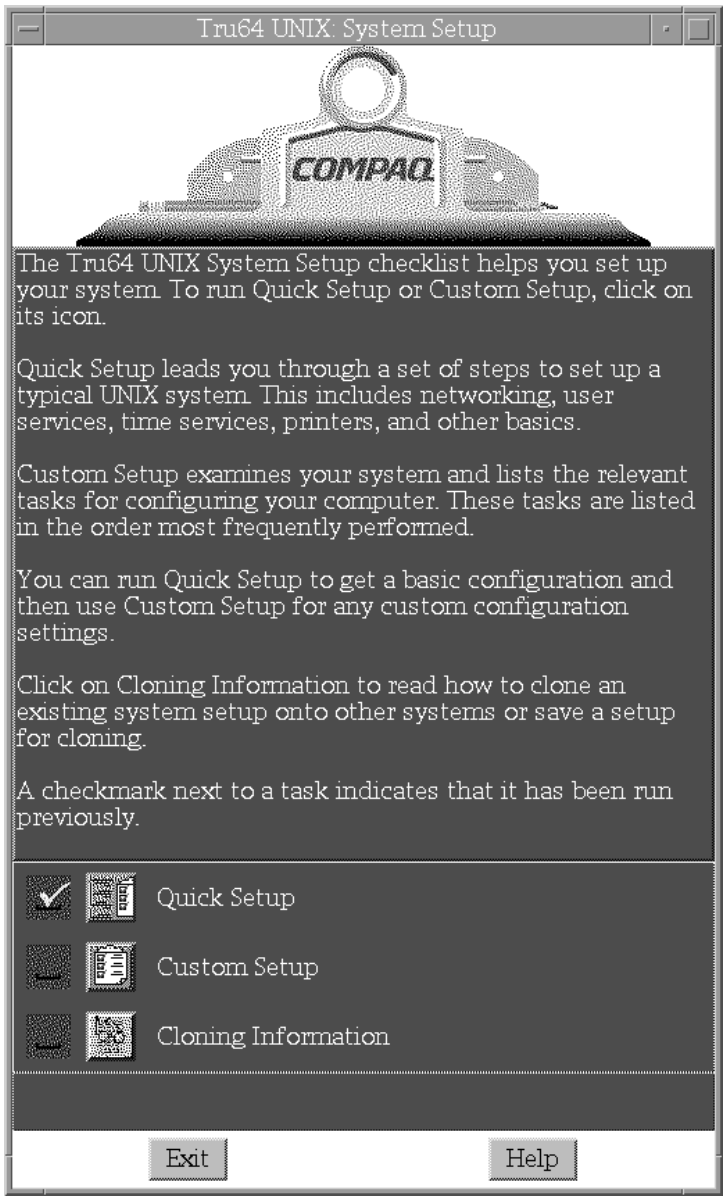
For information on this documentation, see the *Documentation Overview*.

2.2 System Configuration

If your system has graphics capabilities, you can use SysMan System Setup (Figure 2–1) to set up your system after the Tru64 UNIX software is installed. System Setup enables you to invoke the Quick Setup and Custom Setup applications.

The first time you log in as superuser or `root` after a system installation or the first time you log in to a factory-installed software (FIS) system, you will have the option to use either Quick Setup to configure a limited set of system parameters (including network and printer parameters) or use Custom Setup to set up your system for general use.

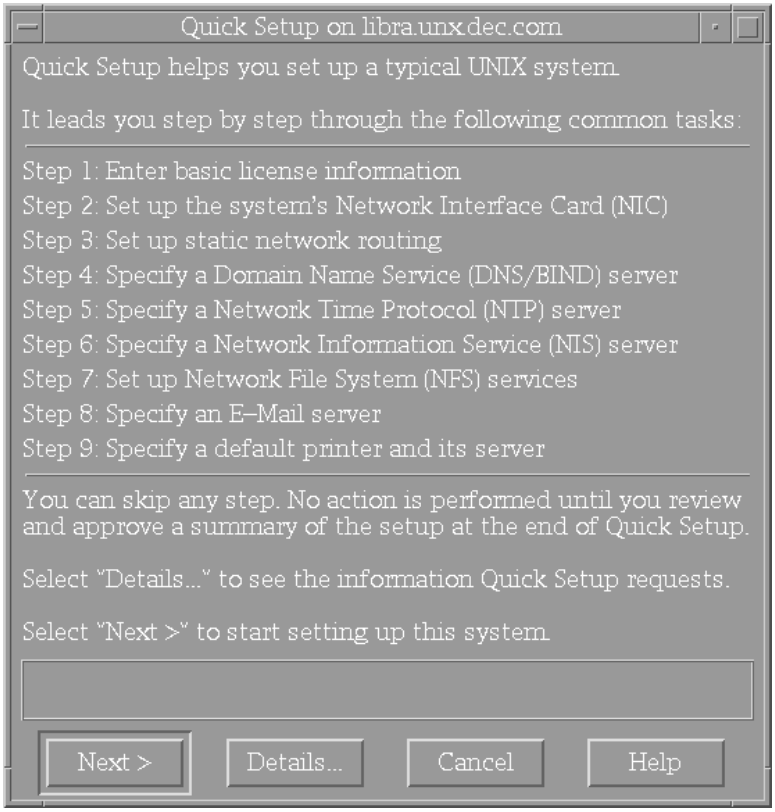
Figure 2–1: System Setup



Quick Setup (Figure 2–2) has a wizard-like design that lets you enter a minimal amount of key information. Quick Setup updates your system with the basic configuration needed to get a client system up and running, including network connection, mail, and print capabilities. Quick Setup should satisfy the configuration needs for most systems. Even for systems that will be configured as servers, it is recommended that you use Quick

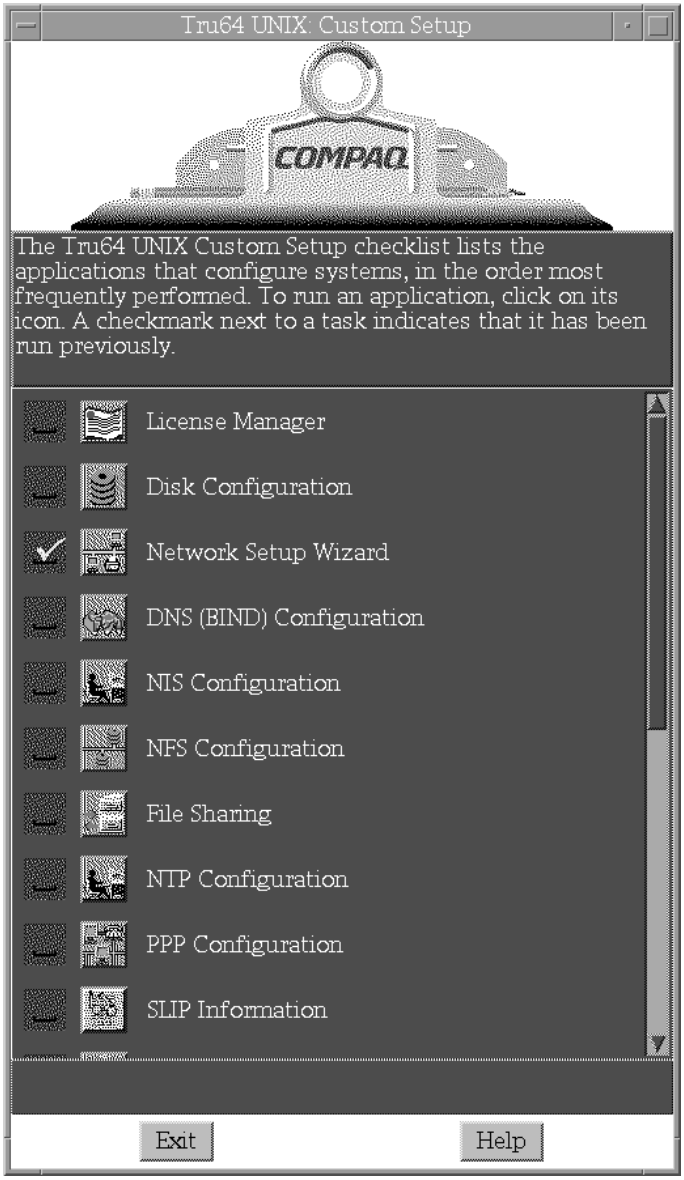
Setup first to configure your system then customize it with advanced applications in Custom Setup.

Figure 2–2: Quick Setup



Custom Setup (Figure 2–3) lets you perform sophisticated system configuration that is beyond Quick Setup's scope. When Custom Setup runs, it examines your system and presents a list of configuration applications that are relevant for your system; these applications are displayed as a checklist. Once you have accessed an application, a checkmark appears next to the application. For information about individual applications, see the Welcome to SysMan online help.

Figure 2–3: Custom Setup



When you are logged in as superuser or `root`, you can invoke the SysMan System Setup at any time by clicking on the Configuration Checklist icon in the System_Administration folder, or by entering the following command on the command line:

```
# /usr/sbin/checklist
```

The utilities you see listed with the Custom System checklist depend upon which subsets are installed on your system. For example, if the optional security subset were not installed, the Audit Configuration application would not be displayed. The following list represents some of the applications you might see with the Custom System checklist:

License Manager	Disk Configuration	Network Setup Wizard
DNS (BIND) Configuration	NIS Configuration	NFS Configuration
Account Manager	Mail Configuration	LAT Configuration
NTP Configuration	Printer Configuration	Security Configuration
Audit Configuration	DOP Configuration	Insight Manager
Update Administration	GUI Selection	

Many of the SysMan System Setup applications are also available in ASCII format for use on character-cell displays.

For more information about system setup in general, see the *Installation Guide*, the *System Administration* manual, the *Network Administration: Connections* and *Network Administration: Services* manuals, the *Software License Management* manual, and the `setup(8)` reference page.

2.3 Logical Storage Manager

The Logical Storage Manager (LSM) software is an optional integrated, host-based disk storage management application. LSM uses RAID technology to enable you to configure storage devices to protect against data loss, maximize disk use, improve performance, provide high data availability, and manage storage without disrupting users or applications accessing data on those disks.

With LSM you manage all of your storage devices (disks, partitions, RAID sets, and such) as a flexible pool of storage from which you create LSM volumes. You configure new file systems, databases, and applications or encapsulate existing ones to use an LSM volume instead of a disk partition. The benefits of using an LSM volume instead of a disk partition include the following:

- Data loss protection

  LSM can automatically store and maintain multiple copies (mirrors) of data or data and parity information. If a storage device fails, LSM protects your data in the following ways:

  – It continues operating using either the mirrors or the remaining data and parity information, without disrupting users or applications, shutting down the system, or backing up and restoring data

  - It can automatically transfer the data from the failed storage device to a designated spare disk, or to free disk space, and send you mail about the relocation

  You can also use LSM to encapsulate the boot disk partitions into LSM volumes, then create mirrors of those volumes. By doing so, you create copies of the boot disk partitions from which the system can boot if the original boot disk fails.

- Maximize disk usage

  You can configure LSM to seamlessly join together storage devices to appear as a single storage device to users and applications.

- Performance improvements

  You can configure LSM to separate data into units of equal size, then write the data units to two or more storage devices. LSM simultaneously writes the data units if the storage devices are on different SCSI buses.

- Data availability

  You can configure LSM in a TruCluster environment. TruCluster software makes AlphaServer systems appear as a single system on the network. The AlphaServer systems running the TruCluster software become members of the cluster and share resources and data storage. This sharing allows applications, such as LSM, to continue uninterrupted if the cluster member on which it was running fails.

LSM is an optional subset located on the Tru64 UNIX CD–ROM. You can install LSM when you install the Tru64 UNIX operating system software or at a later time.

Without an LSM license, you can use LSM to join together disks and partitions, then create an LSM volume to use that storage. All other LSM features require an LSM license. Contact your local Compaq office or your Compaq authorized reseller for information about Compaq's licensing terms and policies or purchasing an LSM license.

The LSM graphical interface, `lsmsa`, uses the Java run-time environment to provide a method of invoking LSM commands and to monitor LSM file status. When the main window is displayed, a hierarchical view of LSM objects is presented. Clicking on an object displays the objects of that type and a table of information about them.

2.4 System Management Utilities

The SysMan application suite makes your job as a system or network administrator easier by providing you with an application for each of your administration tasks, such as installation, configuration, daily administration, monitoring, kernel and process tuning, and storage

management. You can access these applications through the SysMan pop-up menu from the CDE front panel when you log in as `root`. Most of the applications also have supported command-line counterparts.

Although the SysMan utilities were designed to take advantage of the Common Desktop Environment (CDE), most will work outside of CDE with other window or display managers. For example, the following command invokes the Network Interface Configuration tool:

```
# sysman interface
```

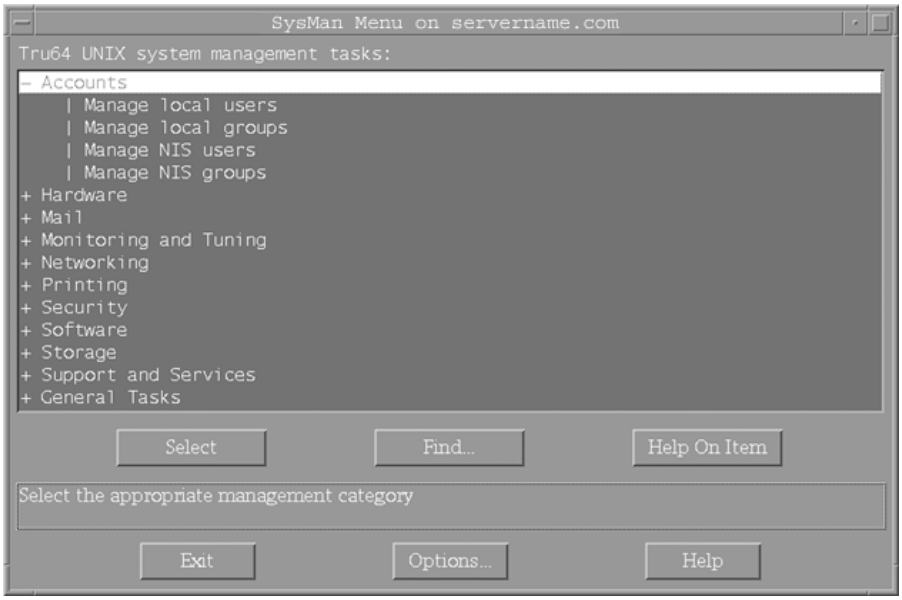
Many of the SysMan utilities are also available as text-based interfaces for use on character-cell displays.

2.4.1 The SysMan Menu

The SysMan Menu (Figure 2–4) provides a menu of system management tasks in a tree-like hierarchy with branches of general categories, and leaves for actual tasks. The categories are Accounts, Mail, Monitoring and Tuning, Networking, Printing, Security, Hardware, Software, Storage, Support and Services, and General Tasks. You can expand or contract a branch to show the subbranches and leaves within a main branch. Selecting a leaf invokes a task that opens a dialog box for performing the task.

The SysMan Menu is invoked from the CDE front panel when you are logged in as `root`, or directly from the command line by entering the `/usr/sbin/sysman` command. Because the SysMan Menu can be run in CDE, HTML, or ASCII text environments, you can use the SysMan Menu on an X11 capable display, on a personal computer running Microsoft Windows products, Linux, or the Macintosh Operating System, or on a character cell terminal.

Figure 2–4: The SysMan Menu



The SysMan Menu offers the following typical applications, depending on what options are installed and configured on the local system:

```
- Accounts [accounts]
    | Manage local users [users]
    | Manage local groups [groups]
    | Manage NIS users [nis_users]
    | Manage NIS groups [nis_groups]
- Hardware [hardware]
    | View hardware hierarchy [hw_hierarchy]
    | View cluster [hw_cluhierarchy]
    | View device information [hw_devices]
    | View central processing unit (CPU) information [hw_cpus]
    | Manage CPUs [hw_manage_cpus]
    | Online Addition/Replacement (OLAR) policy information [hw_olar_policy_info]
- Mail [mail]
    | Configure Mail [mailsetup]
- Monitoring and Tuning [monitoring]
    | View events [event_viewer]
    | Set Up Insight Manager [imconfig]
    - Class Scheduling [class_sched]
        | Configure Class Scheduler [class_setup]
        | [Re]Start Class Scheduler [class_start]
        | Stop Class Scheduler [class_stop]
    | View Virtual Memory (VM) statistics [vmstat]
    | View Input/Output (I/O) statistics [iostat]
    | View Uptime statistics [uptime]
- Networking [network]
    | Network Setup Wizard [net_wizard]
    - Basic Network Services [networkbasic]
        | Set up Network Interface Cards. [interface]
        | Set up static routes (/etc/routes) [route]
        | Set up routing services (gated, routed, IP Router) [routing]
        | Set up hosts file (/etc/hosts) [host]
        | Set up hosts equivalency file (/etc/hosts.equiv) [hosteq]
```

```
        | Set up remote who services (rwhod) [rwhod]
        | Set up the networks file (/etc/networks) [networks]
    - Additional Network Services [networkadditional]
        - Domain Name Service (DNS(BIND)) [dns]
            | Configure system as a DNS client [dns_client]
            | Deconfigure DNS on this system [dns_deconfigure]
        - Serial Line Networking [serial_line]
            - Point-to-Point Protocol (PPP) [ppp]
                | Create option files [ppp_options]
                | Modify pap-secrets file [pap]
                | Modify chap-secrets file [chap]
        - Network Time Protocol (NTP) [ntp]
            | Configure system as an NTP client [ntp_config]
            | View status of NTP daemon [ntp_status]
            | {Re}start NTP daemon [ntp_start]
            | Stop NTP daemon [ntp_stop]
        - Network File System (NFS) [nfs]
            | View NFS configuration status [nfs_config_status]
            | Configure system as an NFS client [nfs_client]
            | Deconfigure system as an NFS client [nfs_deconfig_client]
            | Configure system as an NFS server [nfs_server]
            | Deconfigure system as an NFS server [nfs_deconfig_server]
            | View NFS daemon status [nfs_daemon_status]
            | Start/Restart NFS daemons [nfs_start]
            | Stop NFS daemons [nfs_stop]
        | Configure Network Information Service (NIS) [nis]
    | View network daemon status [dmnstatus]
    | Start or Restart network services [inet_start]
    | Stop network services [inet_stop]
- Printing [printers]
    | Configure line printers [lprsetup]
- Security [security]
    | Configure Division of Privileges (DOP) [dopconfig]
    | Manage DOP Actions [dopaction]
    | Security Configuration [secconfig]
    | Audit Configuration [auditconfig]
- Software [software]
    - Installation [install]
        | Install software [setldload]
        | List installed software [setldlist]
        | Remove installed software [setldd]
        | Cleanup after an OS update (updadmin) [updadmin]
    | Register license data [lmfsetup]
- Storage [storage]
    - File Systems Management Utilities [filesystems]
        - General File System Utilities [generalfs]
            | Dismount a File System [dismount]
            | Display Currently Mounted File Systems [df]
            | Mount File Systems [mount]
            | Share Local Directory (/etc/exports) [export]
            | Mount Network Directory (/etc/fstab) [net_mount]
        - Advanced File System (AdvFS) Utilities [advfs]
            | Manage an AdvFS Domain [domain_manager]
            | Manage an AdvFS File [file_manager]
            | Defragment an AdvFS Domain [defrag]
            | Create a New AdvFS Domain [mkfdmn]
            | Create a New AdvFS Fileset [mkfset]
            | Recover Files from an AdvFS Domain [salvage]
            | Repair an AdvFS Domain [verify]
        - UNIX File System (UFS) Utilities [ufs]
            | Create a New UFS File System [newfs]
    | Create a Bootable Tape [boot_tape]
    | Identify SAN Appliances Wizard [idsanappl]
- Support and Services [support]
```

```
    | Create escalation report [escalation]
    | Create configuration report [config_report]
- General Tasks [general_tasks]
    | Shutdown the system [shutdown]
    | Quick Setup [quicksetup]
    | Configure Prestoserve software [presto]
    | Configure X Display Manager [xsetup]
    | Cloning setup information [cloneinfo]
    | Command line interface information [sysmancli]
```

A keyboard accelerator is provided for each menu item, such as `[ufs]`. The accelerator enables you to quickly launch a specific tool directly from the command line; for example:

sysman ufs

2.4.2 The SysMan Station

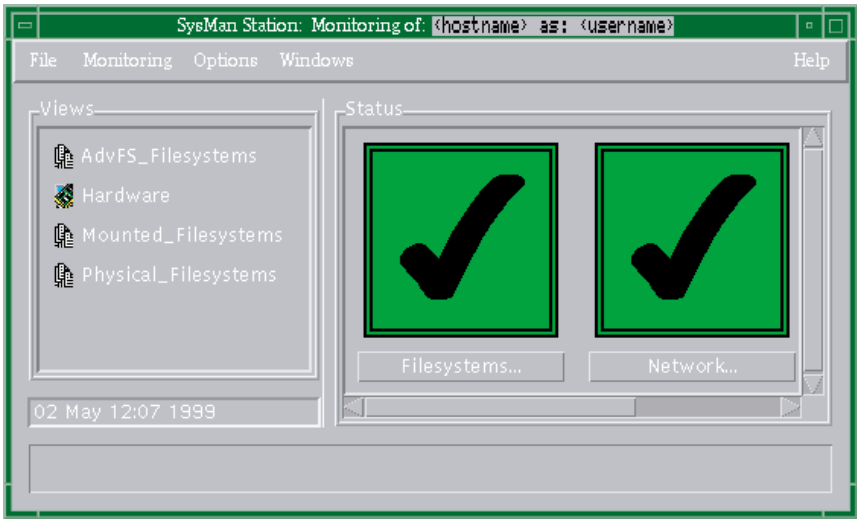
The SysMan Station (Figure 2–5) provides a high profile view and status of a system's physical and logical objects. It is intended to be the central point from which to manage a Tru64 UNIX system. The SysMan Station launches other SysMan utilities to perform the tasks. You can launch the SysMan or invoke applications directly from the Tools menu in the SysMan Station. It can run on a standard Java capable display (such as a UNIX workstation) or within a PC's browser, or it can be downloaded and run directly on a PC.

You can use SysMan Station to perform the following tasks:

- Monitor the status of a system or cluster at a glance
- Display detailed information about a system or cluster
- Provide a single location for management activity
- Display and track events that lead to a problem

You can launch the SysMan Station directly from an icon on the CDE front panel when you log in as `root`, or you can enter `/usr/sbin/sms` from the command line.

Figure 2–5: The SysMan Station



2.4.3 The CDE Application Manager

If you are running Tru64 UNIX with the CDE desktop, you have access to the CDE Application Manager. The System_Admin group on the CDE Application Manager launches the following SysMan utilities:

- SysMan Configuration Checklist
- SysMan Station
- SysMan Menu
- Welcome to SysMan, which provides an overview of the SysMan online help

The icons in the CDE Application Manager, System_Admin folders usually invoke a SysMan menu utility. For example, if you click on the DNS (BIND) icon in the Configuration folder the SysMan Menu is invoked, showing only the SysMan Menu DNS options. However, in a few instances, clicking on an icon will launch an X11-compliant graphical user interface (GUI).

These X11-compliant applications are distinctly different from the analogous tools available from the SysMan Menu. The GUIs take advantage of features that are only available in the X Windows System user environment, such as the ability to perform operations by dragging and dropping icon representations of objects. For example, when you click on the Account Manager icon in the CDE Daily_Admin folder, it does not invoke the SysMan Menu; instead, it displays the X11-complaint `dxaccounts` GUI.

This means that there might be two or more interfaces that you can use to perform an identical task, depending on your personal preference for a particular user environment.

The System_Admin group also contains the following categories of system administration groups:

- The Configuration group contains applications for configuring and setting up a Tru64 UNIX system after it has been installed. Once a system has been set up, it is unlikely that an administrator would have to use these applications on a regular basis. From Configuration group you can launch the following applications:
  - BIND/DNS
  - CDE Setup
  - DHCP Server
  - Disk Config
  - Mail Config
  - NFS Config
  - NIS Setup
  - PPP Config
  - Print Config
  - SLIP
  - latsetup
- The Daily_Admin group contains applications for performing typical administration tasks on a daily basis. From this group, you can launch the following applications:
  - Account Manager
  - Archiver
  - Audit Manager
  - Display Window
  - Event Viewer
  - File Sharing
  - Host Manager
  - License Manager
  - Mail User Administration
  - Power Management
  - Shutdown

  - System Information
- The Monitoring Tuning group contains applications for tuning and monitoring the system once it is up and running. From this group, you can launch the following applications:
  - Kernel Tuner
  - Process Tuner
  - System Information
  - Insight Manager Agents
- Software Management contains applications for managing and installing additional software on the system. From this group, you can launch the following applications:
  - Update Installation Cleanup
  - Update Installation
  - Install Software
  - Delete Software
  - List Software
- Storage Management contains applications for managing and monitoring file systems. From this group you can launch the following applications:
  - Advanced File System
  - Bootable Tape
  - Logical Storage Manager
  - Prestoserve I/O Accelerator

2.5 Performance and Event Management

Tru64 UNIX provides various utilities that you can use to monitor and manage your system and its events. The following sections provide a brief overview of these utilities; for more information and instructions on using them, see the *System Administration* manual and the *Installation Guide*.

2.5.1 Compaq Insight Manager

Compaq Insight Manager is a Web-based utility that functions independently from operating environments. Compaq Insight Manager provides a consistent wrapper for SysMan and other Tru64 UNIX utilities, enabling you to manage supported systems from a Web browser — a method of administering a computing environment that is known as Web-Based Enterprise Services (WEBES).

On a Tru64 UNIX system, you can use Compaq Insight Manager to view device details, but you must invoke SysMan Menu or SysMan Station to perform administrative tasks. On a PC or server running Windows NT, you can both view device details and invoke administrative tasks.

You can activate these Web browsing features from a dedicated HTTP port, or from the Compaq Insight Manager CIM32 or CIMXE Management Consoles running on Compaq Windows NT servers.

The Compaq Insight Manager Device homepage provides the following service icons:

- Compaq Insight Management Agents
- Compaq Tru64 UNIX Configuration Report
- Compaq Tru64 UNIX SysMan

2.5.2 Compaq Analyze

Compaq Analyze provides automatic background analysis of a system by constantly viewing and reading the error log file. When an event triggers an analysis rule, the analysis engine collects the error information and sends the information and analysis results to an email account determined by the system administrator.

Once Compaq Analyze is installed and configured, it starts automatically as part of the system start up procedures and runs as a background process. System administrators can start and stop the process when desired. If Compaq Analyze is already running, no new process is started, although the single, running process can support multiple graphical user interfaces for multiple users.

Compaq Analyze works in conjunction with Compaq's Web-Based Enterprise Services (WEBES). It is the enabling technology for the indictment of failing CPUs in support of the Tru64 UNIX Version 5.1A OLAR functionality (see Section 1.2.1).

2.5.3 Performance Manager

Performance Manager provides a passive, continuous analysis of your system to detect and correct performance problems from a central location.

Performance Manager is an optional subset of the operating system. It is a full-featured, single-system version that does not require a license. Earlier versions of Performance Manager had an optional license that enabled distributed performance monitoring and management. This license has been retired, and the product is in maintenance mode. For more information, see the *Performance Manager* manual.

2.5.4 Monitoring Performance History Utility

The Monitoring Performance History (MPH) utility gathers timely and accurate information on the reliability and availability of the Tru64 UNIX operating system and its hardware environment. MPH is a suite of shell scripts that copy error log and crash dump information twice per week. The information is automatically copied to Compaq for analysis through email. After analysis, reports are generated and distributed to the users of this information, namely software and hardware engineering, manufacturing, and Compaq Services. This data is internally secure to Compaq and is used exclusively for monitoring purposes.

MPH runs as a background task, using very negligible CPU resources. It is invisible to the user, requires no training to use, and does not impact or degrade system performance. For more information, see the *System Administration* manual.

2.5.5 The sys_check Utility

The `sys_check` produces an extensive dump of system performance parameters that you can use to record system values and parameters, providing a useful baseline of system data. This may be useful before you make major changes or perform troubleshooting procedures. When you run `sys_check`, it produces an HTML-formatted document on standard output. For more information, see the *System Administration* manual.

2.5.6 X-Based Utilities

Several graphical utilities are provided for fast checking of one or more aspects of system performance. These are X-based utilities that will display under any X-compliant windowing interface. In the Common Desktop Environment (CDE), these utilities are organized under the Tool Drawer icon on the CDE front panel. This icon displays the Application_Manager folder, which contains monitoring utilities in the following subfolders:

- Desktop_Tools

  This folder contains simple interfaces such as System Load to monitor CPU usage and Disk Usage to obtain the current status of the file system space per disk.

- System_Admin

  This folder provides two subfolders that contain utilities useful for monitoring:

  – Monitoring Tuning

Contains graphical interfaces such as the process tuner, proctuner, and the kernel tuner `dxkerneltuner` that are useful for checking and changing system settings.

– Utilities

  Contains graphical interfaces to command-line utilities such as `iostat` and `netstat` to constantly monitor the output, setting your preferences for update and display.

As with any graphical application, you can place the icons on the System Administration Desktop for quick access to system information or keep the displays open constantly to monitor any aspect of system performance. Programs for the graphical interfaces are located in `/usr/bin/X11`.

For more information, see the *System Administration* manual.

2.5.7 Environmental Monitoring

On any system, thermal levels can increase because of poor ventilation, overheating conditions, or fan failure. Without detection, an unscheduled shutdown could occur, causing a loss of data or damage to the system itself. Environmental Monitoring can monitor the thermal state of AlphaServer systems so that users can be alerted and the system can be shut down in an orderly manner.

The monitoring framework consists of four components: a loadable kernel module and its associated APIs, the Server System MIB subagent daemon, the `envmond` daemon, and the `envconfig` utility. For more information, see the *System Administration* manual.

2.5.8 Event Manager

The Event Manager (EVM) provides a single point of focus for the multiple channels through which system components report event and status information. These channels include various log files, including those generated by the system logger, `syslog`, and the binary error logger, `binlog`. Each of these channels monitors some segment of the system, for example, when a disk fills, a processor begins reporting hardware errors, or whether certain routine tasks have been completed successfully.

The Event Manager combines these events into a single event stream, which the system administrator can monitor in real time or view as historical events retrieved from storage.

EVM's viewing facilities include a graphical event viewer, which is integrated with the SysMan application suite, and a full set of command line utilities, which allow administrators to filter, sort, and format events in a variety of ways.

The system administrator can also configure the Event Manager to perform automatic notification of selected conditions. Rather than replacing the familiar event channels, such as `syslog` and `binlog`, EVM encapsulates them, so these channels remain in place and continue to handle the same set of events. At the same time, EVM makes them more accessible.

EVM has the following key features:

- Centralized event information
- Facilities for users and applications to post and monitor events
- Support for encapsulation of custom event channels
- Integration with DECevent for translation of binary error log events
- A choice of summary-line or detailed view of events, including online explanations
- A full set of command line utilities for posting and handling events from shell scripts and the command line
- A configurable event logger that allows full control over which events are logged and that optimizes storage space used by identical events
- Automatic log-file management to perform daily archiving and purging tasks
- Configurable authorization for posting or accessing events

For more information, see the *System Administration* manual.

2.5.9 DECevent Translation and Reporting Utility

The DECevent Translation and Reporting Utility is an error log formatting utility that translates system event log files into formatted ASCII reports. It supports both a command line and a graphical user interface.

DECevent provides two main functions:

- Translation

  DECevent translates events into ASCII reports derived from system event entries (bit-to-text translations)

- Analysis and Notification

  DECevent constantly monitors system events to isolate failing device components through analysis. It can notify the proper individuals of a potential problem

You can have DECevent report information by event types, date, time, and event entry numbers. Reports can be selected from full disclosure to brief information messages.

For more information, see the *DECevent Translation and Reporting Utility* manual.

2.6 Managing Crash Dumps

When a system shuts down unexpectedly, it can save all or part of the data in memory and the kernel image. Such events are referred to as system crashes or panics. The stored data and status information is called a crash dump. After a crash dump, the system shuts down to the console prompt and must be rebooted after you identify and resolve the problem. Crash dump files and their associated log files are usually required by your technical support representative for problem analysis.

To administer crash dumps, you must understand how crash dump files are created. You must also reserve space for the crash dump and crash dump files. The amount of space you reserve depends on your system configuration and the type of crash dump you want the system to perform.

The extensive Tru64 UNIX crash dump features enable you to specify the following:

- The number of crash dumps that are preserved.
- The option to disable crash dumps.
- How the dump process uses the swap partitions (virtual memory).
- The location where the dump is written. It can be an exempt memory region, a local disk (using the swap space), or a remote host.
- Whether dumps are compressed or uncompressed. (Analysis tools such as debuggers do not require uncompressed dump files.)
- The size of the dump file, by defining its content or by specifying a partial dump.
- A continuable dump. You can copy a snapshot of memory to a dump file without halting the system, a useful method of estimating crash dump size during dump configuration planning.
- A forced crash dump on a system that is not responding.
- How and where the dump files are logged and archived.

See the *System Administration* manual for information on managing crash dumps

2.7 Hardware Management

In most cases, Tru64 UNIX performs hardware management automatically. However, the operating system provides tools that enable you to view device information, and to perform hardware management tasks when necessary.

2.7.1 The hwmgr utility

The `hwmgr` utility for hardware management, introduced in Version 5.0, helps you to manage hardware components, including disk and tape drives, processors, and buses.

The `hwmgr` utility offers a wide variety of options, including the following:

- `-view`

  Displays information. The variations on this option include `-view devices`, which provides the hardware identifier, device special file name, model, and location of all the devices on the system. Another variation, the `-view hierarchy` option, displays the current hardware component hierarchy.

- `-flash`

  Identifies a disk by flashing its LED. The disk can be identified by its SCSI bus number, SCSI target number, logical unit number, or its device special file name.

- `-show component`

  Displays hardware components, including those that were previously registered but may not be currently registered with hardware management. This option returns a series of one-character flags indicating the component is currently registered, has device nodes associated with it, has a clusterwide unique name, has saved attributes associated with it, or is inconsistent with the hardware component database.

For more information, see the `hwmgr(8)` reference page.

2.7.2 Dynamic Device Recognition

Dynamic Device Recognition (DDR) is a framework for describing the operating parameters and characteristics of SCSI devices to the SCSI CAM I/O subsystem. You can use DDR on SCSI devices that are unsupported by the `hwmgr` utility to add new devices and change existing ones. To do this, you use a utility called `/sbin/ddr_config` and a text database called `/etc/ddr.dbase` to make changes to the subsystem after installation. You do not need to reboot the system after modifying the `ddr.dbase` database.

DDR is preferred over the soon-to-be retired static methods of recognizing SCSI devices, because DDR will not disrupt user services and processes.

2.8 Dynamically Loadable Subsystems

Tru64 UNIX provides the ability to package, load, and manage kernel subsystems on Tru64 UNIX systems.

Instructions on how to write and package loadable device drivers so that they will install and execute on Tru64 UNIX systems are discussed in the Device Driver documentation. The *Writing Kernel Modules* manual explains how to write and package loadable kernel subsystems so that they will be installed and execute on Tru64 UNIX systems. The *Programmer's Guide* also discusses the framework that supports the dynamic configuration and tuning of kernel attributes.

2.9 Dynamic System Configuration

To simplify system tuning, Tru64 UNIX allows you to change certain kernel attributes without having to edit the system configuration file or the `param.c` file and without having to rebuild and reboot a target kernel for the changes to take effect. Through the use of attribute tables, each kernel subsystem — whether a Tru64 UNIX kernel subsystem or one developed by a third-party vendor — can define kernel attributes that can be changed at run time. You do this prior to boot time by using the `/sbin/sysconfig` command with the `-r` option (if the kernel attribute supports run-time reconfiguration). At boot time you can do this by adding or modifying entries in the kernel attribute database, `/etc/sysconfigtab`.

For more information, see the *System Administration* manual; and the *System Configuration and Tuning* manual.

You can also modify kernel attributes with the `dxkerneltuner` command; for further information, see the `dxkerneltuner`(8) reference page.

2.10 Dataless Management Services

Tru64 UNIX supports dataless management services (DMS), which allow the `/` (`root`), `/usr`, and `/var` partitions of a system to reside on a DMS server and to be NFS-mounted over the network by a DMS client. The `/` and `/var` partitions are unique to each DMS client, while the `/usr` partition is shared. The DMS client swaps and dumps locally. Additional file systems can be mounted using NFS.

DMS reduces disk needs and simplifies system administration, because administrators can administer and backup their DMS clients on the DMS server.

3

Networking

This chapter describes the components of Tru64 UNIX that enable a wide variety of networking capabilities. The first section provides a brief networking overview (Section 3.1), after which the following topics are discussed:

- The Internet Protocol suite (Section 3.2)
- Network support (Section 3.3)
- Application programming interfaces (Section 3.4)
- Network administration software (Section 3.5)
- Naming services (Section 3.6)
- Time services (Section 3.7)

3.1 Overview

The networking components of Tru64 UNIX come primarily from OSF/1 Version 1.0. Certain modules, such as System V Release 4.0 STREAMS which was unavailable in OSF/1 Version 1.0, were taken from the OSF/1 Version 1.2 code base. Other functions, such as IPv4 multicasting and the packet filter applications, were taken from the public domain, enhanced, and integrated into the operating system as a service to our customers.

The Network File System (NFS) code, as well as the Remote Procedure Calling (RPC) code, Network Information Service (NIS), and remote daemons and their corresponding commands came from Sun Microsystems' Open Network Computing (ONC) Version 4.2.

Functions that Compaq has licensed and enhanced, such as NIS, were ported to Tru64 UNIX from ULTRIX, Compaq's earlier version of the UNIX operating system. Although conforming to the OSF/1 Version 1.2 standards, these components were determined to be more robust than those in the corresponding code from the OSF.

Like all subsystems in the operating system, the networking subsystem is designed to provide a standardized programming interface to enable third-party vendors to develop and port their networking applications with a minimum of difficulty.

3.2 The Internet Protocol Suite

TCP/IP supports a suite of protocols, each of which provides a different service. These protocols allow networking communications to be independent of network hardware. The TCP/IP protocol suite (Figure 3–1) is organized into the following groups:

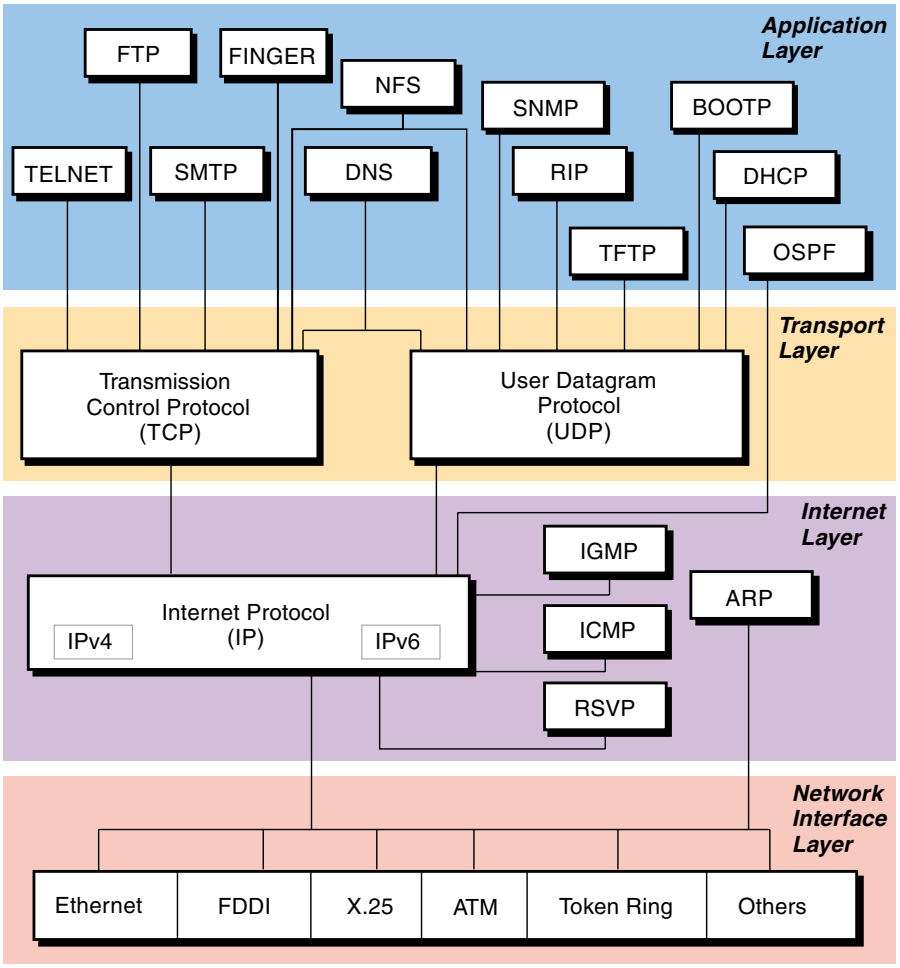
- Application-level protocols, such as DNS, routing protocols (DVMRP, EGP, BGP, RIP, RIPng, and OSPF), File Transfer Protocol (FTP), FINGER, TELNET, Trivial File Transfer Protocol (TFTP), Simple Mail Transfer Protocol (SMTP), and Simple Network Management Protocol (SNMP)
- Transport-level protocols, such as User Datagram Protocol (UDP) and Transmission Control Protocol (TCP)
- Network-level protocols, such as Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP), and Internet Protocol Version 6 (IPv6), Stateless Address Autoconfiguration, Neighbor Discovery, ICMPv6, multicast Listner Discovery (MLD)

Applications programs send messages (streams or blocks of data) to transport protocols — UDP and TCP. These protocols receive the data from the application, divide it into packets, add a transport header, and then pass the packets along to the next protocol layer, the Internet layer.

The Internet layer encloses the packet in an IP datagram, adds the datagram header, decides where to send the datagram (either directly to a destination or else to a gateway), and passes the datagram on to the network interface layer. The network interface layer accepts IP datagrams and transmits them as frames over specific network hardware.

Frames received by a network go through the protocol layers in reverse. Each layer strips off the corresponding header information until the data is back at the application level. Frames are received by the network interface layer (for example, an Ethernet adapter), which strips off the physical layer header and sends the datagram to the Internet layer. In the Internet layer, the Internet protocol strips off the IP header and sends the packet to the transport layer. The transport layer strips off the TCP or UDP header and sends the data to the application layer.

Figure 3–1: TCP/IP Protocols



3.2.1 Application-Layer Protocols

When an application needs to send data to an application on another host, the application sends the information through transport layer protocols to prepare the information for transmission. These protocols include DNS, EGP, BGP, RIP, OSPF, FTP, NFS, TELNET, TFTP, FINGER, SMTP, and SNMP.

3.2.1.1 Domain Name System Protocol

The Domain Name System (DNS) allows one or more hosts in a domain to act as a name server for other hosts within the domain. DNS uses UDP or TCP as its underlying protocol and allows a local network to assign

host names within its domain independently from other domains. UDP is the preferred protocol for use with DNS; however, if the UDP response is truncated, TCP can be used.

For more information on DNS, see Section 3.6.1.

3.2.1.2 Routing Protocols

Routing Protocols allow systems on either internal or external LANs to share routing information. In addition to the Exterior Gateway Protocol (EGP), Tru64 UNIX supports the Border Gateway Protocol (BGP) and both the Routing Information Protocol (RIP) and Open Shortest Path First Protocol (OSPF) as part of the `gated` V4.0.6 routing daemon from NextHop Technologies, Inc. For more information on `gated`, see Section 3.5.6.

Exterior Gateway Protocol (EGP)

The Exterior Gateway Protocol (EGP) allows the exterior gateway of an autonomous system to share routing information with exterior gateways on other autonomous systems.

An autonomous system is a group of networks and gateways for which one administrative authority has responsibility. Gateways are interior neighbors if they reside on the same autonomous system and exterior neighbors if they reside on different autonomous systems. Gateways that exchange routing information using EGP are said to be EGP peers (neighbors). Autonomous system gateways use EGP to provide reachability information to their EGP neighbors.

EGP allows an exterior gateway to provide remote communications among systems as follows:

- Ask another exterior gateway to agree to exchange reachability information
- Continually check to ensure that its EGP neighbors are responding
- Allow EGP neighbors to exchange reachability information by passing routing update messages

EGP restricts exterior gateways by allowing them to advertise only those destination networks reachable entirely within that gateway's autonomous system. Thus, an exterior gateway using EGP passes on information to its EGP neighbors but does not advertise reachability information about its EGP neighbors.

EGP does not interpret the distance metrics that appear in routing update messages from other protocols. EGP uses the distance field to specify whether a path exists. (A value of 255 means that the network is unreachable.) The value cannot be used to compute the shorter of two

routes, unless those routes are both contained within a single autonomous system. For this reason, EGP cannot be used as a routing algorithm. As a result, there is only one path from an exterior gateway to any network.

EGP routes are predetermined in the `/etc/gated.conf` file. This contrasts with the Routing Information Protocol (RIP), which can be used within (that is, interior to) an autonomous system of Internet networks that dynamically reconfigure routes. EGP assumes that IP is the underlying protocol. See the `gated`(8) reference page for further information.

Border Gateway Protocol

The Border Gateway Protocol (BGP) is an exterior routing protocol used for exchanging routing information between autonomous systems that are either multiple transit autonomous systems or transit and stub autonomous systems. BGP operates with more capability, greater flexibility, and less required bandwidth than EGP. For example, BGP uses path attributes to provide more information about each route and, unlike EGP, maintains an autonomous system (AS) path, which provides enough information (such as the AS number of each autonomous system the route has traversed) to prevent routing loops in an arbitrary topology.

Like EGP, BGP supports both internal and external sessions. When sending routes to an external peer, BGP prepends the local AS number to the AS path so that routes received from an external peer are guaranteed to have the AS number of that peer at the start of the path.

Routes received from an internal neighbor will not in general have the local AS number prepended to the AS path and in general have the same AS path that the route had when the originating internal neighbor received the route from an external peer. Routes with no AS numbers in the path may be legitimately received from internal neighbors; these indicate that the received route should be considered internal to your own AS.

The Tru64 UNIX implementation of BGP supports three versions of the BGP protocol (Versions 2, 3, and 4). BGP Versions 2 and 3 are similar in capability and function. They will propagate only classed network routes, and the AS path is a simple array of AS numbers. BGP Version 4 will propagate fully general address-and-mask routes, and the AS path has some structure to represent the results of aggregating dissimilar routes.

Routing Information Protocol (RIP)

The Routing Information Protocol (RIP) is an implementation of a distance-vector or Bellman-Ford routing protocol for local networks. In Tru64 UNIX it is contained in the `gated` daemon from NextHop Technologies, Inc. RIP classifies routers as active and passive: active routers advertise their routes to other routers; passive routers listen and update their routes

based on the advertisements they receive but do not advertise themselves. Typically, routers run RIP in active mode, while hosts use passive mode.

A router running RIP in active mode broadcasts updates at set intervals. Each update contains paired values, where each pair consists of an IP network address and an integer distance to that network. RIP uses a hop count metric to measure the distance to a destination. The number of hops along a path from a given source to a given destination refers to the number of gateways that a datagram would encounter along that path.

For example, a router advertises directly connected networks as having a hop count of one. Networks that are reachable through another gateway are two hops away, networks that are reachable through two gateways are three hops away, and so forth. RIP chooses the path with the shortest hop count.

Of course, using hop counts to calculate shortest paths between networks may not always produce optimal results. For example, a path with a hop count of three that crosses three Ethernet connections may be substantially faster than a path with a hop count of 2 that crosses two slow-speed serial lines. To compensate for differences in network and serial line rates of transfer, administrators can configure RIP routers to advertise artificially high hop counts for slow links.

Routing Information Protocol for IPv6 (RIPng)

The Routing Information Protocol for IPv6 (RIPng) is based on the Routing Information Protocol (RIP). RIPng allows routers to exchange information for computing routes through an IPv6–based network. In Tru64 UNIX, RIPng is contained in the `ip6rtrd` daemon. The `ip6rtrd` daemon supports active mode only and should be run on only Tru64 UNIX nodes acting as IPv6 routers. Tru64 UNIX IPv6 hosts should run the `nd6hostd` daemon.

Open Shortest Path First

Open Shortest Path First (OSPF) routing protocol is a shortest path first or link-state interior gateway protocol that distributes routing information between routers in a single autonomous system. Suitable for complex networks with a large number of routers, OSPF provides equal cost multipath routing whereby packets to a single destination can be sent simultaneously by more than one network interface.

A link-state protocol dictates that each router maintains a database describing the entire AS topology, which it builds out of the collected link-state advertisements of all routers. Each participating router distributes its local state (that is, the router's usable interfaces and reachable neighbors) throughout the AS by flooding. Each multiaccess network that has at least two attached routers has a designated router and a backup designated router. The designated router floods a link-state advertisement

for the multiaccess network and has other special responsibilities. The designated router concept reduces the number of adjacencies required on a multiaccess network.

OSPF allows networks to be grouped into areas. Routing information passed between areas is abstracted, potentially allowing a significant reduction in routing traffic. OSPF uses four different types of routes, listed in order of preference:

- intra-area
- inter-area
- type 1 external
- type 2 external

Intra-area paths have destinations within the same area and inter-area paths have destinations external to the AS.

Routes imported into OSPF as type 1 routes are supposed to be from EGPs whose external metrics are directly comparable to OSPF metrics. When a routing decision is being made, OSPF will add the internal cost to the AS border router to the external metric.

Type 2 ASEs are used for EGPs whose metrics are not comparable to OSPF metrics. In this case, only the internal OSPF cost to the AS Border router is used in the routing decision.

From the topology database, each router constructs a tree of the shortest paths with itself as the root. This shortest-path tree gives the route to each destination in the AS. Externally derived routing information appears on the tree as leaves. The link-state advertisement format distinguishes between information acquired from external sources and information acquired from internal routers, so there is no ambiguity about the source or reliability of routes. Externally derived routing information (for example, routes learned from EGP or BGP) is passed transparently through the autonomous system and is kept separate from OSPF's internally derived data. Each external route can also be tagged by the advertising router, enabling a passing of additional information between routers on the borders of the autonomous system.

3.2.1.3 File Transfer Protocol

File Transfer Protocol (FTP) allows hosts to transfer files, and provides such tasks as listing remote directories, changing the current remote directory, creating and removing remote directories, and transferring multiple files in a single request. FTP maintains a secure transport by passing user and account passwords to the foreign host. FTP allows interactive user-oriented sessions.

FTP uses reliable stream transport (TCP/IP) to send the files and uses a TELNET-like connection to transfer commands and replies. FTP also understands several basic file formats, including ASCII, IMAGE, and Local 8. TCP/IP implements FTP in the `ftp` user command and the `ftpd` daemon.

3.2.1.4 Network File System Protocol over UDP Transport

The Network File System (NFS) provides access to files via standard UNIX system calls, thereby allowing any program to access files across the network. NFS uses the UDP transport layer; therefore, it has to handle lost datagrams. NFS does this by retransmitting requests if a reply has not been received within a reasonable amount of time.

Some requests can be reexecuted on the server without problems, but others (such as file deletion) cause an error if the first request reaches the server but the reply is lost. If the second request is executed, the server finds that the file does not exist and returns an error. NFS servers hold on to such replies and retransmit them if they see a duplicate request.

On the other hand, the protocol is designed so that the servers need no other state information. Server performance is improved by running multiple copies of the server daemon. Server crashes are tolerated with no special code on either client or server.

For more information on NFS, see Section 4.6.

3.2.1.5 Network File System Protocol over TCP Transport

Tru64 UNIX contains NFS support over the TCP transport. UDP may still be the preferred transport in local area networks, but for NFS access over wide area, congested, or lossy networks, TCP may perform better.

Separate threads are used to maintain performance optimizations made to the UDP code paths. The `nfsiod` daemon spawns kernel threads, instead of forking multiple processes. Each `nfsiod` thread can handle UDP or TCP mounts, so the `nfsiod` command accepts one argument.

For more information on NFS, see Section 4.6.

3.2.1.6 Telnet Protocol

The Telnet Protocol (TELNET) provides a standard method for terminal devices and terminal-oriented processes to communicate. TELNET is commonly used by terminal emulation programs that allow you to log in to a remote host. However, TELNET can also be used for terminal-to-terminal communications and interprocess communications.

TCP/IP implements TELNET in the `telnet` user command and the `telnetd` daemon.

3.2.1.7 Trivial File Transfer Protocol

The Trivial File Transfer Protocol (TFTP) can read and write files to and from a foreign host. Like FTP, TFTP can transfer files as either 8-bit NETASCII characters or as 8-bit binary data. Unlike FTP, TFTP cannot be used to list or change directories at a foreign host and it has no provisions for security, such as password protection. Data usually can be written or retrieved only in public directories.

TCP/IP implements TFTP in the `tftp` user command and in the `tftpd` daemon.

3.2.1.8 Finger Protocol

The Finger Protocol (FINGER) is an application-level Internet protocol that provides an interface between the `finger` command and the `fingerd` daemon. The `fingerd` daemon returns information about the users currently logged in to a specified remote host. If you execute the `finger` command specifying a user at a particular host, you obtain specific information about that user. The Finger Protocol must be present at the remote host and at the requesting host. FINGER uses TCP as its underlying protocol.

3.2.1.9 Simple Mail Transfer Protocol

The Simple Mail Transfer Protocol (SMTP) is the standard for mail exchange between machines attached to the Internet. It specifies the format of control messages sent between two machines to exchange electronic mail.

As its name implies, SMTP is simple in design and purpose. Its objective is to provide a reliable and efficient mail delivery system across the links between machines. SMTP does not specify the user interface.

3.2.1.10 Simple Network Management Protocol

The Simple Network Management Protocol (SNMP) is the Internet standard protocol for exchanging network management information. The SNMP agent provides a local or remote network manager with information by accessing Management Information Bases (MIBs). The `snmpd(8)` reference page discusses the configuration, security, cluster support, and supported RFCs for the SNMP agent.

Tru64 UNIX supports both industry standard (IETF RFCs) and Compaq enterprise-specific MIBs. (See Section 3.5.5 for more details.) Compaq enterprise-specific MIB specifications are located in the `/usr/share/sysman/mibs` directory.

The daemons that provide MIB support are `/usr/sbin/os_mibs`, `/usr/sbin/srvSystem_mib`, `/usr/sbin/svrMgt_mib`, `/usr/sbin/cpq_mibs`, and `/usr/sbin/clu_mibs`.

Tru64 UNIX includes Compaq Insight Manager, which uses Web-based access to SNMP MIB data. Section 2.5.1 discusses Compaq Insight Manager.

See Section 3.4.7 for information about the extensible SNMP programming interface.

3.2.1.11 POP3

The Post Office Protocol (POP3) is a client/server protocol from Qualcomm, Inc. that allows users to download their email from a mail server to a remote client. After messages are delivered to a server, the user connects to the server and downloads the messages to the client machine (a desktop or laptop computer running Windows, MacOS, UNIX, or another operating system). Thereafter, all message processing is local to the client machine and environment. This is the protocol used widely today by Internet Service Providers (ISP) to provide e-mail services for their consumers. For more information, see the *Network Administration: Services* manual.

Tru64 UNIX supports POP3 Version 2.5.3. The protocol is described in RFC 1939.

3.2.1.12 IMAP4

The Internet Message Access Protocol (IMAP4) is a client/server protocol, based on the Cyrus IMAP4 Revision 1 server from Carnegie Mellon University, that allows mail clients to access mail messages on a server. With it, a user can access his or her mail folders and manipulate the contents remotely without having to log in to the server. The protocol allows clients to create, delete, and rename mail folders, to check for new messages and remove old messages, and to retrieve messages selectively for local viewing. In addition, the user can select messages by attributes and parse messages in the RFC 822 and MIME formats. For more information, see the *Network Administration: Services* manual.

Tru64 UNIX supports IMAP4 Version 1.6.19. The protocol is described in RFC 2060.

3.2.1.13 Resource Reservation Protocol

The Resource Reservation Protocol (RSVP) is an Internet network layer defined in RFC 2205. It is one of the components in the management of network bandwidth and provides a mechanism in which quality-of-service requests for specific application data streams or flows, simplex unicast or

multicast, can be sent and received through a network. If accepted, these requests reserve a specific amount of network bandwidth for the flow.

Applications can use the RSVP API (RAPI) to request enhanced quality-of-service when the default best effort delivery is unacceptable; for example, for video and audio. The types of quality-of-service that applications may request are defined by Internet Integrated Services (RFC 1633 and RFC 2210).

RSVP on Tru64 UNIX supports FDDI and Ethernet interfaces and unicast and multicast data flows. For more information, see the *Network Programmer's Guide*.

3.2.2 Transport-Level Protocols

The TCP/IP transport-level protocols (UDP and TCP) allow application programs to communicate with other application programs. The User Datagram Protocol (UDP) and the Transmission Control Protocol (TCP) are the basic transport-level protocols for making connections between Internet hosts. When an application sends a message to the transport layer, UDP and TCP break the information into packets, add a packet header including the destination address, and send the information to the network layer for further processing.

Other protocols and applications use UDP to make datagram connections and TCP to make stream connections. The socket interface implements these protocols.

3.2.2.1 User Datagram Protocol

The User Datagram Protocol (UDP) uses datagrams to communicate between applications on Internet hosts. UDP uses destination protocol ports (abstract destination points within a machine), identified by positive integers, to send messages to one of multiple destinations on a host. The protocol ports receive and hold messages in queues until applications on the receiving host can retrieve them.

UDP relies on the underlying Internet Protocol to send its datagrams and provides the same connectionless message delivery as IP. It offers no assurance of datagram delivery or duplication protection. However, UDP allows the sender to specify source and destination port numbers for the message and also calculates a checksum of both the data and header. These two features allow the sending and receiving applications to ensure the correct delivery of a message.

3.2.2.2 Transmission Control Protocol

The Transmission Control Protocol (TCP) provides reliable stream delivery of data between Internet hosts. Like UDP, TCP relies on the underlying Internet Protocol to transport datagrams and supports the block transmission of a continuous stream of datagrams between process ports. Unlike UDP, TCP provides reliable message delivery and ensures that data is not damaged, lost, duplicated, or delivered out of order to a receiving process. Because of this transport reliability, application programmers are not required to build communication safeguards into their software.

Both TCP and UDP allow programs to send messages to and receive messages from applications on other hosts, and both use protocol ports on the host to identify the specific destination of the message. The TCP retransmission time-out value is dynamically determined for each connection, based on round-trip time.

TCP has the following operational characteristics:

- Basic Data Transfer

  TCP transfers a continuous stream of 8-bit octets in each direction between its users by packaging some number of bytes into segments for transmission through the Internet system. In general, TCP determines the best time to block and forward packets.

- Reliability

  TCP recovers data that is damaged, lost, duplicated, or delivered out of order by the Internet. To achieve this reliability, TCP assigns a sequence number to each octet it transmits, and requires a positive acknowledgment (ACK) from the receiving TCP. If the ACK is not received within the time-out interval, the data is retransmitted. At the receiver, the sequence numbers are used to correctly order segments that are received out of order and to eliminate duplicates. To detect damage, TCP adds a checksum to each segment transmitted, checks it at the receiver, and discards damaged segments.

- Flow Control

  To control how much data is sent, TCP returns the following information with every acknowledgment:

  – The sequence numbers it will accept next; these numbers are always greater than the number of the last segment that was successfully received.

  – The number of octets that the sender is allowed to transmit before receiving further permission.

- Multiplexing

Many processes within a single host can use TCP communications facilities simultaneously. TCP maintains a set of addresses (ports) within each host. TCP combines the port number with the network address and the host address to uniquely identify each connection endpoint (socket). A pair of sockets uniquely identifies each connection.

- Connections

  TCP must initialize and maintain certain status information for each data stream. The combination of this information, including sockets, sequence numbers, and window sizes, is called a connection. Each connection is unique.

3.2.3 Internet Network-Level Protocols

The Internet network-level protocols (IP, ARP, ICMP) handle machine-to-machine communications. These protocols provide for transmission and reception of transport requests and handle network-level control.

3.2.3.1 Internet Protocol Version 4

The Internet Protocol Version 4 (IPv4) is the primary network-level protocol and the format of all the data sent over the Internet. IPv4 also specifies packet processing and error handling.

IP is connectionless because it treats each packet independently. It is unreliable because it does not guarantee delivery or the order of arrival of packets. However, underlying mechanisms guarantee data integrity, assuming the packet arrives.

IP provides the interface to the network interface level protocols. The physical connections of a network transfer information in a frame with a header and data. IP uses an Internet datagram, which contains a source host address, along with sequencing and control information.

IP automatically adds an IP header to outgoing packets and removes the IP header from incoming packets before sending them to higher level protocols. IP provides for the universal addressing of hosts in the Internet network.

IP is not a reliable communications facility because it does not require acknowledgments from the sending host, the receiving host, or intermediate hosts.

The total length of IP packets can be configured independently for each interface. Packets that are too large for a given interface can be split into multiple, smaller packets (fragments). The original packet is reassembled from the fragments when they reach their destination.

IPv4 Multicasting

The operating system supports IPv4 multicasting on a Local Area Network (LAN), as described in RFC 1112, and also supports Version 3.5 of the IPv4 multicast kernel and Version 3.6 of the `mrouted` implementation of the Distance Vector Multicast Routing Protocol (DVMRP), which provides support for "tunnelling" and "pruning."

Unlike IPv4 broadcasting, IPv4 multicasting allows packets to be taken off the network only by those clients who have configured their systems to receive the packets. Packets are accepted or rejected at the hardware level, thereby greatly reducing processing overhead. In addition, IPv4 multicasting does not consume much network bandwidth, because applications do not have to send separate packets with identical data to reach several destinations, as they do with point-to-point connections. With IPv4 multicasting, one packet is sent to all interested hosts.

As a result, IPv4 multicasting is valuable to video conferencing applications and applications that provide constant updates to ever-changing information, like applications that provide stock market quotes.

The IPv4 multicasting code was taken from the public domain and is supported on all Ethernet and FDDI adapters.

3.2.3.2 Internet Protocol Version 6

The Internet Protocol Version 6 (IPv6) is the emerging network-level protocol and the format of data sent over the Internet. IPv6 also specifies packet processing and error handling.

IPv6 is supported over the following network interface types:

- Ethernet
- Fiber Distributed Data Interface (FDDI)
- Point-to-Point Protocol (PPP)
- IP tunnel
- Loopback

Neighbor Discovery

IPv6 nodes on the same link use Neighbor Discovery to discover each other's presence, to determine each other's link-layer addresses, to find routers, and to maintain reachability information about the paths to active neighbors.

Nodes (hosts and routers) use the Neighbor Discovery protocol to determine the link-layer addresses for neighbors known to reside on attached links and to quickly purge cached values that become invalid. Hosts also use

Neighbor Discovery to find neighboring routers that are willing to forward packets on their behalf. Nodes also use the protocol to actively keep track of which neighbors are reachable and which are not and to detect changed link-layer addresses.

Multicast Listener Discovery

Multicast Listener Discovery (MLD) enables each IPv6 router to discover the presence of multicast listeners (nodes that want to receive multicast packets) on its directly attached links and to discover specifically which multicast addresses are of interest to those neighboring nodes. This information is then provided to whichever multicast routing protocol is being used by the router, to ensure that multicast packets are delivered to all links where there are interested receivers.

Internet Control Message Protocol Version 6 (ICMPv6)

The Internet Control Message Protocol Version 6 (ICMPv6) is a required part of every IPv6 implementation. ICMPv6 handles error and control messages for IPv6.

ICMPv6 does the following:

- Tests whether a destination is alive and reachable
- Reports parameter problems with a datagram header
- Performs multicast listener discovery
- Obtains IPv6 addresses
- Solicits and processes router advertisements
- Performs duplicate address detection
- Solicits and processes link-layer addresses of neighbors
- Provides transport-level reachability information
- Updates routing information

ICMPv6 provides feedback about problems in the communications environment but does not make IPv6 reliable. That is, ICMPv6 does not guarantee that an IPv6 packet will be delivered reliably or that an ICMPv6 message will be returned to the source host when an IPv6 packet is not delivered or is incorrectly delivered.

Commands and Daemons Supporting IPv6

The following commands and daemons have been modified to work in both an IPv6 and IPv4 environment:

/sbin/dump	/sbin/ifconfig	/sbin/init.d/inet	/sbin/init.d/route
/sbin/named	/sbin/ping	/sbin/restore	/sbin/route
/usr/bin/finger	/usr/bin/ftp	/usr/bin/nrdist	/usr/bin/rcp
/usr/bin/rdist	/usr/bin/rdistd	/usr/bin/telnet	/usr/sbin/dump
/usr/sbin/ftpd	/usr/sbin/ifconfig	/usr/sbin/inetd	/usr/sbin/named
/usr/sbin/nd6hostd	/usr/sbin/netstat	/usr/sbin/ping	usr/sbin/pppd
/usr/sbin/rcinet	/usr/sbin/rdump	/usr/sbin/restore	/usr/sbin/rexecd
/usr/sbin/rlogind	/usr/sbin/route	/usr/sbin/rrestore	/usr/sbin/rshd
/usr/sbin/rsvpd	/usr/sbin/rwhod	/usr/sbin/tcpdump	/usr/sbin/telnetd
/usr/sbin/traceroute			

3.2.3.3 Address Resolution Protocol

The Address Resolution Protocol (ARP) maps 32–bit IPv4 addresses into 48–bit hardware addresses for Ethernet and FDDI links. ARP does not translate addresses for the Serial Line Interface Protocol (SLIP) or Point-to-Point Protocol (PPP) because SLIP and PPP have no hardware addresses.

ARP dynamically traces IPv4 addresses to hardware addresses on local area networks. The result of this tracing is called a map. The mapped information is stored in mapping tables. TCP/IP uses ARP to collect and distribute the information for mapping tables.

The kernel maintains the mapping tables, and ARP is not directly available to users or applications. When an application sends an Internet packet to an interface driver, the driver requests the appropriate address mapping. If the mapping is not in the table, an ARP broadcast packet is sent through the requesting interface driver to the hosts on the local area network.

When a host that supports ARP receives an ARP request packet, the host notes the IPv4 and hardware addresses of the requesting system and updates its mapping table, if necessary. If the receiving host's IPv4 address does not match the requested address, the host ignores the request packet. If the IPv4 address does match, the receiving host sends a reply packet to the requesting system. The requesting system stores the new mapping and uses it to transmit future IPv4 packets.

Unlike most protocols, ARP packets do not have fixed-format headers. Instead, the message is designed to be useful with a variety of network technologies.

3.2.3.4 Internet Control Message Protocol

The Internet Control Message Protocol (ICMP) is a required part of every IPv4 implementation. ICMP handles error and control messages for IPv4.

ICMP does the following:

- Tests whether a destination is alive and reachable
- Reports parameter problems with a datagram header
- Performs clock synchronization and transit time estimations
- Obtains Internet addresses and subnet masks
- Provides transport-level reachability information
- Updates routing information

ICMP provides feedback about problems in the communications environment, but does not make IPv4 reliable. That is, ICMP does not guarantee that an IPv4 packet will be delivered reliably or that an ICMP message will be returned to the source host when an IPv4 packet is not delivered or is incorrectly delivered.

ICMP messages are sent in varying situations, including the following:

- When a packet cannot reach its destination
- When a gateway host does not have the buffering capacity to forward a packet
- When a gateway can direct a host to send traffic on a better route

3.3 Network Interface Layer

Tru64 UNIX supports the following network transports:

- Asynchronous Transfer Mode
- Ethernet
- Fast Ethernet
- Gigabit Ethernet
- Fiber Distributed Data Interface
- Token Ring
- Serial Line IP and Compressed Serial Line IP
- Point-to-Point Protocol
- Multiple Adapters
- Link Aggregation
- NetRAIN

3.3.1 Asynchronous Transfer Mode

Tru64 UNIX supports PCI and TURBOchannel machines on 155.5 Mb per second Asynchronous Transfer Mode (ATM) networks; there is also an adapter for the PCI bus that supports 622 Mb ATM networks. ATM is a high-speed, connection-based, cell-switched network that, unlike traditional packet switched networks, can carry different kinds of traffic (voice, video, and data) simultaneously. In addition, ATM provides predictable services to those classes of traffic that require bounded latencies and dedicated bandwidths, and, because ATM separates the physical interface from the datalink layer, the same cell and packet formats can be used over a wide variety of physical interfaces from 1 MB per second to 10 GB per second.

The Tru64 UNIX implementation of ATM consists of permanent virtual circuit support; switched virtual circuit support through ATM Forum UNI 3.0 and 3.1 signalling for point-to-point connections; ATM Forum Integrated Local Management Interface (ILMI) for dynamic network address registration; Classical IP (as defined by RFC 1577, RFC 1483, and RFC 1626); and LAN Emulation over ATM (as defined by The ATM Forum Version 1 standard). For more information on ATM, see the *Asynchronous Transfer Mode* manual and the *Network Administration: Connections* manual.

3.3.2 Ethernet Networks

Tru64 UNIX supports 10 MB per second Ethernet networks.

At the physical and IP levels, Tru64 UNIX supports an Ethernet network with a Maximum Transfer Unit (MTU) of 1500 bytes at a maximum of 10 MB per second.

The default MTU at the IP level is 1500 bytes at a maximum of 10 MB per second, although this value can be decreased using the `ifconfig` command.

In conformance with Ethernet standards, Tru64 UNIX always ensures a minimum packet size of 60 bytes.

3.3.3 Fast Ethernet Networks

Tru64 UNIX supports 100 MB per second Fast Ethernet (IEEE 802.3 100Base-TX) networks in full and half duplex.

MTU sizes at the physical and IP levels are the same as those for regular 10 MB per second Ethernet.

3.3.4 Gigabit Ethernet Networks

Tru64 UNIX supports 1000 MB per second Gigabit Ethernet (IEEE 802.3z 1000Base-T) networks on all PCI-based Alpha hardware platforms. It

supports symmetric and asymmetric Pause Frame Flow control (X-on/X-off) and Jumbo frame compatibility.

MTU sizes at the physical and IP levels are the same as those for regular Ethernet networks, although this value can be changed using the `ifconfig` command.

3.3.5 Fiber Distributed Data Interface Networks

Tru64 UNIX supports 100 MB per second Fiber Distributed Data Interface (FDDI) networks in conformance with RFC 1042 and RFC 1188 on all Alpha hardware platforms.

At the physical level, Tru64 UNIX supports an FDDI network with a Maximum Transfer Unit (MTU) of 4500 bytes at a maximum of 100 MB per second. At the IP level, the MTU is 4352 bytes at a maximum of 100 MB per second.

The default MTU at the IP level is always 4,352 bytes at a maximum of 100 MB per second, although this value can be decreased using the `ifconfig` command.

3.3.6 Token Ring Networks

Tru64 UNIX supports 4 MB per second and 16 MB per second Token Ring networks and source routing in conformance with RFC 1042.

At the physical level, Tru64 UNIX supports a Token Ring network with a Maximum Transfer Unit (MTU) of 4472 bytes at a maximum of 4 MB per second and 17800 bytes at a maximum of 16 MB per second. At the IP level, the MTU is 4092 bytes at a maximum of 4 MB per second and 8188 bytes at a maximum of 16 MB per second.

The default MTU at the IP level is always 4092 for both 4 and 16 MB per second, although this value can be increased or decreased using the `ifconfig` command.

3.3.7 Serial Line IP and Compressed Serial Line IP

The operating system has complete IP support for a serial line, so that users can transfer files or NFS-mount file systems across phone lines. The CSLIP `slattach` command can compress headers to improve performance.

The SLIP/CSLIP code is from OSF/1 Version 1.0. However, because the OSF/1 code did not provide a way to access the CSLIP feature, Compaq modified the `slattach` command to provide the necessary access to CSLIP.

3.3.8 Point-to-Point Protocol

The Point-to-Point (PPP) protocol provides a method for transmitting datagrams over serial point-to-point links. Unlike SLIP, PPP supports standard encapsulation, simultaneous multiplexing of different network layer protocols, an HDLC frame check sequence for error detection, an HDLC escaping mechanism for use with miscellaneous non-8-bit-transparent telephone and switching equipment, and the dynamic negotiation of IP addresses.

In addition, while SLIP supports only `clist tty` drivers, PPP supports both `clist` and STREAMS-based `tty` drivers, as well as remote logins over LANs.

Note that the PPP code was taken from the public domain and includes contributions identified by the footnoted copyright notices.[1] Certain sections of the PPP code were derived from the RSA Data Security, Inc., MD5 Message-Digest Algorithm.

Tru64 UNIX supports PPP Version 2.3.1. (RFC 1661)

For more information on PPP, see the *Network Administration: Connections* manual and the `pppd`(8), `pppstats`(8), and `chat`(8) reference pages.

3.3.9 Multiple Adapter Support

Tru64 UNIX supports single systems that have multiple active network adapters in the same subnet; for example, consider tu0 configured with IP address 192.24.156.20 and tu1 configured with IP address 192.24.156.21, both with the same netmask.

On connection establishment, the kernel chooses the interface that has the fewest number of connections. This connection-balancing effect could lead to greater throughput than on a system with just one network adapter per subnet. See the *Network Administration: Connections* manual for information on configuring multiple adapter support.

3.3.10 Link Aggregation

Link aggregation, also called trunking, enables administrators to combine two or more physical Ethernet Network Interface Cards (NICs) and create a single logical link. (Upper-layer software sees this link aggregation group as a single logical interface.) The single logical link can carry traffic at higher data rates than a single interface because the traffic is distributed across all the physical ports that make up the link aggregation group.

[1] © 1993 The Australian National University. © 1989 Carnegie Mellon University. © 1991 Gregory M. Christy. © 1989 Regents of the University of California. © 1990 RSA Data Security, Inc.

Using link aggregation provides the following capabilities:

- Increased network bandwidth

  The increase is incremental based on the number and type of ports, or NICs, added to the link aggregation group.

- Fault tolerance

  If a port in a link aggregation group fails, the software detects the failure and reroutes traffic to the other available ports.

- Load sharing

  Traffic is distributed across all ports of a link aggregation group.

You can use a link aggregation group virtual interface for the following point-to-point connections: server-to-server and server-to-switch.

See the *Network Administration: Connections* manual and `lag`(7) and `lagconfig`(8) for information on link aggregation configuration.

3.3.11 NetRAIN

NetRAIN (Redundant Array of Independent Network adapters) detects the physical loss of network connectivity and automatically switches network traffic to a working network interface.

The NetRAIN virtual interface configures two or more interfaces on the same LAN segment into a single interface. One of the physical interfaces is always active while the others remain idle. All interfaces, including the idle interfaces, are constantly monitored to ensure that traffic can flow on each.

If the active interface fails or loses network connectivity, NetRAIN switches network traffic to the next available, working interface. All the context of the previous interface is maintained (for example, hardware address and multicast addresses). The actual failover time is adjustable depending on your network configuration and operation.

Tru64 UNIX provides support for Ethernet, FDDI, and ATM controllers.

See the *Network Administration* manual and the `nr`(7) and `ifconfig`(8) reference pages for information on NetRAIN configuration.

3.4 Application Programming Interfaces

The network programming environment includes the programming interfaces for application, kernel, and driver developers who write network applications and implement network protocols. It also includes the kernel-level resources that an application requires to process and transmit data, some of which include libraries, data structures, header files, and transport protocols.

The following sections briefly discuss the supported application programming interfaces:

- X/Open Transport Interface (XTI/TLI)
- BSD Sockets
- System V Release 4.0 STREAMS Framework
- Data Link Interface (DLI)
- Data Link Provider Interface (DLPI)
- Extensible SNMP (eSNMP)
- Basic Sockets API for IPv6
- RSVP API (RAPI)

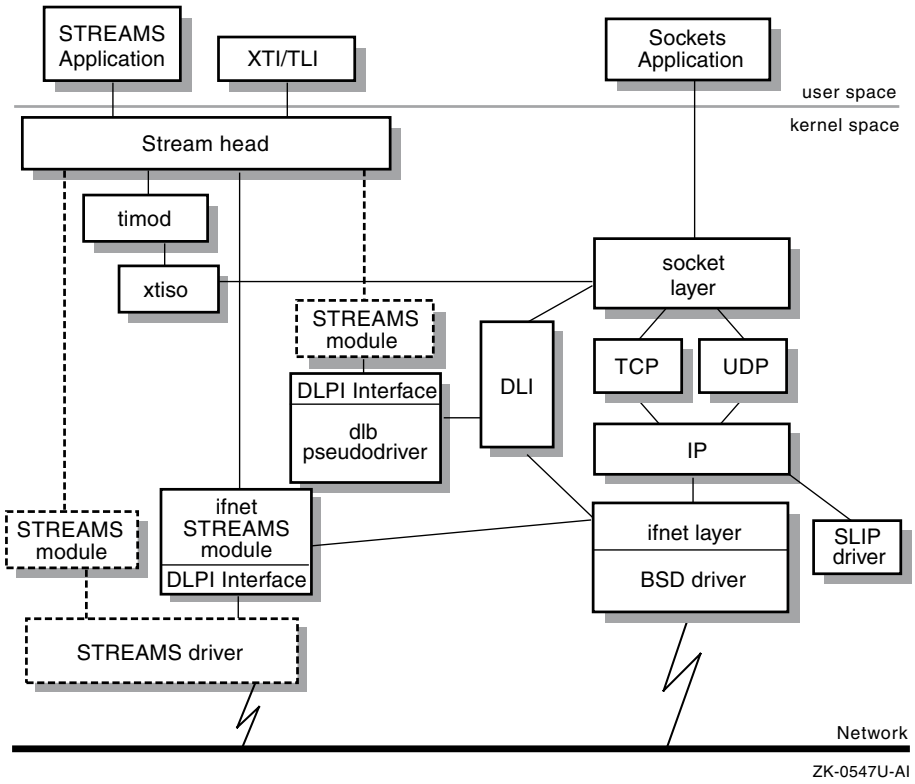
For more information on the network programming environment, see the *Network Programmer's Guide*.

3.4.1 X/Open Transport Interface

The X/Open Transport Interface (XTI) defines a transport layer application interface that is independent of any transport provider. This means that applications written to XTI can be run over a variety of transport providers, such as the Transmission Control Protocol (TCP) or the User Datagram Protocol (UDP). The application specifies which transport provider to use.

Figure 3–2 illustrates the interaction between XTI and the STREAMS and sockets frameworks.

Figure 3–2: Interaction Between XTI and STREAMS and Sockets Frameworks



Depending on the transport provider specified by the application, data can flow along one of two paths:

- If a STREAMS-based transport provider is specified, data follows the same route that it did for an application written to run over STREAMS. It passes first through the Stream head, then to any modules that the application pushed onto the STREAM, and finally to the STREAMS driver, which puts it onto the network. Tru64 UNIX does not provide any STREAMS-based transport providers.
- If a socket-based transport provider (TCP or UDP) is specified, data is passed through `timod` and `xtiso`. The appropriate socket layer routines are called and the data is passed through the Internet protocols and `ifnet` layer to the BSD-based driver, which puts it on to the network.

3.4.2 Sockets

Sockets are the industry standard programming interface. In Tru64 UNIX, sockets are the interface to the Internet Protocol suite; for example, TCP,

UDP, IP, ARP, ICMP, and SLIP. Tru64 UNIX supports the 4.3BSD (the default), 4.4BSD, XNS5.0, XNS4.0, and POSIX 1003.1g Draft 6.6 interfaces.

The sockets framework consists of a series of system and library calls, header files, and data structures. Applications can access networking protocols through socket system calls and can use socket library calls to manipulate network information. For example, the `getservent` library call maps service names to service numbers, and the `htonl` library call translates the byte order of incoming data to that appropriate for the local system's architecture.

With sockets, the application in user space passes data to the appropriate socket system calls, which then pass it to the network layer. Finally, the network layer passes it, via the `ifnet` layer, to the BSD driver, which puts it on the network.

For more information on sockets, see the *Network Programmer's Guide*, X/Open CAE Specification, Networking Services (XNS), Issue 5; *X/Open CAE Specification, Networking Services, Issue 4* (XNS4.0); Protocol Independent Interfaces (POSIX 1003.1g Draft 6.6, Section 5); and the `netintro`(7) reference page.

3.4.3 STREAMS

The STREAMS framework provides an alternative to sockets. The STREAMS interface consists of system calls, kernel routines, and kernel utilities that are used to implement everything from networking protocol suites to device drivers. Applications in user space access the kernel portions of the STREAMS framework using system calls, such as `open`, `close`, `putmsg`, `getmsg` and `ioctl`.

Tru64 UNIX supports System V Release 4.0 STREAMS from the OSF/1 Version 1.2 code base, which provides support for the STREAMS `tty` interface (although Tru64 UNIX continues to support the existing CLIST and Berkeley-based `tty` interface). For more information on STREAMS, see the *Network Programmer's Guide*.

3.4.4 Sockets and STREAMS Interaction

Tru64 UNIX provides the `ifnet` STREAMS module to allow programs using BSD-based TCP/IP to access STREAMS-based drivers. The module provides the Data Link Bridge (DLB) pseudodriver to allow programs using a STREAMS-based protocol stack to access BSD-based drivers.

3.4.5 Data Link Interface

The Data Link Interface (DLI) is provided as a backward compatibility feature to the ULTRIX operating system. DLI support allows programs written to DLI on the ULTRIX operating system to access the data link layer. For more information on DLI, see the *Network Programmer's Guide*.

3.4.6 Data Link Provider Interface

The Data Link Provider Interface (DLPI) is a kernel-level interface that maps to the data link layer of the OSI reference model. DLPI frees users from specific knowledge of the characteristics of the data link provider, allowing those characteristics to be implemented independently of a specific communications medium. It is primarily a kernel-level interface targeted for STREAMS modules that use or provide data link services.

Tru64 UNIX supports only a partial subset of the DLPI interface. For more information, see the *Network Programmer's Guide*.

3.4.7 Extensible SNMP Interface

Tru64 UNIX supports extensible SNMP (eSNMP), an application-layer Application Programming Interface (API) that permits user-written programs to function as part of a distributed SNMP agent on a Tru64 UNIX host system.

User programs (subagents) can dynamically register SNMP MIB objects with the eSNMP master agent (`/usr/sbin/snmpd`) and can subsequently handle the SNMP protocol operations for those objects.

The distribution of MIB objects between cooperating processes is transparent to SNMP applications, which can access all MIB objects using the standard transport endpoints specified in the SNMP RFCs. The eSNMP API (`libesnmp.so`) uses RFC 2741 (AgentX) to communicate with `snmpd`. This backward-compatible change permits subagents to interoperate with any SNMP agent conforming to RFC 2741.

The extensible SNMP development tools are contained in the optional programming subset (PMR). For more information, see the *Network Programmer's Guide*.

3.5 Network Administration Software

Tru64 UNIX supports a variety of network administration software, which is briefly described in the following sections.

3.5.1 Networking Commands and Utilities

Tru64 UNIX supports the entire suite of networking commands from OSF/1 Version 1.2, including `finger`, `ftp`, `rdump`, `rdist`, `routed` and `gated`, `telnet`, and `tftp`. The `bootpd` functions have been folded into the `joind` daemon, which provides configurations to clients using either the DHCP or BOOTP protocol.

Additionally, Tru64 UNIX supports the following Open Network Computing (ONC) Version 4.2 utility programs, which are invoked by the `inetd` daemon: `rusers` and `rusersd`, `rwall` and `rwalld`, `spray` and `rsprayd`, `rup` and `rstatd`, `rquotad`, and `pcnfsd`.

3.5.2 Ethernet Packet Filter and Packet Filter Applications

The Ethernet packet filter is a software driver interface that demultiplexes networking packet headers, as well as provides reception and transmission of packets containing user-defined network protocols. The packet filter can function also as an Ethernet monitor when used to filter specific network protocols.

Note

The Ethernet packet filter is an optional kernel subsystem; application programs that make calls to the packet filter kernel routines may fail if the packet filter is not configured in the currently running kernel. For more information, see the `packetfilter`(7) reference page.

Tru64 UNIX supports the following packet filter applications:

- Reverse Address Resolution Protocol daemon (`/usr/sbin/rarpd`)

  The `rarpd` daemon uses the `/etc/ethers` file to map the Ethernet address of a machine to the machine's Internet Protocol (IP) address. It can serve IP addresses to remote PC clients.

  Some ULTRIX systems use the Reverse Address Resolution Protocol (RARP) to supply remote stations with their IP address.

- TCP/IP tracing and monitoring utility (`/usr/sbin/tcpdump`)

  Tru64 UNIX supports Version 2.2.1 of the `tcpdump` utility. This version of `tcpdump` uses the Berkeley Packet Filter (BPF) language.

  The `tcpdump` utility is used to debug and analyze TCP/IP network activity on Ethernet, Token Ring, Memory Channel, FDDI, and ATM networks; it and has some support for other protocol suites and interfaces (including PPP, NFS, and loopback). The `tcpdump` utility includes

software developed by the University of California, Lawrence Berkeley Laboratory and its contributors.

- Log file utility (`/usr/sbin/tcpslice`)

  The `tcpslice` utility manipulates `tcpdump` trace log files by either extracting pieces of or joining `tcpdump` log files. It can select portions of a large `tcpdump` log file and display selected traces without having to dump the entire log file.

- NFS monitoring utility (`/usr/sbin/nfswatch`)

  Tru64 UNIX supports Version 4.1 of `nfswatch` from Purdue University. The `nfswatch` utility is curses-based and displays the NFS traffic between any number of NFS servers and clients on a LAN.

- NFS log file summary utility (`/usr/sbin/nfslogsum`)

  The `nfslogsum` utility condenses the log files produced by `nfswatch` into a traffic analysis summary and is useful for troubleshooting networks.

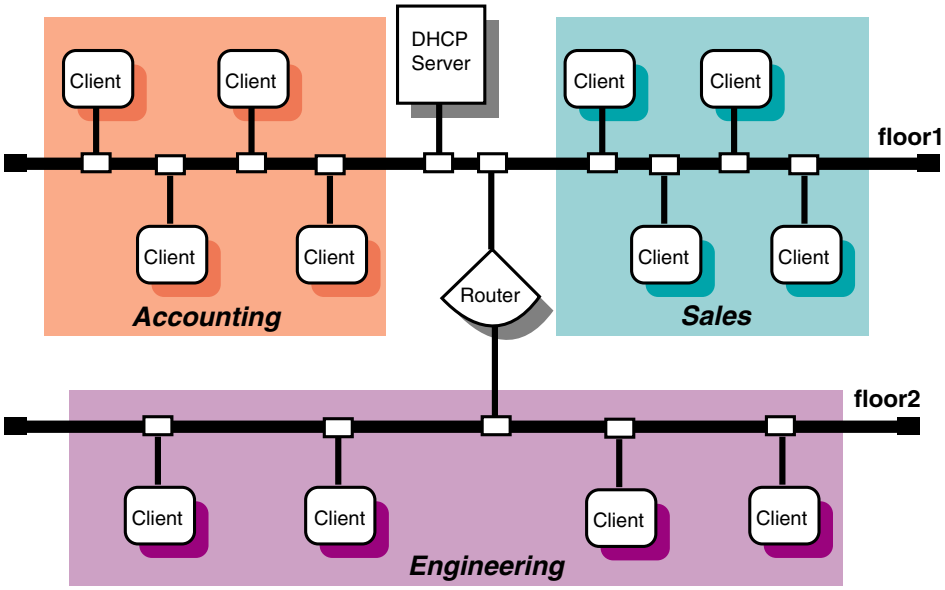
3.5.3 Dynamic Host Configuration Protocol

Tru64 UNIX supports the Dynamic Host Configuration Protocol (DHCP) based on JOIN Server Version 4.1 from JOIN Systems, Inc.; it is a client/server framework in which the DHCP server can dynamically assign an IP address to a client as the client boots on to the network. Additionally, a DHCP server can provide configuration information to the client, such as the name of the DNS server or the name of the default router for that client.

For example, when a new system is booted for the first time, the DHCP server assigns that system a unique IP address. If that system is moved to another location on the same LAN (perhaps on a different subnet), the DHCP server ensures that a new IP address appropriate to that subnet is assigned to the system, if necessary, when it boots up for the first time.

With DHCP (see Figure 3–3), customers with hundreds of clients no longer have to worry about the assignment of IP addresses; DHCP automatically assigns IP addresses and requires no intervention by a system administrator.

Figure 3–3: DHCP Configuratrion



For more information on DHCP, see the *Network Administration: Connections* manual and the `dhcp`(7) reference page.

3.5.4 The Internet Boot Protocol Daemon

The Internet boot protocol daemon, `bootpd`, implements an Internet boot protocol (BOOTP) server as defined in RFC 951, RFC 1542, and RFC 2132.

BOOTP is an extensible UDP/IP-based protocol that allows a booting host to configure itself dynamically without having to rely on user intervention. The BOOTP protocol assigns IP addresses to hosts, makes available a file containing a boot program that can be downloaded from a server, provides the address of that server, and the address of an Internet gateway, if one is present.

Like DHCP, the BOOTP protocol supports the centralized management of network addresses.

The `joind` daemon provides the `bootpd` functions; the `joind` daemon also provides DHCP services.

3.5.5 SNMP Agent

The SNMP agent for Tru64 UNIX provides access to a great deal of management information typically used in network administration:

- System, network interface, address resolution (ARP), routing (IP and ICMP), and transport layer (TCP and UDP) information is available through the Internet MIB (RFC 1213).
- Host information, such as processes, file systems, memory, and attached devices, is available via the Host Resources MIB (RFC 1514).
- FDDI interface information is available via the FDDI MIB (RFC 1285).
- Token Ring interface information is available via the Token Ring MIB (RFC 1748).
- Ethernet counters are available via the Ethernet-like Interface MIB (RFC 1398).
- IP routing-related MIBs are available, as described in Section 3.5.6.
- Server information, such as firmware, memory, device configuration, and environmental monitoring, is available in the Server System MIB.
- Thresholds, alarms, and actions can be configured by the Server Management MIB.
- System information is available from Compaq-specific MIBS in support of Compaq Insight Manager.

The extensible SNMP agent permits the dynamic addition of supported Management Information Bases (MIBs) on any Tru64 UNIX host.

The master agent, API, and base operating system MIB support are all contained in the standard networking subset (CLINET).

The extensible SNMP development tools are contained in the optional programming subset (PGMR).

3.5.6 The gated Daemon

The `gated` daemon allows any host with multiple network interfaces to function as an IP router by participating in various IP routing protocols (for example, RIP, OSPF, EGP, and BGP). Tru64 UNIX supports the GateD Release 4.0.6 `gated` daemon from NextHop Technologies, Inc., which contains support for the following:

- RIP Version 1 (RFC 1058)

  Stipulates the proper subsuming of host routes, split horizon without poison reverse, and graceful shutdowns.

- RIP Version 2 (RFC 1388)

  Stipulates using IP multicast where available; supports classless routing; and uses next hop (if different).

- OSFP Version 2 (RFC 1583)

Uses local-wire IP multicast support, MIB support (RFC 1253), and reconfiguration.

- Routing Table MIB (RFC 1354)
- EGP 2 (RFC 904)

  Complete implementation of the specification, with optimizations for MILNET.

- BGP Versions 2 and 3 (RFC 1163 and RFC 1267)

  Complete implementations of the BGP MIB (RFC 1269), AS path pattern matching (RFC 1164), and OSPF/BGP Interaction (RFC 1403) specifications

- BGP Version 4 (RFC 1654)
- Variable subnet masks through routing socket support and improved synchronization of the kernel routing table
- Routing Table Enhancements

  Based on BSD 4.3 Reno radix tree, `gated` implements filtered routing that is based on policy. This enables network administrators to control the import and export of routing information by individual protocol, by source and destination autonomous system, by source and destination interface, by previous hop router, and by specific destination address.

  Network administrators can also specify a preference level for each combination of routing information being imported by using a flexible masking capability. Once the preference levels are assigned, `gated` decides which route to use — independent of the protocols involved.

- MIB Protocol Support ("Get Object" Support Only)
  - OSPF V2 MIB (RFC 1253)
  - EGP MIB (RFC 1213)
  - BGP V3 MIB (RFC 1269)

For more information on the `gated` daemon, see the `gated`(8), `gated.conf`(4), `gated.control`(4), `gated.proto`(4), and `gated_intro`(7) reference pages.

3.5.7 The screend Daemon

The `screend` daemon is used with the gateway screen facility to decide which IP packets should be forwarded when the system is acting as an IP gateway.

The gateway packet screening facility, on a Tru64 UNIX system acting as a gateway, allows the system manager to control which packets are forwarded

or rejected. As a result, the gateway packet screening facility can be used as one part of a comprehensive network security policy.

The facility consists of a kernel-resident mechanism and a user-level daemon, `/usr/sbin/screend`. When a packet is ready to be forwarded, the kernel mechanism submits the packet's headers to the daemon. The `screend` daemon then examines the headers and tells the kernel to forward or reject the packet, based on a set of rules defined in the configuration file, `/etc/screend.conf`.

Optionally, some or all decisions can be logged, thereby enabling the network manager to detect improper configurations or potential security problems.

3.5.8 UNIX-to-UNIX Copy Program

Tru64 UNIX supports the HoneyDanBer version of the UNIX-to-UNIX Copy Program (UUCP), which is a group of programs that supports communications between two computers running UNIX operating systems. UUCP enables batched, error-free file transfer and remote command execution between two UNIX systems. UUCP is most frequently used to transfer electronic mail, network news, and public domain software over low-speed, low-cost communications links.

UUCP supports only direct connections between two systems; electronic news and mail delivery depend on third-party forwarding. To facilitate mail and news delivery, most connected sites are willing to relay files for other sites. The UUCP network depends on direct distance dialing networks and off-peak long distance rates for its continued functioning. For more information on UUCP, see the `uucp_setup`(8) reference page.

3.5.9 Local Area Transport

Local Area Transport (LAT) is a protocol that supports communications between host computer systems and terminal servers with terminals, PCs, printers, modems, and other devices over local area networks (LANs). LAT software has the features required for a host to function as a service node, so requests for connections can be made by server users. The software also permits host applications to initiate connections to server ports, designated as application ports, to access remote devices. The LAT driver is STREAMS-based and supports up to 4000 incoming connections, with a theoretical limit of 5000 users.

In Tru64 UNIX, LAT supports both SVR4 and BSD-style terminal devices. Integral serial terminal devices and serial terminal options share the same BSD `tty` namespace as LAT, which means that if special files are allocated for serial lines, those special files will reduce the number of BSD LAT devices that can be configured.

For more information on LAT, see the `lat_intro`(7) reference page and the *Network Administration: Connections* manual.

3.5.10 Network Interface Monitoring

The `niffconfig` command arranges for one or more network interfaces to be monitored for possible loss of connectivity. Timing parameters that govern how quickly an interface can be declared suspect or dead can be manipulated with this command.

Once an interface has been specified for monitoring, the kernel Traffic Monitor Thread (TMT) checks the connectivity of the monitored interface and, if necessary, informs the Network Interface Failure Finder daemon (`niffd`) to generate traffic for the network interface that has been classified inactive. The `niffd` daemon's purpose is to get the interface packet counters to increment, signifying the interface is still alive and well.

See the *Network Administration: Connections* manual and the `niffconfig`(8), `niffd`(8), and `niffmt`(7) reference pages.

3.6 Naming Services

Tru64 UNIX supports the following distributed naming services:

- The Domain Name Service (DNS)
- The Network Information Service (NIS), formerly called Yellow Pages (YP)

The library routines in `/usr/lib/libc.so` allow transparent access to DNS, NIS, and local `/etc` files. The name services configuration file, `/etc/svc.conf`, dictates which naming services are queried, and in what order, for a particular database.

Tru64 UNIX allows you to convert from an NIS-distributed environment to a DNS-distributed environment or to run both services in the same environment. Because the source files for both DNS and NIS can be `/etc` style files, a distributed Berkeley Software Distribution (BSD) source area can be shared between the two services by means of symbolic links.

3.6.1 Domain Name System

The Domain Name System (DNS) is a mechanism for resolving unknown host names and Internet Protocol (IP) addresses that originate from sites on your company's intranet or the Internet.

The implementation of DNS in Tru64 UNIX is based on the Berkeley Internet Name Domain (BIND) service, which is supported by the Internet

Software Consortium. BIND service is a client/server model that allows client systems to obtain host names and addresses from DNS servers.

In this hierarchical naming system, local resolver routines may resolve Internet names and addresses using a local name resolution database maintained by the `named` daemon. If the name requested by the host is not in the local database, the resolver routine or the local named daemon queries the remote DNS name server.

Tru64 UNIX supports BIND Version 8.2.2 (based on RFC 1034 and RFC 1035), which includes the following features:

- Notification and dynamic update of slave servers by master server when DNS resource records are changed (RFC 1996 and RFC 2136)
- Dynamic update of DNS resource records with IPv6 or Microsoft Windows clients are added to a network (RFC 2136).
- A flexible, categorized logging system
- IP-address-based access control and authentication for queries, zone transfers, and updates

You can use DNS to replace or supplement the host table mapping provided by the local `/etc/host` file or NIS. You should use NIS for all other distributed database applications.

For more information about the DNS environment, DNS planning and configuration, and DNS management, see the *Network Administration: Services* manual, the *BIND Configuration File Guide*, and the `bind_intro`(7) reference page.

3.6.2 Network Information Service

The Network Information Service (NIS) is a distributed name service that allows participating hosts to share access to a common set of system and network files. NIS allows system administrators to manage these shared files on a single system.

NIS is intended for use in a secure environment only, where gateways do not allow outside access from the Internet to the NIS protocol.

3.7 Time Services

Tru64 UNIX supports the following time services:

- Network Time Protocol (NTP)
- Time Synchronization Protocol (TSP)

Because NTP can be traced to clocks of high absolute accuracy, it provides a more accurate time service than TSP. By contrast, TSP synchronizes time to the average of the network host times. TSP is an acceptable time service if your system is not on the Internet and does not have access to a highly accurate time server; otherwise, NTP is recommended.

3.7.1 Network Time Protocol

The Network Time Protocol (NTP) provides accurate, dependable, and synchronized time for hosts on both wide area networks (like the Internet) and local area networks. In particular, NTP provides synchronization traceable to clocks of high absolute accuracy, and avoids synchronization to clocks keeping bad time.

Hosts running NTP periodically exchange datagrams querying each other about their current estimate of the time. Using the round-trip time of the packet, a host can estimate the one-way delay to the other host. (The delay is assumed to be roughly equal in both directions.) By measuring the one-way delay and examining the timestamps that are returned with the NTP packet, a host computes the difference between its clock time and that of the host it queried.

A host queries a remote host several times over a period and feeds the results from the multiple samples to a digital-filtering algorithm. The algorithm provides a more accurate estimate of the delay, clock offset, and clock stability than could be obtained with a single sample.

NTP messages also contain information about the accuracy and reliability of the time sources. An NTP host connected directly to a highly accurate time source, such as a radio receiver tuned to a time code signal broadcast by a government agency, is called a stratum 1 server. Every other NTP host adopts a stratum number that is one higher than the host from which it sets its own time. For example, a host synchronized to a stratum 1 server becomes a stratum 2 host. Stratum determination is done automatically, and the stratum of a host can vary as its connectivity changes.

A host running NTP collects information to decide which of the hosts it queries provides the most accurate time. This information includes the output of the digital-filtering algorithm and the stratum numbers of the hosts it queried. By communicating with several other hosts, an NTP host can usually detect those hosts that are keeping bad time and can stay synchronized even if some of the other hosts become unavailable for long periods.

In practice, NTP can synchronize clocks to within a few tens of milliseconds even over wide-area networks spanning thousands of miles.

Tru64 UNIX supports NTP Version 4. New features in the Tru64 UNIX Version 5.1 release include the following:

- A burst mode that results in better accuracy with intermittent connections typical of PPP and ISDN services
- Two new association modes based on multicast technology that provide for automatic discovery and configuration of servers and clients
- Improved logging support

The NTP protocol is described in RFC 1305. For more informatiion, see the *Network Programmer's Guide*, `ntp_intro`(7), and `ntp_conf`(4).

3.7.2 Time Synchronization Protocol

The Time Synchronization Protocol (TSP) is the protocol used by the `/usr/sbin/timed` daemon. In its simplest application, the TSP servers on a broadcast network (for example, an Ethernet) periodically broadcast TSP packets. The hosts on the network elect one of the hosts on the network running TSP as a master.

The master then controls the further operation of the protocol until that master fails and a new master is elected. The master collects time values from the other hosts and computes the average of all the times reported. It then sets its own clock to this average and tells the other hosts to synchronize their clocks with it.

TSP quickly synchronizes all participating hosts. However, because TSP does not trace time back to sources of known accuracy, it is unable to correct for systematic errors. If a clock drifts significantly, or if a mistake is made in setting the time on a participating host, the average time calculated and distributed by the master can be affected significantly.

4

File Systems

This chapter describes the various file systems provided by the Tru64 UNIX operating system. Following a brief overview (Section 4.1), the following file systems are discussed:

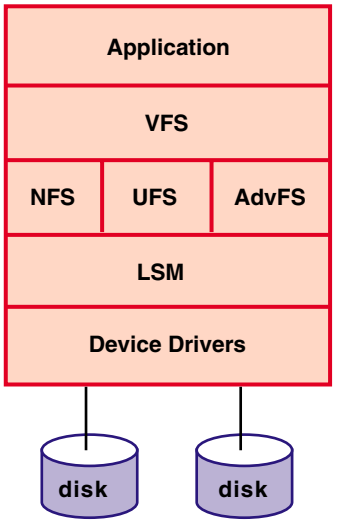
- Virtual File System (Section 4.2)
- Advanced File System (Section 4.3)
- UNIX File System (Section 4.4)
- Cluster File System (Section 4.5)
- Network File System (Section 4.6)
- CD–ROM File System (Section 4.7)
- DVD File System (Section 4.8)
- Memory File System (Section 4.9)
- The `/proc` File System (Section 4.10)
- File-on-File Mounting File System (Section 4.11)
- File Descriptor File System (Section 4.12)

4.1 Overview

The file systems provided by Tru64 UNIX are all accessed through a Virtual File System (VFS) layer, and are integrated with the virtual memory Unified Buffer Cache (UBC).

The file system that you see is handled by the Virtual File System layer, which interacts with the local file system or the networked file system. Under Tru64 UNIX, the default file system is the Advanced File System (AdvFS), although the traditional UNIX File System (UFS) is also available. From there, the networked file system or the local file system might interface with the Logical Storage Manager, and in turn, the device drivers and the physical storage devices. Figure 4–1 illustrates this interplay of the file systems, the Logical Storage Manager (LSM), and the physical storage devices.

Figure 4–1: File Systems



ZK-1577U-AI

4.2 Virtual File System

The Virtual File System (VFS) is based on the Berkeley 4.3 Reno virtual file system. VFS presents a uniform interface to users and applications, an interface that is abstracted from the file system layer to allow common access to files, regardless of the file system on which they reside. As a result, file access across different file systems is transparent to the user.

A structure known as a `vnode` contains information about each file in a mounted file system. The `vnodes` are analogous to inodes: they are more or less wrappers around file system-specific nodes. If, for example, a read or write request is made on a file, the `vnode` points the system call to the routine appropriate for that file system. A read request is pointed to `advfs_read` when the request is made on a file in AdvFS; to `ufs_read` when the request is made on a file in a UFS; or to `nfs_read` when the request is made on a file in an NFS-mounted file system.

The Tru64 UNIX VFS implementation supports Extended File Attributes (XFAs), including support for any application that wants to assign an XFA to a file. Both AdvFS and UFS support XFAs. For more information on XFAs, see the `setproplist`(2) reference page.

4.3 Advanced File System

The Advanced File System (AdvFS), the default root file system for Tru64 UNIX, provides flexibility, compatibility, data availability, high performance, and simplified system management. This log-based file system handles files and filesets approaching 16 terabytes in length.

The configuration of AdvFS differs from the traditional UNIX file system. In AdvFS, the physical storage layer is managed independently of the directory layer. System administrators can add and remove storage without unmounting the file system or halting the operating system. As a result, configuration planning is less complicated and more flexible.

From a user's perspective, AdvFS behaves like any other UNIX file system. You can use the `mkdir` command to create new directories, the `cd` command to change directories, and the `ls` command to list directory contents. AdvFS logical structures, quota controls, and backup capabilities are based on traditional file system design. AdvFS has its own complement of file system maintenance utilities, including `mkfdmn` and `mkfset`, which create file systems, and `vdump` and `vrestore`, which back up and restore filesets. AdvFS commands and utilities are described in the *AdvFS Administration* manual.

Without taking an AdvFS off line, system administrators can perform backups, file system reconfiguration, and file system tuning. End users can retrieve their own unintentionally deleted files from predefined trashcan directories or from clone filesets without assistance from system administrators.

The separately licensed AdvFS Utilities provide additional file management capabilities and a Web-based graphical user interface to simplify system administration. The graphical interface, which runs under the Common Desktop Environment (CDE), features menus, graphical displays, and comprehensive online help that make it easy to perform AdvFS operations. In addition, the graphical interface displays summarized system status information.

The AdvFS Utilities support multivolume file systems, which enables file-level striping (spreading data to more than one volume) to improve file transfer rates for I/O intensive applications. Logical Storage Manager (LSM), which allows volume-level striping, can be incorporated into AdvFS configurations.

4.4 UNIX File System

The UNIX File System (UFS) is a local file system. At one time, UFS was the principal file system, and it is still a vital alternative to the use of AdvFS.

UFS is compatible with the Berkeley 4.3 Tahoe release. UFS allows a pathname component to be 255 bytes, with the fully qualified pathname length restriction of 1023 bytes. The Tru64 UNIX implementation of UFS supports file sizes that exceed 2 GB.

UFS supports file block clustering, thereby producing sequential read and write access that is equivalent to the raw device speed of the mass storage device. UFS supports file-on-file mount for STREAMS (see Section 4.11).

4.5 Cluster File System

In a TruCluster Server cluster, the Cluster File System (CFS) is a virtual file system that sits above the physical file systems to provide clusterwide access to mounted file systems.

In general, the CFS makes all files visible to all cluster members and accessible by each member. Each cluster member has the same view; regardless of whether a file is stored on a device that is connected to all cluster members or on one that is private to a single member. By maintaining cache coherency across cluster members, CFS guarantees that all members at all times have the same view of file systems mounted in the cluster.

From the perspective of the CFS, each file system or AdvFS domain is served to the entire cluster by a single cluster member. Any cluster member can serve file systems on devices anywhere in the cluster. File systems mounted at cluster boot time are served by the first cluster member to have access to them. This means that file systems on devices on a bus private to one cluster member are served by that member.

For information about the Cluster File System, see the *Cluster Technical Overview*.

4.6 Network File System

The Network File System (NFS) is a facility for sharing files in a heterogeneous environment of processors, operating systems, and networks. NFS does so by mounting a remote file system or directory on a local system and then reading or writing the files as though they were local.

The Tru64 UNIX environment supports NFS Version 3 and NFS Version 2. The NFS Version 2 code is based on ONC Version 4.2, which is licensed from Sun Microsystems; the NFS Version 3 code is derived from prototype code from Sun Microsystems.

Because Tru64 UNIX supports both NFS Version 3 and Version 2, the NFS client and server bind at mount time using the highest NFS version number they both support. For example, a Tru64 UNIX client will use NFS Version 3 when it is served by an NFS server that supports NFS Version 3; however,

when it is served by an NFS server running only NFS Version 2, the NFS client will use NFS Version 2. For more detailed information on NFS Version 3, see the paper *NFS Version 3: Design and Implementation* (USENIX 1994).

In addition to the basic NFS services, Tru64 UNIX supports the following enhancements:

- NFS over TCP
- Write-gathering
- NFS-locking
- Automounting
- PC-NFS
- WebNFS

4.6.1 NFS Version 3 Features

NFS Version 3 supports all the features of NFS Version 2 and the following:

- Improved performance
  - Support for reliable asynchronous writes, which improves write performance over NFS Version 2 by a factor of seven, thereby reducing client response latency and server I/O loading
  - Support for a `READDIRPLUS` procedure that returns file handles and attributes with directory names to eliminate `LOOKUP` calls when scanning a directory
  - Support for servers to return metadata on all operations to reduce the number of subsequent `GETATTR` procedure calls
  - Support for weak cache consistency data to allow a client to manage its caches more effectively
- Improved security
  - Provides an `ACCESS` procedure that fixes the problems in NFS Version 2 with superuser permission mapping, and allows access checks at file-open time, so that the server can better support Access Control Lists (ACLs)
  - File names and pathnames specified as strings of variable length, with the maximum length negotiated between the client and server using the `PATHCONF` procedure
- Guaranteed exclusive creation of files

4.6.2 Enhancements to NFS

In addition to the NFS Version 3.0 functions, Tru64 UNIX features the following enhancements to NFS:

- NFS over TCP

  Although NFS has been traditionally run over the UDP protocol, Tru64 UNIX supports NFS over the TCP protocol. For more information, see the `mount(8)` reference page.

- Write-gathering

  On an NFS server, multiple synchronous write requests to the same file are combined to reduce the number of actual writes as much as possible. The data portions of successive writes are cached and a single metadata update is done that applies to all the writes. Replies are not sent to the client until all data and associated metadata are written to disk to ensure that write-gathering does not violate the NFS crash recovery design.

  As a result, write-gathering increases write throughput by up to 100 percent and the CPU overhead associated with writes is substantially reduced, further increasing server capacity.

- NFS-locking

  Using the `fcntl` system call to control access to file regions, NFS-locking allows you to place locks on file records over NFS protecting segments of a shared, NFS-served database. The status daemon, `rpc.statd`, monitors the NFS servers and maintains the NFS lock if the server goes down. When the NFS server comes back up, a reclaiming process allows the lock to be reattached.

- Automounting

  On an NFS client, the `automount` and `autofsd` daemons offer alternatives to mounting remote file systems with the `/etc/fstab` file, allowing you to mount the file systems on an as-needed basis.

  When a user on a system running one of these daemons invokes a command that must access a remotely mounted file or directory, the daemon mounts that file system or directory and keeps it mounted for as long as the user needs it. When a specified amount of time elapses (the default is 5 minutes) without the file system or directory being accessed, the daemon unmounts it.

  You specify the file systems to be mounted in map files, which you can customize to suit your environment. You can administer map files locally or through NIS, or through a combination of the two,

  Automounting NFS-mounted file systems provides the following advantages over static mounts:

– If NIS maps are used and file systems are moved to other servers, users do not need to do anything to access the moved files. Every time the file systems need to be mounted, the daemon will mount them from the correct locations.

– In the case of read-only files, if more than one NFS-server is serving a given file system, the daemon will connect you to the first server that responds. If at least one of the servers is available, the mount will not hang.

– By unmounting NFS-mounted file systems that have not been accessed for more than a certain interval (five minutes by default), the daemon conserves system resources, particularly memory.

The `autofsd` daemon has additional benefits over the `automount` daemon. It is more efficient because it requires less communication between the kernel and the user space daemon and it provides higher availability.

Although `autofsd` must be running for mounts and unmounts to be performed, if it is killed or becomes unavailable, existing auto-mounted NFS file systems continue to be available.

For more information about the `automount` and `autofsd` daemons, see the *Network Administration: Services* manual, the *Release Notes for Version 5.1A*, and the `automount`(8), `autofsd`(8), and `autofsmount`(8) reference pages.

- PC-NFS

  Compaq supports the PC-NFS server daemon, `pcnfsd`, which allows PC clients with PC-NFS configured to do the following:

  – Mount NFS file systems

    The PC-NFS `pcnfsd` daemon, in compliance with Versions 1.0 and 2.0 of the `pcnfsd` protocol, assigns UIDs and GIDs to PC clients so that they can talk to NFS.

    The `pcnfsd` daemon performs UNIX login-like password and user name verification on the server for the PC client. If the authentication succeeds, the `pcnfsd` daemon then grants the PC client the same permissions accorded to that user name. The PC client can mount NFS file systems by talking to the `mountd` daemon as long as the NFS file systems are exported to the PC client in the `/etc/exports` file on the server.

    Because there is no mechanism in Windows to perform file permission checking, the PC client calls the authentication server to check the user's credentials against the file's attributes. This happens when the PC client makes NFS requests to the server for file access that requires permission checking, such as opening a file.

– Access network printers

  The `pcnfsd` daemon authenticates the PC client and then spools and prints the file on behalf of the client.

- WebNFS

  WebNFS is an NFS protocol that allows clients to access files over the Internet in the same way that local files are accessed. WebNFS uses a public file handle that allows it to work across a firewall. This public file handle also reduces the amount of time required to initialize a connection. The public file handle is associated with a single directory (public) on the WebNFS server. For more information, see the `exports`(4), `exportfs`(2), and `nfs_intro`(4) reference pages.

4.7 CD–ROM File System

The Compact Disk Read-Only Memory File System (CDFS) is a local file system. Tru64 UNIX supports the ISO-9660 CDFS standard for data interchange between multiple vendors; the High Sierra Group standard for backward compatibility with earlier CD-ROM formats; and an implementation of the Rock Ridge Interchange Protocol (RRIP), Version 1.0, Revision 1.09.

The RRIP extends ISO-9660 system use areas to include multiple sessions, mixed-case and long file names; symbolic links; device nodes; deep directory structures; user IDs and group IDs and permissions on files; and POSIX timestamps.

All the code for CDFS has been taken from the public domain and enhanced for Tru64 UNIX.

Additionally, Tru64 UNIX supports the X/Open Preliminary Specification (1991) CD-ROM Support Component (XCDR). XCDR allows users to examine selected ISO-9660 attributes through defined utilities and shared libraries. XCDR also allows system administrators to substitute different file protections, owners, and file names for the default CD-ROM files. For more information, see the `cdfs`(4) reference page .

4.8 DVD File System

The Digital Versatile Disk (DVD) file system enables the reading of disks formatted in the Optical Storage Technology Association (OSTA) Universal Disk Format (UDF) specification. These interfaces conform to the ISO/ITEC 13346:1995 and ISO 9660:1988 standards.

User data sectors in a DVD–ROM can contain any type of data in any format. However, for Tru64 UNIX support through the DVDFS system, the OSTA UDF file format standard is mandatory. Additionally, DVD-ROM

standards require that the logical sector size and the logical block (the user data block) size be 2048 bytes.

See the dvdfs(4) reference page for more information.

4.9 Memory File System

The Memory File System (MFS) is essentially a UNIX File System that resides in memory. No permanent file structures or data are written to disk, so the contents of an MFS are lost on system reboots, unmountings, or power failures. Because it does not write data to disk, the MFS is a very fast file system — quite useful for storing temporary files or read-only files that are loaded into it after it is created.

For example, if you are building software that would have to be restarted if it failed, MFS is a good choice to use for storing the temporary files that are created during the build process, because the speed of MFS would reduce the build time. For more information about MFS, see the newfs(8) reference page.

4.10 The /proc File System

The /proc file system is a local file system that enables running processes to be accessed and manipulated as files by the system calls open, close, read, write, lseek, and ioctl.

The /proc file system is layered beneath the VFS; it is a pseudo file system that occupies no actual disk space. You can use the mount and unmount commands to manually mount and unmount the file system, or you can define an entry for it in the /etc/fstab file.

While the /proc file system is most useful for debuggers, it enables any process with the correct permissions to control another running process. Thus, a parent/child relationship does not have to exist between a debugger and the process being debugged. For more information, see the proc(4) reference page.

4.11 File-on-File Mounting File System

The File-on-File Mounting (FFM) file system allows regular, character, or block-special files to be mounted over regular files.

FFM is used, for the most part, by the Tru64 UNIX system calls fattach and fdetach of a STREAMS-based pipe (or FIFO). The two system calls are SVR4–compatible. By using FFM, a FIFO, which normally has no file system object associated with it, is given a name in the file system space. As a result, a process that is unrelated to the process that created the FIFO can then access the FIFO.

In addition to programs using FFM through the `fattach` system call, users can mount one regular file on top of another by using the `mount` command. Mounting a file on top of another file does not destroy the contents of the covered file; it simply associates the name of the covered file with the mounted file, thereby making the contents of the covered file temporarily unavailable. The covered file can be accessed only after the file mounted on top of it is unmounted.

Note that the contents of the covered file are still available to any process that had the file open at the time of the call to `fattach` or had the file open when a user issued a `mount` command that covered the file. For more information on FFM, see the `ffm`(4) reference page.

4.12 File Descriptor File System

The File Descriptor File System (FDFS) allows applications to reference a process' open file descriptors as if they were files in UFS. The association is accomplished by aliasing a process's open file descriptors to file objects. When FDFS is mounted, opening or creating a file descriptor file has the same effect as using the `dup` system call.

FDFS allows applications that were not written with support for UNIX style I/O to use pipes, named pipes, and I/O redirection. FDFS is not configured into the Tru64 UNIX system; it must be mounted by command or must be placed as an entry in the system's `/etc/fstab` file. For more information on FDFS, see the `fd`(4) reference page.

5

Kernel, Symmetric Multiprocessing, NUMA, Virtual Memory, and Device Support

This chapter discusses the Tru64 UNIX the following topics:

- The Tru64 UNIX kernel (Section 5.1)
- Symmetric multiprocessing (Section 5.2)
- Non-uniform memory access (NUMA) (Section 5.3)
- The virtual memory subsystem (Section 5.4)
- Tru64 UNIX support for various devices (Section 5.5)

5.1 The Tru64 UNIX Kernel

The kernel manages the Tru64 UNIX system resources. It can be adjusted for maximum performance by setting system attributes. Furthermore kernel tuning and debugging tools allow the examination of these attributes.

5.1.1 Kernel Tuning

Tru64 UNIX includes various subsystems that are used to define or extend the kernel. Kernel variables control subsystem behavior or track subsystem statistics after boot time.

Kernel variables are assigned default values at boot time. For certain configurations and workloads, especially memory or network-intensive systems, the default values of some attributes may not be appropriate, so you must modify these values to provide optimal performance.

Although you can use a debugger to directly change kernel variable values on a running kernel, Compaq recommends that you use kernel subsystem attributes to access the kernel variables. See `sys_ttrs`(5)

Subsystem attributes are managed by the configuration manager server, `cfgmgr`. You can display and modify attributes by using `sysconfig` and `sysconfigdb` commands and by using the Kernel Tuner, `dxkerneltuner`. In some cases, you can use the `sysconfig` to modify attributes while the system is running. You use `sysconfigdb` to make changes that will be preserved when the system is rebooted.

For more information, see the *System Configuration and Tuning* manual and the `sysconfig`(8) and `sysconfigdb`(8) reference pages.

5.1.2 Enhanced Kernel Debugging

The `dbx` debugger is a symbolic debugger that enables you to examine, modify, and display a kernel's variables and data structures.

With `dbx` you can perform the following tasks:

- Debug stripped images
- Examine memory contents
- Display the values of kernel variables and the value and format of kernel data structures
- Debug multiple threads
- Debug kernel core files using the `-k` option
- Perform breakpoint debugging of a running kernel across a serial line using the `-remote` option

A front-end to `dbx`, called `kdbx`, supports the entire suite of `dbx` commands, in addition to a C library API that allows programmers to write C programs to extract and format kernel data more easily than they can with just `dbx -k` or `dbx -remote`.

The `dbx` debugger is a command-line program. The `ladebug` debugger, an alternate debugger, provides both command-line and graphical user interfaces.

For more information, see the *Kernel Debugging* manual, the *Programmer's Guide*, the *Ladebug Debugger Manual*, and the `ladebug`(1), `dbx`(1), and `kdbx`(1) reference pages.

5.2 Symmetric Multiprocessing

Symmetric multiprocessing (SMP) is the ability of two or more processes (or multiple threads of a threaded application) to execute simultaneously on two or more CPUs. This concurrency of execution greatly improves performance. Additionally, it provides the opportunity to extend the life and increase the cost-effectiveness of multiprocessor systems by adding CPU cards (and their compute power) to multiprocessors rather than buying more systems.

Tru64 UNIX supports an implementation of SMP that is designed to optimize the performance of compute servers (systems dedicated to compute-bound, multithreaded applications) and data servers (file servers, DBMS servers, TP systems, and mail routers that serve a large number of network clients). The operating system also supports multithreaded application development

in an SMP environment. Note that SMP does not adversely affect using a multiprocessor as a timesharing system.

Tru64 UNIX SMP uses the following:

- Simple locks (also called spin locks, because they "spin" for a specified period of time waiting for held locks to be freed before timing out).
- Complex locks (read/write locks that can block waiting for a lock to be freed).
- Funneling, which is used in rare cases where locks would not be of benefit. Funneling forces a process to execute on a specific CPU; it is typically used for nonthread and multiprocessor-safe legacy programs.

Tru64 UNIX SMP achieves as much concurrency as possible by reducing the size of the system state that must be protected by locks, thereby reducing the necessity for locks and their overhead.

Tru64 UNIX, including its kernel, is fully parallel so that multiple processes or multiple threads can run simultaneously on multiple CPUs. The operating system uses concurrency and its locking strategy to ensure the integrity of the same kernel data structures. Multiple processes and multiple threads can access the same kernel data structures, but Tru64 UNIX makes sure that this access is performed in a logical order. Multiple processes and multiple threads cannot hold and request each others' locks, deadlocking the system. There are no architectural limits on the number of CPUs supported.

Tru64 UNIX SMP also supports processor binding, the ability to bind a particular process to a specified CPU, and load balancing, whereby the scheduler attempts to distribute all runable processes across all available CPUs. (Note that load balancing will not override processor binding.)

To improve performance, the scheduler also attempts to execute each process on the last CPU where it ran to take advantage of any state that may be left in that CPU's cache.

SMP is configurable and any of the following five modes can be configured at system boot time:

- Uniprocessing
- Optimized real-time preemption
- Optimized SMP
- Optimized real-time preemption and SMP
- Lock debug mode

When uniprocessing is set, only those locks required to support multiple threads are initialized into the kernel at system boot time.

When lock debug mode is set, the system:

- Checks the lock hierarchy and minimum system priority level (SPL)
- Stores debugging information by classes and maintains lock statistics
- Records the simple locks that are held by each CPU in CPU-specific arrays
- Records all of the complex locks that a thread is holding in the thread structure

You can use the `dbx` debugger to access this debugging information.

In addition, the development environment supports multithreaded application development. The `dbx`, `profile`, and `pixie` utilities support multiple threads, and the system includes thread-safe libraries.

For information on the Tru64 UNIX development environment and the threads package that Tru64 UNIX supports, see the *Programmer's Guide* and the *Guide to the POSIX Threads Library*. For information on configuring SMP, see the *System Administration* and *System Configuration and Tuning* manuals.

5.3 Non-Uniform Memory Access (NUMA)

Symmetric multiprocessor (SMP) systems typically provide one interconnect, either a bus or a switch, that links all system resources. This means that all CPUs in the system are subject to the same latency and bandwidth restrictions when accessing the system's memory and I/O channels. The drawback of the architecture of traditional SMP systems is that scaling the system to large numbers of CPUs causes the system bus to become a performance bottleneck.

One way to address this bottleneck is to build a system from SMP blocks (each with a limited number of CPUs, memory arrays, and I/O ports) and add a second-level bus or switch to connect the blocks. Non-uniform memory access (NUMA) is the term used to describe this type of system architecture, because it results in bandwidth and latency differences, depending on whether a particular CPU accesses memory and I/O resources locally (in the same building block where the CPU resides) or remotely (in another building block).

5.3.1 Hardware Requirements for NUMA Support

The GS80, GS160, and GS320 AlphaServer systems are the first Alpha implementations of cache-coherent NUMA (CC-NUMA) systems, meaning that the system hardware handles cache coherency between the system building blocks as well as within them. Therefore, software is relieved of

this responsibility. The operating system and user applications can treat a CC-NUMA system the same way they treat a traditional SMP system and still be programmatically correct.

5.3.2 Performance Implications of NUMA Support

Although software can treat a CC-NUMA system as a traditional SMP system and still be programmatically correct, obtaining optimal performance from a CC-NUMA system depends on appropriate use of its capabilities. In a network of systems, an application must sometimes run on a remote system rather than a local one. However, the application will almost always run faster on a local system that has all the resources needed by the application. This is because the connection between the systems increases response latency.

The same principal applies when you consider a CC-NUMA multiprocessor system as a network of building blocks, each of which contains a set of CPUs, memory arrays, and I/O ports. The CPUs in one system building block can access memory that is available locally (in their own block) or remotely (in another block). However, using the local memory is faster because memory access through the interblock switch increases response latency.

Starting with Tru64 UNIX Version 5.1, the operating system includes kernel algorithms, utilities, and programming APIs that are NUMA aware. These algorithms and user interfaces maximize the ratio of local to remote memory accesses and thereby help ensure optimal performance on CC-NUMA hardware.

In most software product documentation pertaining to NUMA support, references to a NUMA system assume a CC-NUMA hardware implementation. Therefore, this document also refers to CC-NUMA systems as NUMA systems.

5.3.3 Resource Affinity Domains

On Tru64 UNIX systems, the building blocks that make up a NUMA platform are mapped to structures called Resource Affinity Domains (RADs). A RAD identifies the set of CPUs, memory arrays, and I/O busses that, when used together, allow a system to work most efficiently.

On the GS80, GS160, or GS320 AlphaServer platform, the system can be made up of two, four, or eight Quad Building Blocks (QBBs). Each QBB contains a set of CPUs, memory, and I/O busses that have affinity for one another. Therefore, on these systems, the operating system maps each QBB to a RAD.

Single-processor platforms and multiprocessor platforms that do not use NUMA architecture contain one set of CPUs, memory, and I/O busses, all

with equal affinity for one another. Therefore, the operating system treats these systems as single-RAD systems.

5.3.4 Default NUMA-Aware Behavior of the Operating System

The following defaults are in place to increase the likelihood that NUMA system resources are used efficiently for most types of applications:

- The operating system defines a "home RAD" for each process and each of its threads. Default process or thread scheduling and memory allocation are done on the assigned home RAD whenever possible.

  In other words, the operating system attempts to schedule a process and all its threads on CPUs in the home RAD. Furthermore, the operating system attempts to allocate memory for application and kernel data on the home RAD. The cache affinity algorithms previously available only for traditional SMP systems are also used. Therefore, if a thread that previously ran on a particular CPU needs to be scheduled, the operating system attempts to schedule that thread on the same CPU.

  The operating system also defines a default overflow set of RADs. When there is insufficient free memory for application and kernel data on the home RAD, the operating system attempts to allocate memory from one or more remote RADs based on the default overflow set.

- For data that is globally accessed, the operating system attempts to replicate the data in or stripe it across all RADs where it might be accessed. More specifically, the operating system attempts to:
  - Replicate kernel code and kernel read-only data on all RADs at boot time
  - Replicate other kinds of read-only data, such as shared program and library code, on all RADs where a running process or thread needs to access it

    If there is insufficient free memory on the RAD where the process or thread is running to replicate shared, read-only data, the operating system will use a copy on a remote RAD rather than wait for free memory on the local RAD to make the copy.
  - Stripe System V shared memory across all RADs

    Striping minimizes the likelihood that certain processes and threads always access System V shared memory locally while others always access it remotely. Striping also spreads the load across multiple RADs.

- The operating system attempts to balance the load on each RAD so that local CPU cycles and local memory pages are both available to the processes running on the RAD.

Local availability of memory and CPU cycles influences RAD selection at the time a process is created. The same factors might cause the operating system to migrate a process and associated memory pages from one RAD to another in response to changing resource requirements and access patterns.

5.3.5 NUMA Options in System Utilities

Since Version 5.1, the `runon` command includes the `-r` option, which can be used to run a user program or shell in a particular RAD. See `runon`(1) for more information.

5.3.6 NUMA Application Programming Interfaces

When a mix of applications with differing resource needs are run on the same system, it is best for user applications to rely on the default behavior of operating system software. However, large and highly specialized user applications might realize additional performance advantages through direct use of NUMA APIs. For example:

- An application for which I/O requests are extremely large might realize significant performance advantages when the CPU cycles and memory pages associated with an I/O request are striped across all available RADs. This optimization strategy works only if the data being read from or written to disk is also striped across controllers that are attached to the I/O ports of different RADs. (If I/O ports on different RADs channel data into the same RAID controllers, device latency will likely offset the bandwidth increase for CPU cycles and memory.)
- Applications with many subprocesses or threads that operate on large but different subsets of the same data might benefit from explicit resource management. In this case, NUMA APIs can help to increase the ratio of local to remote accesses by changing the default algorithms for replicating or striping program data and System V shared memory.

NUMA APIs are included in the following libraries:

- The NUMA Library (`libnuma`)
- The Standard C Library (`libc`)

  Certain routines required for NUMA-aware programming are included in the `libc` library because they perform operations that are also useful in more generic types of programs.
- The POSIX Threads Library (`libpthread`)

  NUMA routines that are useful only in multithreaded programs are included in the `libpthread` library.

See the following reference pages for more information about the NUMA APIs:

- `numa_intro(3)` lists and describes the NUMA functions.
- `numa_types(4)` describes the data types, structures, and macros used with the NUMA functions.
- `numa_scheduling_groups(4)` describes a NUMA scheduling group.

For a more detailed discussion of NUMA support, plus a complete program example, see the *NUMA Overview*, a Web-only document included in the online documentation sets for Tru64 UNIX Version 5.1 or higher at the following site:

http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html

5.4 Virtual Memory

The virtual memory subsystem performs the following functions:

- Allocates memory to processes
- Tracks and manages all the pages in the system
- Uses paging and swapping to ensure that there is enough memory for processes to run and to cache file system I/O

The total amount of physical memory is determined by the capacity of the memory boards installed in your system. The system distributes this memory in 8 KB units called pages. The system distributes pages of physical memory among three areas:

- Wired memory

  Memory is wired statically at boot time and dynamically at run time.

  At boot time, the operating system and the Privileged Architecture Library (PAL) code wire a contiguous portion of physical memory to perform basic system operations. Static wired memory is reserved for operating system data and text, system tables, the metadata buffer cache, which temporarily holds recently accessed UNIX File System (UFS) and CD-ROM File System (CDFS) metadata, and the Advanced File System (AdvFS) buffer cache. Static wired memory cannot be reclaimed through paging.

- Virtual memory

  The virtual memory subsystem uses a portion of physical memory to cache processes' most recently accessed anonymous memory (modifiable virtual address space) and file-backed memory. The subsystem allocates memory to competing processes and tracks the distribution of all the

physical pages. This memory can be reclaimed through paging and swapping.

- Unified Buffer Cache

  The Unified Buffer Cache (UBC) uses a portion of physical memory to cache most recently accessed file system data. The UBC contains actual file data for reads and writes and for page faults from mapped file regions and also AdvFS metadata. By functioning as a layer between the operating system and the storage subsystem, the UBC can decrease the number of disk operations. This memory can be reclaimed through paging.

The virtual memory subsystem and the UBC compete for the physical pages that are not wired. Pages are allocated to processes and to the UBC, as needed. When the demand for memory increases, the oldest (least recently used) pages are reclaimed from the virtual memory subsystem and the UBC, are moved to swap space, and are then reused. Various attributes control the amount of memory available to the virtual memory subsystem and the UBC and the rate of page reclamation.

5.4.1 Managing and Tracking Pages

The virtual memory subsystem allocates physical pages to processes and the UBC, as needed. Because physical memory is limited, these pages must be periodically reclaimed so that they can be reused.

The virtual memory subsystem uses page lists to track the location and age of all the physical memory pages. At any one time, each physical page can be found on one of the following lists:

- Wired list

  Pages that are wired and cannot be reclaimed.

- Free list

  Pages that are clean and are not being used. The size of this list controls when page reclamation occurs.

- Active list

  Pages that are being used by the virtual memory subsystem or the UBC.

  To determine which pages should be reclaimed first, the page stealer daemon identifies the oldest pages on the active list and designates the least recently used pages as follows:

  – Inactive pages are the oldest pages that are being used by the virtual memory subsystem.

  – UBC least recently used pages are the oldest pages that are being used by the UBC.

Tru64 UNIX virtual memory is NUMA aware. It maintains a separate set of lists and worker threads per RAD.

5.4.2 Prewriting Modified Pages

The virtual memory subsystem attempts to keep memory pages clean to ease the recovery of memory shortages. When the virtual memory subsystem anticipates that the pages on the free list will soon be depleted, it prewrites to swap space the oldest modified (dirty) inactive pages. In addition, when the number of modified UBC least recently used pages exceeds 10 percent of the total UBC least recently used pages, the virtual memory subsystem prewrites to swap space the oldest modified UBC least recently used pages.

5.4.3 Using Attributes to Control Paging and Swapping

When the demand for memory significantly depletes the free list, paging begins. The virtual memory subsystem takes the oldest inactive pages and UBC least recently used pages, moves the contents of the modified pages to swap space, and puts the clean pages on the free list, where they can be reused.

If the free page list cannot be replenished by reclaiming individual pages, swapping begins. Swapping temporarily suspends processes and moves entire resident sets to swap space, which frees large amounts of physical memory.

The point at which paging and swapping start and stop depends on the values of various tunable virtual memory subsystem kernel attributes.

Because the UBC competes with the virtual memory subsystem for the physical pages that are not wired by the kernel, the allocation of memory to the UBC can affect file system performance and paging and swapping activity. The UBC is dynamic and consumes varying amounts of memory while responding to changing file system demands.

By default, the UBC can consume up to 100 percent of memory. However, part of the memory allocated to the UBC is only borrowed from the virtual memory subsystem. When paging starts, borrowed UBC pages are the first to be reclaimed. The amount of memory allocated to the UBC can be controlled by various virtual memory subsystem kernel attributes.

5.4.4 Paging Operation

When the memory demand is high and the number of pages on the free page list falls below the paging threshold, the virtual memory subsystem uses paging to replenish the free page list. The page reclamation code controls

paging and swapping. The page out daemon and task swapper daemon are extensions of the page reclamation code.

The page reclamation code activates the page stealer daemon, which first reclaims the pages that the UBC has borrowed from the virtual memory subsystem, until the size of the UBC reaches the borrowing threshold. (The default is 20 percent.) If the reclaimed pages are dirty (modified), their contents must be written to disk before the pages can be moved to the free page list. Freeing borrowed UBC pages is a fast way to reclaim pages, because UBC pages are usually unmodified.

If freeing UBC borrowed memory does not sufficiently replenish the free list, a page out occurs. The page stealer daemon reclaims the oldest inactive pages and UBC least recently used pages.

Paging becomes increasingly aggressive if the number of free pages continues to decrease. If the number of pages on the free page list falls below 20 pages (the default), a page must be reclaimed for each page taken from the list. To prevent deadlocks, when the number of pages on the free page list falls below 10 pages (the default), only privileged tasks can get memory until the free page list is replenished. Both these limits are controlled by tunable attributes.

Page out stops when the number of pages on the free list rises above the paging threshold. If paging individual pages does not sufficiently replenish the free list, swapping is used to free a large amount of memory.

5.4.5 Swapping Operation

If there is a high demand for memory, the virtual memory subsystem may be unable to replenish the free list by reclaiming pages. Swapping reduces the demand for physical memory by suspending processes, which dramatically increases the number of pages on the free list. To swap out a process, the task swapper suspends the process, writes its resident set to swap space, and moves the clean pages to the free list. Swapping can have a serious impact on system performance.

Idle task swapping begins when the number of pages on the free list falls below the swapping threshold (the default is 74 pages) for a period of time. The task swapper then suspends all tasks that have been idle for 30 seconds or more.

If the number of pages on the free list continues to decrease, hard swapping begins. The task swapper suspends, one at a time, the tasks with the lowest priority and the largest resident set size.

Swapping of an individual task stops when the number of pages on the free list reaches the high water swapping threshold. (The default is 1280.)

A swap in occurs when the number of pages on the free list has been sufficiently replenished for a period of time. The task's working set is paged in from swap space and it can now execute. By default, a task must remain in the swapped-in state for one second before it can be swapped out.

Increasing the rate of swapping (swapping earlier during page reclamation) increases throughput. As more processes are swapped out, fewer processes are actually executing and more work is done. Although increasing the rate of swapping moves long-sleeping threads out of memory and frees memory, it degrades interactive response time. Swapped out processes have a long latency.

Decreasing the rate of swapping (swapping later during page reclamation), improves interactive response time, but at the cost of throughput.

5.4.6 Swap Space Allocation Mode

You can use two modes to allocate swap space. The modes differ in how the virtual memory subsystem reserves swap space for anonymous memory (modifiable virtual address space). Anonymous memory is memory that is not backed by a file, but is backed by swap space (for example, stack space, heap space, and memory allocated by the `malloc` function). Neither mode has a performance benefit attached to it:

- Immediate mode

  This mode reserves swap space when a process first allocates anonymous memory. Immediate mode is the default swap space allocation mode and is also called eager mode.

  This mode may cause the system to reserve an unnecessarily large amount of swap space for processes. However, it ensures that swap space will be available to processes if it is needed.

- Deferred mode

  This mode reserves swap space only if the virtual memory subsystem needs to write a modified virtual page to swap space. It postpones the reservation of swap space for anonymous memory until it is actually needed. Deferred mode is also called lazy mode.

  This mode requires less swap space than immediate mode and may cause the system to run faster, because it requires less swap space bookkeeping. However, because deferred mode does not reserve swap space in advance, the swap space may not be available when a process needs it, and processes may be killed asynchronously.

In addition, you can override the system-wide swap space allocation mode for a specific command or application by using the `swapon` command. For more information, see the `swapon(8)` reference page.

5.4.7 Using Swap Buffers

To facilitate the movement of data between memory and disk, the virtual memory subsystem uses synchronous and asynchronous swap buffers. The virtual memory subsystem uses these two types of buffers to immediately satisfy a page in request without having to wait for the completion of a page out request, which is a relatively slow process.

Synchronous swap buffers are used for page-in page faults and for swap outs. Asynchronous swap buffers are used for asynchronous page outs and for prewriting modified pages.

5.4.8 Unified Buffer Cache

The Unified Buffer Cache (UBC) is a Tru64 UNIX virtual memory feature. The UBC uses a portion of the machine's physical memory to cache the most recently accessed file system data. The UBC contains actual file data, which includes reads and writes from conventional file activity, page faults from mapped file sections, and AdvFS metadata.

The UBC shares (contends for) physical memory pages with the virtual memory subsystem, but not pages that are wired by the kernel. The UBC is dynamic, consuming varying amounts of memory in response to changes in file system demands for its service. For information about the UBC, see the *System Configuration and Tuning* manual.

5.5 Device Support

The kernel supports hot-swap I/O devices, which provides the capability to automatically fault-in a device driver when a hot-swappable I/O device comes on line.

When the hardware code detects a new device and determines that the device driver is not present in the kernel, it can make a kernel function call to automatically load the device's driver into the kernel. Additionally, hot-swapping provides the flexibility of not having to prebuild the kernel subsystem or driver into the kernel. Instead, it can be faulted in when the device is first accessed.

6

Development Environment

This chapter highlights the major features provided by the development environment for Tru64 UNIX. The following topics are discussed:

- The Compaq C compiler, which is standards compliant and fully optimized that produces extremely efficient code to fully exploit the 64-bit address space of the Alpha architecture (Section 6.1)
- Debuggers that support C, Assembler, Fortran (f77 and f90), C++, Pascal, and Ada (Section 6.2)
- The ATOM, `gprof`, `prof`, and `uprofile` profiling tools (Section 6.3)
- Shared libraries (Section 6.4)
- Run-time libraries (Section 6.5)
- The Java development kit (JDK) (Section 6.6)
- Development commands (Section 6.7)
- Support for POSIX Threads, Visual Threads, and Thread-Independent Services (Section 6.8)
- The Berkeley Memory-Mapped File Support (mmap) function (Section 6.9)
- The real-time user and programming environment (Section 6.10)
- The network programming environment (Section 6.11)

In addition, Tru64 UNIX supports internationalization, standard UNIX development tools (for example, `awk`, `lint`, `make`, and `prof`), and run-time libraries for Ada, C++, Cobol, Fortran, and Pascal programs.

For more more information, see the *Programmer's Guide*, the *Programming Support Tools* manual, the *Assembly Language Programmer's Guide*, and *Writing Software for the International Market*.

6.1 Compaq C Compiler

The Compaq C compiler for Tru64 UNIX is a standards-compliant, full-featured, highly-optimizing compiler that was specifically developed to exploit the Alpha architecture. In particular, the Compaq C compiler recognizes these popular C dialects:

- ANSI and ISO C with extensions (`-std`, the default)

  Strict ANSI and ISO C (`-std1`, `-isoc94`)
- Traditional K&R C (`-std0`)
- Microsoft C (`-ms`)
- VAX C (`-vaxc`)

The Compaq C compiler complies with the following standards:

- ANSI and ISO C (`cc -std1`)
- XPG4-UNIX (`c89 -D_XOPEN_SOURCE_EXTENDED`)

For more information on standards compliance, see the `standards`(5) reference page.

The Compaq C compiler supports language extensions including:

- OpenMP parallel decomposition directives (`-omp`, `-check_omp`, `-mp`)
- Microsoft C structured exception handling (`try-except` and `try-finally`) and thread-local storage
- 32-bit pointers (64-bit pointers are the default) to reduce memory consumption and facilitate porting (`-taso`, `-xtaso`, `-xtaso_short`. (See the `protect_headers_setup`(8) reference page for more information.)
- User-defined assembly language sequences using `#pragmas` and intrinsic functions that expand and optimize these instruction sequences in line
- `#pragmas` and command line options for controlling data alignment (`-assume`, `-misalign`, `-Zp`, `-member_alignment`)

The Compaq C compiler supports enhancements for mathematical computing such as:

- IEEE floating point (including proper handling for exceptional conditions like NaN, INF, and so forth)
- Fast math mode (INF, NaN, and so forth, translated to avoid exception handling)
- Quad-precision (128-bit) floating point representation for `long double`

For more information on the Compaq C compiler for Tru64 UNIX, see the `cc`(1) reference page.

6.2 Debuggers

Tru64 UNIX supports the following source code debuggers:

- `dbx`

- ladebug

6.2.1 The dbx Debugger

The dbx debugger is used to debug programs at the source-code level or the machine-code level. The interface to dbx is a command line interface. The tasks you can perform with dbx include the following:

- Debug programs written in C, Fortran (f77 and f90), assembly language, or Pascal.
- Debug active kernels, core dumps, and programs that use multiple threads (by using /proc to attach to running processes), and shared libraries.
- Analyze kernel core dumps.
- Patch the on-disk copy of either user programs or the kernel.
- Perform multiprocess debugging and debugging across fork and exec calls.

6.2.2 The ladebug Debugger

The ladebug debugger is a source-level symbolic debugger that has both a graphical user interface (GUI) and a command line interface similar to the dbx command line interface. The tasks you can perform with ladebug include the following:

- Attach to and detaching from running processes.
- Detect problems and debug across fork and exec calls.
- Debug multiple processes.
- Debug multithreaded programs, either POSIX Threads (formerly called DECthreads) applications or kernel modules that make use of kernel threads.
- Debug programs written in C++, C, Fortran 77, Fortran 90, Ada, Cobol, and Assembler.

  The ladebug debugger is a full C++ debugger, which demangles C++ names, understands C++ expressions, provides support for inline functions, templates, and C++ exceptions. Its support for f77 and f90 includes case insensitivity, common blocks, alternate entry points, language-dependent type printing, and assume shape arrays.
- Debug machine level code.
- Debug running programs or core dumps.
- Debug shared objects.

- Catch unaligned access problems.
- Debug active kernels, either locally or remotely, and analyzing kernel crash dumps.
- Evaluate expressions using the syntax of the source programming language.
- Remotely debug programs running on different target machines by way of the remote debugging server.

  The `ladebug` remote debugging protocol is also available along with the C source code for a sample remote debugging server that adheres to the protocol.

Internationalization support is available in a separate kit. The internationalized `ladebug` debugger accepts multibyte characters as input, and outputs local language characters according to the current global locale set in the debugger. It also supports the `wchar_t` datatype in C and C++.

6.3 Profiling Tools

Tru64 UNIX provides the following profiling toolkit:

- ATOM

  Provides a flexible code instrumentation interface that is capable of building a wide variety of user-defined program analysis tools. The toolkit comprises an instrumentation control utility and a library whose procedural interface helps you to easily develop special-purpose instrumentation and analysis tools.

  Tru64 UNIX provides the following ATOM-based instrumentation and analysis tools:

  – `hiprof`

    A call-graph profiling tool with output that can be postprocessed by `gprof`.

  – `third` (Third degree)

    A tool to find memory leaks and check for incorrect memory accesses.

  – `pixie`

    A superset of the existing `pixie` basic block profiler, which can profile a program's executable files and its shared libraries. The output of `pixie` can be analyzed by `prof`.

Tru64 UNIX also supports the following profiling tools:

- `gprof`

For programs compiled with the `-pg` option, `gprof` displays the number of calls that named procedures have made to each other and how much CPU time each procedure consumed. It does this by using PC sampling statistics. It also analyzes the output of programs instrumented with `hiprof`.

- `prof`

  For programs compiled with the `-p` option, `prof`displays how much CPU time was consumed by each procedure in a program and its shared libraries. It does this using PC-sampling statistics. It also analyzes the output of programs instrumented with `pixie` or monitored with `uprofile` and `kprofile`.

- `uprofile` and `kprofile`

  These use the Alpha chip's built-in performance counters to sample a variety of events in the CPU during the execution of the kernel or an application program. These tools report on CPU cycles, memory and cache effects, and so forth, which `prof` can then analyze.

For more information on profiling tools, see the *Programmer's Guide* and the appropriate reference pages.

6.4 Shared Libraries

Tru64 UNIX provides a full complement of dynamic shared libraries, compatible with System V semantics for shared library loading and symbol resolution, and the System V API for dynamic loading (`dlopen`, `dlclose`, `dlsym`, and `dlerror`).

Because they allow programs to include only information about how to load and access routines rather than the routines themselves, shared libraries increase system performance, reduce disk and memory requirements, and simplify system management.

The shared libraries are located in the `/usr/shlib` directory.

The Tru64 UNIX implementation of X11R6.3 and Motif also makes use of both static and shared libraries. The X11 shared libraries are located in the `/usr/shlib/X11` directory.

6.4.1 Quickstart

Quickstart allows shared libraries with unique addresses to start faster than if their addresses were in conflict. Each shared library must have a unique address placed in the `/usr/shlib/so_locations` file, which allows applications that link against these shared libraries to start execution faster. This happens because the shared objects do not have to be relocated

at run time. The `ld` utility can read and write an `so_locations` file when it creates a shared library.

6.4.2 Dynamic Loader

Tru64 UNIX uses a System V Release 4.0 compatible loader to load shared libraries dynamically. This loader provides the following features:

- The ability to call into dynamically loaded shared libraries
- System V Release 4.0 symbol resolution semantics, including symbol preemption
- The ability to prelink libraries for fast program loading

6.4.3 Versioning

Tru64 UNIX supports full and partial duplication of shared libraries. The loader looks for backward-compatible versions of shared libraries using a path constructed by appending the version string as a subdirectory of the normal search path. As a result, any changes to kernel interfaces or to global data definitions that would ordinarily break binary compatibility will not affect your applications, because you can maintain multiple versions of any shared library and link your application against the appropriate version of that shared library.

In Motif Version 1.2, for example, the OSF changed several of the interfaces, thereby breaking binary compatibility with applications built against Motif 1.1.3 libraries. To preserve binary compatibility, Tru64 UNIX supports both Motif 1.1.3 and Motif 1.2 shared libraries, so that applications can access the Motif 1.1.3 shared libraries. For more information on versioning, see the *Programmer's Guide*.

6.5 Run-Time Libraries

Tru64 UNIX supports the following run-time libraries. With these libraries, you can run previously compiled programs without having to install the corresponding language programs on your system.

- COMPAQ C++ run-time-libraries (`libcxx`, `libcomplex`, and `libtask`)

  These libraries support such DEC C++ run-time functions as I/O handling, complex arithmetic, and multitasking.

- Compaq COBOL run-time libraries (`libcob`, `libots2`, and `libisamstub`)

  These libraries support such COBOL run-time functions as I/O handling, decimal arithmetic, COBOL ACCEPT and DISPLAY statements,

STRING and UNSTRING operations, and CALL and CANCEL statements.

- DEC Fortran run-time libraries (`libfor`, `libfutil`, and `libUfor`)

  These libraries support such Fortran run-time functions as I/O handling, intrinsic functions, data formatting, data conversion, math functions, and Fortran bindings to common operating system services.

- DEC Pascal run-time library (`libpas`)

  This library supports such Pascal run-time functions as I/O handling, math functions, time and date services, and file services.

6.6 Java Development Kit

The Java Development Kit (JDK) is a component of Tru64 UNIX. You can use this kit to develop and run Java applets and programs.

The JDK for Tru64 UNIX contains a just-in-time compiler (JIT), which provides on-the-fly compilation of your application's Java byte-code and run-time calls into native Alpha machine code. This results in significantly faster execution of your Java application compared with running it using the Java interpreter. The JIT runs by default when you enter the `java` command.

The JDK implements Java threads on top of native (POSIX) threads. This allows different Java threads in your application to run on different processors, provided that you have a multiprocessor machine. It also means that your Java application will run properly when linked with native methods or native APIs (such as DCE) that are also implemented using POSIX Threads.

For more information, see the Java documentation in the following directory on your Tru64 UNIX system where the JDK is installed: `/usr/share/doclib/java/index.html`

6.7 Development Commands

Tru64 UNIX supports the full array of development tools, including `ar`, `as`, `btou`, `cb`, `cc`, `cflow`, `cpp`, `ctags`, `cxref`, `c89`, `dbx`, `dis, error`, `file`, `indent`, `ld`, `lex`, `lint`, `loader`, `m4`, `make`, `mig`, `mkstr`, `nm`, `odump`, `pixie`, `ppu`, `prof`, `ranlib`, `size`, `stdump`, `strings`, `strip`, `tsort`, `xstr`, and `yacc`, as well as the source code control systems `rcs` and `sccs`.

Many of the development commands are specified by the System VT POSIX, XPG4, and XPG4-UNIX standards to which Tru64 UNIX is fully compliant. Tru64 UNIX supports both the OSF version of the `make` command and the ULTRIX version of `make` command. The ULTRIX `make` command is POSIX 1003.2 compliant and more robust than the OSF `make` command.

6.8 Thread Support

Tru64 UNIX offers the POSIX Threads Library (formerly DECthreads), Visual Threads, and Thread-Independent Services.

6.8.1 POSIX Threads Library

The POSIX Threads Library is an implementation of the POSIX 1003.1c-1995 standard multithreading API. POSIX Threads provides efficient two-level scheduling (POSIX "process contention scope") that is tightly integrated with the kernel to automatically maintain the maximum level of computational and I/O concurrency at all times. For applications that require real-time scheduling with respect to hardware events or threads in other processes or in the kernel, POSIX Threads also provides "system contention scope" and scheduling.

6.8.2 Visual Threads

Visual Threads is a diagnistic tool for analyzing and debugging multithreaded applications. You can Visual Threads to automatically diagnose common problems associated with multithreading; these problems include deadlock, protection of shared data, and thread-usage errors.

You can also use this tool to monitor the thread-related performance of the application; it will help you identify bottlenecks or locking granularity problems. It is a unique debugging tool because it can be used to identify problem areas even if an application does not show any specific problem symptoms.

Visual Threads can be used with any Tru64 UNIX application that uses POSIX Threads (DECthreads) or is written in Java. It is designed for multithreaded applications of all sizes; it can handle applications with from two threads to hundreds of threads.

Visual Threads is licensed as part of the Developers' Toolkit.

6.8.3 Thread-Independent Services

Tru64 UNIX supports Thread-Independent Services (TIS) routines, which are provided to enable application writers to write thread-safe code for nonthreaded libraries and applications. In the presence of threads, these routines provide thread-safe functionality. In the absence of threads, these routines impose the minimum possible overhead on their caller. Note that the TIS routines are used by the C run-time library to provide support for both single and multithreaded applications.

6.9 Memory-Mapped File Support

Tru64 UNIX supports the Berkeley Memory-Mapped File Support (mmap) function, which allows an application to access data files with memory operations rather than file I/O operations.

6.10 Real-Time User and Programming Environment

Tru64 UNIX supports a real-time user and programming environment; it is shipped as an optional subset. The Tru64 UNIX real-time programming environment conforms to the POSIX 1003.1b-1993 standard for real time, which allows you to develop and run portable real-time applications in a POSIX environment. The POSIX 1003.1b interfaces are collected in the real-time and asynchronous I/O libraries `librt` and `libaio`, respectively.

If you enable kernel preemption, a higher priority process can preempt a lower-priority process regardless of whether it is running in kernel mode or user mode. With this fully preemptive kernel, the Process Preemption Latency (the amount of time it takes to preempt a lower-priority process) is minimized.

In addition to a preemptive kernel, the Tru64 UNIX real-time programming environment supports the following POSIX 1003.1b features:

- Real-time clocks and timers
- Real-time queued signals
- Fixed priority scheduling policies
- Real-time scheduler priority levels
- Counting semaphores
- Shared memory
- Process memory locking
- Asynchronous I/O
- Synchronized I/O
- Message-passing interfaces
- Thread-safe implementation of real-time libraries

For more information on the real-time programming environment, see the *Guide to Realtime Programming*. For information on configuring the real-time kernel, see the *System Administration* manual.

6.11 Network Programming Interfaces

The network programming environment includes the programming interfaces for application, kernel, and driver developers writing network applications and implementing network protocols. Additionally, it includes the kernel-level resources that an application requires to process and transmit data, some of which include libraries, data structures, header files, and transport protocols.

The following programming interfaces are supported by the operating system:

- X/Open Transport Interface (XTI/TLI)
- BSD Sockets
- System V Release 4.0 STREAMS
- Data Link Interface (DLI)
- Data Link Provider Interface (DLPI)
- Extensible SNMP (eSNMP)

For more detailed information on the network programming environment, see the *Network Programmer's Guide*.

7

UNIX and Windows NT Interoperability

This chapter describes the Tru64 UNIX applications that will help you achieve seamless Windows NT interaction with your UNIX system. The first section provides a brief interoperability overview (Section 7.1), after which the following topics are discussed:

- The Advanced Server for UNIX (Section 7.2)
- The Windows 2000 Single Sign-On (SSO) application, which helps Windows 2000 users to log in to a Tru64 UNIX system (Section 7.3)
- The INTERSOLV DataDirect software products that enable ODBC and JDBC connectivity for applications (Section 7.4)
- The Component Object Model (COM) middleware (Section 7.5)

7.1 Overview

Most corporations today own a mix of technology that has been purchased over a number of years. Some installations are based largely on UNIX and are just beginning to include Microsoft's Windows NT. Others are Windows-based and are just beginning to incorporate UNIX.

Compaq's integration solution begins with your current assets — software, data, and skills. Building on current technology and user knowledge base, Compaq speeds integration between UNIX and Windows NT, thereby increasing their combined contribution to the enterprise's productivity.

7.2 Advanced Server for UNIX

The Advanced Server for UNIX (ASU) software is a Tru64 UNIX layered application that integrates Tru64 UNIX and Windows environments. The ASU software implements Windows NT Server Version 4.0 services, security, and functionality on a system running the Tru64 UNIX operating system software. The Tru64 UNIX system on which the ASU software is running appears as a Windows NT Server to other Windows systems and to users of Windows systems, and can participate in a Windows NT and Windows 2000 domain.

You use native Windows commands and utilities to manage the ASU software and to make UNIX based file systems and printers available to Windows users as shares. Windows users connect to shares without

modification to their software. Once connected, the Tru64 UNIX directory or printer associated with a share appears as a transparent extension to a Windows user's local computing environment.

The ASU software is packaged with the Tru64 UNIX media kit and provides two free connects. Contact your local Compaq office or your Compaq authorized reseller for information about Compaq's licensing terms and policies or purchasing an ASU license.

7.3 Windows 2000 Single Sign-On

The Windows 2000 Single Sign-On (SSO) application gives Windows 2000 users the capability of logging into Tru64 UNIX system using their Windows 2000 user name and password. With SSO, system administrators can create a single user account from which a user can request access to resources on a Tru64 UNIX system or a Windows 2000 system in an Intranet environment.

Some Windows 2000 Single Sign-On software is installed on a Windows 2000 system and some on a Tru64 UNIX system. The software that you install on the Tru64 UNIX system creates a Security Integration Architecture (SIA) module that directs user authentication requests to the Windows 2000 Active Directory. The Tru64 UNIX system must not be using C2 security.

The software that you install on the Windows 2000 system extends the Active Directory to include Tru64 UNIX user account and group attributes such as user login name, User ID (UID), Group ID (GID), a comment, a path to a home directory, and a login shell. See the *System Administration* manual for more information on Tru64 UNIX user account and group attributes.

Secure authentication between the Tru64 UNIX system and the Active Directory service occurs using MIT Kerberos Version 5 client software, provided with the Windows 2000 Single Sign-On kit. UNIX user account information can be stored in the Active Directory, to give system administrators a single user account directory spanning Tru64 UNIX and Windows 2000.

Administrators can also manage the additional Tru64 UNIX parameters using the Microsoft Management Console (MMC) snap-in extensions provided with the SSO kit.

7.4 Data Access (ODBC and JDBC)

Tru64 UNIX provides the family of INTERSOLV DataDirect software products to enable ODBC and JDBC connectivity for your applications. This is optional software for use in developing and deploying applications and is licensed as part of the Tru64 UNIX operating system license.

SequeLink ODBC Edition is a universal ODBC client component. DataDirect SequeLink ODBC provides transparent connectivity to almost any type of client, network, server, or database.

For developers working with Java, JDBC provides Java applications to access data sources and databases across platforms. The SequeLink Java Edition is a universal standards-based implementation of JDBC. It is also flexible, providing scalable connectivity from multivendor client, server, and Web environments to industry-leading databases. It is optimized and tuned for the Java environment, extending the functionality and performance of existing systems and easily incorporating new technologies.

7.5 COM for Tru64 UNIX

COM, the Component Object Model, is middleware developed by Microsoft for the Windows platform. COM implements a binary standard that allows two or more applications to work together regardless of whether they were written by different vendors, in different languages, at different times, or on different platforms running different operating systems. DCOM, the Distributed Component Object Model, extends the COM model and provides applications with a way to interact remotely over a network.

COM for Tru64 UNIX implements Microsoft COM and DCOM, as well as the required underlying Windows capabilities for the Compaq Tru64 UNIX platform.

The Compaq implementation provides all the basic functions, libraries, and tools that a COM application in a heterogeneous Windows NT client and Tru64 UNIX server environment requires. COM for Tru64 UNIX supports the creation of server applications in C and C++ and supports client applications in C, C++, Java, and Visual Basic.

Programmers who develop exclusively in Windows NT environments will find the same COM Application Programming Interface (API) and the same behavior in a heterogeneous Windows NT client and Tru64 server environment.

COM for Tru64 UNIX provides traditional COM and DCOM capabilities for your application. These capabilities conform to the Microsoft ActiveX Core Technology specification. They include the following:

- MIDL, the Microsoft Interface Definition Language Compiler that you use to create the component object interface.
- The interfaces and APIs defined by Microsoft as those needed to support COM on non-Windows platforms.
- Support for COM capabilities such as Monikers, OLE Automation, Uniform Data Transfer (UDT), Connectable Objects, surrogate processes

(including custom surrogate processes), structured storage, and type libraries.

- Single-Threaded Apartment (STA) and Multi-Threaded Apartment (MTA) threading models.
- Remote Procedure Call System Services (RPCSS), which include Service Control Manager (SCM), Object Exporter, and Running Object Table.
- Registry, the database of COM components and relevant configuration information, and Registry tools, such as `sermon`, and `regsvr`, that allow you to modify Registry contents.
- Security in the form of call security that allows a client or a server to apply an appropriate security level to method calls and Security Support Provider Interface (SSPI) standard that defines security providers that can be accessible to DCOM applications. Microsoft uses the Windows NT Distributed Security Support Provider Interface (also called the NTLM-SSP). COM for Tru64 UNIX supports "pass-through" NTLM SSP calls for user authentication and 40-bit encryption.
- Internationalization capability, including Unicode support of wide characters. COM for Tru64 UNIX converts platform-specific differences between the 64-bit implementation, which uses 4-byte wide characters, and other implementations, which use 2-byte wide characters.
- Error-handling convention that allows COM objects in different environments to share status information.
- Microsoft Remote Procedure Call (MS RPC), which provides transparent communication so that remote clients appear to directly communicate with the server. MS RPC is wire-level compatible (as opposed to call-level compatible) with the Open Software Foundation (OSF) implementation of Distributed Computing Environment (DCE RPC).
- Localized Messaging, which supports localization of system messages in COM libraries and utility applications.
- Microsoft ActiveX Template Library support, with which you can create COM objects from a set of template-based C++ classes.

8

The Windowing Environment

This chapter discusses the Common Desktop Environment (CDE), and the X11R6.3 and Motif components of the Tru64 UNIX windowing environment. The following topics are discussed:

- The Common Desktop Environment (Section 8.1)
- The X Window System (Section 8.2)
- The Motif Version 1.0 suite of components (Section 8.3)

8.1 Common Desktop Environment

The Common Desktop Environment (CDE) is a jointly developed graphical user interface based on industry standards. It is built upon the Open Software Foundation's Motif user interface. CDE provides a consistent look and feel as well as common APIs across multivendor platforms.

CDE presents a visual desktop that you can customize. Using the CDE interface, you can use the mouse or keyboard to navigate and interact with applications. The desktop itself offers a Front Panel, which is a graphical display at the bottom of the screen area that provides access to applications, printers, and frequently used objects, including online help.

In addition to user services, CDE provides everything needed to implement fully integrated applications. Because CDE is standards based, such integration work is transportable to other platforms that comply with these standards. For example, the help files and the means to access them apply across all compliant platforms. For more information, see the *Common Desktop Environment: Programmer's Overview*.

The CDE Front Panel displays the tools that you use to start applications, manage tasks in a desktop session, or change workspaces. Each tool is represented by an icon that indicates its purpose. A workspace is the screen itself, which includes the Front Panel. A tool on the Front Panel is provided to switch between different workspaces. The following tools are available on the Front Panel:

- Clock
- Calendar
- File Manager

- Text Editor
- Mailer
- Lock
- Workspace Switches
- Busy Light
- Printer
- Exit
- SysMan Menu
- Style Manager
- Application Manager
- Help Manager
- Trash Can

For detailed information on the use of each tool, see the *Common Desktop Environment: User's Guide*.

CDE Window List is incorporated into the Common Desktop Environment; it is invoked by clicking the middle mouse button while the mouse pointer is located in the root window. It finds application windows on any workspace of your desktop and searches for a specific application window given its name, application class, or workspace name. For more information, see the online help for CDE Window List.

CDE Setup facilitates the setting up and customization of CDE through a graphical user interface, and facilitates the creation of Front Panel controls, which can be assembled into Front Panel types. These types can be customized for different users. Furthermore, the user can set parameters for CDE windows, icons, and bindings; set options for the `dtterm`, `dtmail`, and `dtfile` applications; and determine their Front Panel terminal emulator.

The system administrator has increased capabilities including the configuration of the X server, login greeting, and system services. See the `dtsetup`(8) reference page and the online help for CDE Setup for further information. For more information, see the CDE documentation set.

8.2 X Window System

The X Window System Version 11, Release 6.3 software consists of the following components:

- X client libraries
- X server

- Display Manager
- X Protocol Extensions
- Font server
- X clients
- X print server (Xp)

8.2.1 X Client Libraries

Tru64 UNIX supports the complete set of X11R6.3 X client libraries:

- Athena Widget Set (`libXaw`)

  A high-level library of user-interface components (scroll bars, labels, buttons)

- X Intrinsics Library (`Xt`)

  Middle-level routines that call into `Xlib`

- X library (`Xlib`)

  Low-level routines that interface with the X server

For more information on individual X client libraries, see the X Window System documentation in the Tru64 UNIX documentation set.

8.2.2 X Server

Through the extensive use of shared libraries, Tru64 UNIX supports a single X11R6.3 X server image for all graphic options. The X server dynamically configures itself during initialization, loading only those server components required by a specific system configuration, and rarely requires any intervention by a system administrator.

8.2.3 Multihead Graphic Support

Multihead graphic support is transparent in Tru64 UNIX, provided the proper option cards are installed and support is built into the kernel. The PanoramiX extension, known as Xinerama in X11R6.4, can be used to take better advantage of multihead systems.

8.2.4 X Server Extensions

Tru64 UNIX supports the following X server extensions. (Note that to conserve memory, the X server, by default, defers loading most server extensions until it receives a request from a client for that specific extension.)

- The Keyboard Extension for X11R6.3

The Keyboard Extension for X11R6.3 (XKB) server extension enhances control and customization of the keyboard under the X Window System by providing:

  – Support for the ISO 9996 standard for keyboard layouts
  – Compatibility with the core X keyboard handling; no client modifications needed
  – Standard methods for handling keyboard LEDs and locking modifiers such as CapsLock and NumLock
  – Support for keyboard geometry

  Additionally, the X11R5 AccessX server extension for people with physical impairments has been incorporated into the XKB server extension. These accessibility features include StickyKeys, SlowKeys, BounceKeys, MouseKeys, and ToggleKeys, as well as complete control over the autorepeat delay and rate.

- XKME (X Keyboard Management Extension)

  Provides an internal extension for better support of international X clients. Note that XKME has been made obsolete by the XKB extension, but is provided for backwards compatibility.

- MIT-SHM (MIT Shared memory)

  Enhances performance for local image-intensive applications.

- MIT-SUNDRY-nonstandard

  Miscellaneous extensions from the X Consortium, which controls bug compatibility modes for the X server.

- Multibuffering

  Supports smooth animations by drawing to multiple buffers.

- SHAPE

  Supports nonrectangular windows used for round, oval, and irregularly shaped windows.

- SMT (Shared Memory Transport)

  Allows for the use of shared memory as an X transport for local clients, giving a significant performance boost. The transport is specified by `local:0.0`.

- XIE (X Imaging Extension, Versions 3 and 5)

  Provides advanced control over imaging, as well as device-independent image display.

  Tru64 UNIX ships both Version 3 (`/usr/lib/Xie.a` and `/usr/shlib/libXie.so`) and the de facto industry standard, Version 5 (`/usr/lib/libXIE.a` and `/usr/shlib/libXIE.so`).

- X Input

  Allows users to write their own drivers for third-party input devices, and then load them dynamically into the X server by making entries in the X server configuration file (`/usr/var/X11/Xserver.conf`). The new input devices are then recognized the next time the X server is reset.

  In traditional, statically linked X servers, each time a new extension device is added the X server must be rebuilt. The Tru64 UNIX loadable X server implementation has overcome this limitation by permitting system administrators to add new input device support as external sharable devices that are loaded by the X server at initialization.

  Sample code showing how such a driver should be written is included in the `/usr/examples` directory.

- X Screen Saver

  Enables a client to receive notification when the screen has been inactive for a specified amount of time or whenever it cycles. This extension is useful to developers writing screen saver applications.

- XSync

  The `XSync` function, in conjunction with the `XFlush`, `XEventsQueued`, and `XPending` functions, allows synchronization between X clients to take place entirely within the X server, thereby eliminating any errors introduced by the network and enabling different hosts running different operating systems to synchronize X clients. This extension is particularly useful for multimedia applications that require the synchronization of audio, video, and graphics, and for animation applications that can have their requests synchronized to internal, X server timers.

- XTest

  Allows applications to simulate X events for testing purposes.

- XTrap

  Supports the recording and playback of X events for the purpose of X client testing.

- XV (X Video)

  Allows clients to control video options, such as the live video PIP option for the TX graphic device.

- X Print Extension

  Allows X applications to output directly to a print device.

- X Print Server (Xp)

  Allows X imaging to nondisplay devices such as printers and fax machines. The core of the X Print service is the X Print Server. Applications that require printing operations can make a connection to

the X Print Server and list the available printers, select a printer, and submit print requests.

- PanoramiX

  Allows a system configured with multiple video monitors (a multiheaded system) to operate the monitors as a single large screen. Windows can span multiple screens and can move from one screen to another. This extension is only supported in homogeneous graphics environments; the environment must consist of common devices, visuals, depths, resolutions, and so on.

- Application Group

  Allows for grouping of windows such that an application group window can intercept certain requests for windows within its group and handle them appropriately. This is used with the remote execution extension to allow X applications to be embedded in Web pages.

- X Security

  Provides for enhanced security for the X Window System. Security can be specified on a per-user basis or a per-resource basis.

- Remote Execution

  Allows X applications to be invoked remotely, specifically from within a Web browser.

- Low Bandwidth X (lbx)

  Allows X applications to run more efficiently over low bandwidth transports such as a modem.

For more information on individual X client libraries, see the X Window System documentation in the Tru64 UNIX documentation set and the `X`(1X) and the `Xdec`(1X) reference pages.

8.2.5 Display Manager

Tru64 UNIX supports the standard `Xdm` terminal manager software. The `Xdm` terminal manager starts the X server locally and allows for network-transparent login prompting, so that users can log in to any system on their network supported by `xdm` as if the remote system's graphic console were in front of them. This software provides for the seamless integration of X terminals into the Tru64 UNIX environment. For more information on using `Xdm`, see the *System Administration* manual and the `xdm`(1X) reference page.

Keymap Format

The default keymaps used by Tru64 UNIX use the XKB standard keymap format. These keymaps can be used by any X server that supports and runs

the XKB extension. The XKB keymap files are text files that can be easily customized and compiled for use with the system.

The xmodmap keymap format is also supplied for backward compatibility. Should you want to run the xmodmap keymap format, custom key maps should use this format. The xmodmap keymap format is a de facto industry standard, which uses symbolic key names and can be easily customized.

Both these keymap formats support the ability to specify modifier keys (Compose, Alt, Shift, and so forth).

You should use the XKB standard keymap format instead of the xmodmap keymap format.

XDM-AUTHORIZATION-1

Whenever an X client application establishes a connection to the X server, it passes an authorization code, called a key, to the X server. If the X server recognizes this key, the connection is allowed. When the user's X session is started, xdm (the X Display Manager) writes one or more keys into the .Xauthority file in a user's home directory. The X Display Manager (xdm) also writes these keys into a file that the X server can read.

To improve security, Tru64 UNIX supports both the MIT-MAGIC-COOKIE-1 key format as well as the XDM-AUTHORIZATION-1 encrypted key format, which is the default.

8.2.6 Font Server

Tru64 UNIX supports a standard scalable font server that supplies a network of systems with access to fonts that reside on any Tru64 UNIX system. The font server maintains a repository of fonts and responds to requests from other X servers on the network for fonts that they may not have locally. In addition to providing network-transparent access to fonts, the font server unloads the compute burden of font scaling from local X servers, because it scales fonts appropriately before supplying them to the requesting X server.

Before a font can be displayed by an X server, its glyphs must be converted from their on-disk formats into bitmaps. This conversion is done by font renderer code in the X server or in a font server that may be supplying fonts to the X server.

Tru64 UNIX supports loadable font renderers, so that users who adhere to the X11R6.3 standard can write their own font renderer for their own set of fonts and install them on a Tru64 UNIX system. After the fonts and the font renderer are installed, the necessary entries for them are placed in the X server configuration file (/usr/var/X11/Xserver.conf), the font server configuration file (/usr/var/X11/fs/config), or in both configuration

files. The new font renderer is then recognized the next time the X server or font server (whichever has the font renderer configured) is reset.

8.2.7 X Clients

Tru64 UNIX supports the entire X11R6.3 suite of X clients, including `appres`, `atobm`, `bdftopcf`, `bitmap`, `bmtoa`, `chooser`, `editres`, `fs`, `fsinfo`, `fslsfonts`, `fstobdf`, `iceauth`, `ico`, `imake`, `listres`, `lndir`, `maze`, `mkfontdir`, `oclock`, `optacon`, `puff`, `puzzle`, `restart`, `resize`, `showfont`, `showrgb`, `smproxy`, `twm`, `uil`, `viewres`, `x11perf`, `x11perfcomp`, `xauth`, `xbiff`, `xcalc`, `xcd`, `xclipboard`, `xclock`, `xcmsdb`, `xconsole`, `xcutsel`, `xdm`, `xdpr`, `xdpyinfo`, `xedit`, `xemacs`, `xev`, `xeyes`, `xfd`, `xfindproxy`, `xfwp`, `xfontsel`, `xfwp`, `xhost`, `xkbbells`, `xkbcomp`, `xkbprint`, `xkbdfltmap`,, `xkbvleds`, `xkbwatch`, `xkill`, `xload`, `xlogo`, `xlsatoms`, `xlsclients`, `xlsfonts`, `xmag`, `xman`, `xmbind`, `xmh`, `xmkmf`, `xmodmap`, `xon`, `xpr`, `xprop`, `xrdb`, `xrefresh`, `xset`, `xsetroot`, `xsoundsentry`, `xstdcmap`, `xterm`, `xwd`, `xwininfo`, and `xwud`.

For more information on individual X clients, see the appropriate reference page.

8.3 Motif Suite of Components

Tru64 UNIX supports the entire suite of Motif Version 1.0 components, including the widget library (`Xm`), the resource manager (`Mrm`), the widget metalanguage (`wml`), the User Interface Language (UIL), the Motif window manager (`mwm`), the key binding utility (`xmbind`), and the Motif Demonstration programs (`examples`).

In Motif Version 1.2, the Open Software Foundation added support for ANSI C, Internationalization, Drag and Drop, and TearOff Menus. Unfortunately, much of this support required breaking binary compatibility with Motif Version 1.1.3.

To mitigate this problem, Tru64 UNIX provides the Motif Version 1.1.3 libraries through versioning to allow applications that are linked against Motif Version 1.1.3 to continue to run. These libraries are available in an optional subset. For more information on versioning, see the *Programmer's Guide*.

For more information on Motif, see the *OSF/Motif Programmer's Guide* and the appropriate reference pages. For information on Motif support for internationalization, see Chapter 10.

8.3.1 Extended Widget Set

Tru64 UNIX supports the Extended Widget Set (DXm), which contains the following widgets:

- DXmColorMix

  Supports editing and the selection of colors

- DXmPrintWidget

  Presents graphical print options

- DXmCSText

  Supports the editing of compound strings in a user interface similar to XmText

- DXmHelpWidget

  Displays help topics

- DXmSvn

  Supports structured navigation through lists of data

8.3.2 X Clients

In addition to the entire X11R6.3 suite of X clients, Tru64 UNIX supports a variety of X clients including accessx, dxconsole, dxdiff, dxkbledpabel, dxkeyboard, dxkeycaps, dxpresto, and dxterm.

9

Security

This chapter looks at the security features provided by Tru64 UNIX. The first section provides a brief security overview (Section 9.1), after which the following topics are discussed:

- Identification and authentication (Section 9.2)
- Audit features (Section 9.3)
- Discretionary access controls (Section 9.4)
- Security administration (Section 9.5)
- Object reuse (Section 9.6)
- The protected environment for trusted components (Section 9.7)
- Integrity features (Section 9.8)
- Additional security features (Section 9.9)

9.1 Overview

Tru64 UNIX offers a range of security configuration options, and can be tailored to the appropriate security level of your installation. The range extends from traditional UNIX security, the default, to the optional enhanced security subsets that, when enabled, satisfy or exceed the requirements of the C2 evaluation class of *DoD 5200.28-STD Trusted Computer System Evaluation Criteria* (TCSEC), also known as the Orange Book.

For specific information, see the *Security* manual.

9.2 Identification and Authentication

Security Integration Architecture (SIA) allows a single set of identification and authentication (I and A) utilities to work in both base mode (as a nontrusted system) and in enhanced mode.

Base mode is the default and enhanced mode is available through optional security subsets. The SysMan interface can enable enhanced mode in one of two ways:

- Shadow passwords only
- Custom (providing the means to completely configure)

The following I&A features are provided on a system running enhanced security:

- Password control
  - Configurable maximum password length up to 80 characters.
  - Configurable password lifetimes. This includes an optional minimum interval between password changes.
  - A floating value of the minimum password length, based directly on the *Department of Defense Password Management Guideline* (the Green Book) guidelines and the password lifetime.
  - User password-generation flags, which include the ability to require a user to have a generated password.
  - Recording of who (besides the user) last changed the user's password.
- Login control
  - Optional recording of the last terminal and time of the last successful login and of the last unsuccessful login attempt.
  - Automatic account lockout after a specified number of consecutive bad access attempts (break-in detection and evasion).
  - A per-terminal setting for the delay between consecutive login attempts and the maximum amount of time each attempt is allowed before being declared a failed attempt.
  - A per-terminal setting for maximum consecutive failed login attempts before locking any new accesses from that terminal.
- Differentiation between "retired" and "locked" accounts.
- Configurable multilevel system default values for the various I&A fields (templates).
- A CDE-based graphical interface (`dxaccounts`) to perform user-account management and other I&A administration tasks.
- The `edauth`, `convauth`, and `convuser` utilities to ease the migration of accounts to the enhanced security level.

9.3 Audit System

The following audit features are provided:

- A graphical interface, `audit_setup`, to simplify system auditing
- Command line interfaces
- The ability to send audit logs to a remote host
- Event profiles allowing simplified configuration based on system type

- Fine-grained preselection of system events, application events, and site-definable events
- Fine-grained post-analysis of system events, application events, and site-definable events
- Optional automated audited log cleanup
- The `dxaudit` GUI

When used with enhanced security, auditing includes support for a per-user audit characteristics profile with enhanced identification and authorization. The audit system is set up through SysMan or by using the `audit_setup` utility from the command line. Maintenance for the audit system is done from the command line or with the `dxaudit` GUI.

9.4 Discretionary Access Controls

Discretionary access controls (DACs) provide the capability for users to define how the resources they create can be shared. The traditional UNIX permission bits provide this capability.

Tru64 UNIX also provides optional access control lists (ACLs) for object protection at the individual user level. ACLs are supported under the UFS, NFS, and AdvFS file systems. To simplify ACL management, an ACL GUI, named `dxsetacl` is available in addition to the command line interface.

9.5 Security Administration

Tru64 UNIX provides system administrators with tools to ease the of use of administration of system security.

9.5.1 Configuring System Security

System administrators can select the security level for their system. The default base security level consists of object reuse and discretionary access control (DAC). System administrators can select enhanced security features and ACLs by using SysMan or by running the`secsetup` utility from the command line. The audit system is configurable through SysMan or by using the `audit_setup` utility from the command line.

9.5.2 Windows-Based Administration Utilities

Tru64 UNIX provides four graphical utilities to deal with day-to-day security administration on the local machine:

- The `dxaccounts` utility (Account Manager under the CDE-based system administration utilities), which operates in both base and enhanced security modes, provides the ability to create template accounts and to

modify selected system defaults. Under enhanced security, you can use `dxaccounts` to specify a set of events to audit (audit mask) for each user.

- The `dxaudit` utility controls the administration of the audit system (for instance, defining which events to audit) and the generation of audit reports. Auditing is available with both base and enhanced security.
- The `dxsetacl` utility (ACL Manager under the CDE based system administration utilities) is used to create, modify, and delete ACLs on files and directories.
- The `dxdevices` utility configures secure devices in enhanced security mode.

For more information, see the `dxaccounts`(8X), `dxsetacl`(8X), `dxaudit`(8X), and `dxdevices`(8X) reference pages.

9.6 Object Reuse

Object reuse ensures that the following types of physical storage (memory or disk space) is cleared ("scrubbed."):

- Physical storage that is assigned to shared objects
- Physical storage that is released prior to reassignment to another user

Examples of object reuse are disk space that is released after a file is truncated or physical memory that is released prior to reassignment to another user to read.

9.7 Protected Environment for Trusted Components

Tru64 UNIX uses hardware memory management to maintain a kernel address space for itself and to maintain separate address spaces for each instance of an executing application process. Processes may try to write to the same address space. DACs control the sharing of this address space among processes; the default is to disallow sharing.

The administrator can disable the sharing of sections as read-only address space; for example, shared libraries. Thus, the security-relevant components of the system (the trusted computing base, or TCB) are protected while they execute.

Tru64 UNIX protects the on-disk security components using discretionary access control. Attempted violations of the DAC protections can be audited so that remedial action can be taken by the system security officer.

In addition, the security components are structured into well defined, largely independent modules.

Tru64 UNIX is designed, developed, and maintained under a configuration management system that controls changes to the specifications, documentation, source code, object code, hardware, firmware, and test suites. Tools, which are also maintained under configuration control, are provided to control and automate the generation of new versions of the security components from source code and to verify that the correct versions of the source have been incorporated into the new version.

The master copies of all material used to generate the security components are protected from unauthorized modification or destruction.

9.8 Integrity Features

Tru64 UNIX provides the capability to validate the correct operation of hardware, firmware, and software security components. The firmware includes power-on diagnostics and more extensive diagnostics that optionally can be enabled. The firmware itself resides in EEPROM and can be physically write protected. It also can be compared against, or reloaded from, an offline master copy. Additional hardware diagnostics can be used also.

The firmware can require authorization to load any operating software other than the default or to execute privileged console monitor commands that examine or modify memory.

Once the operating system is loaded, administrators can run system diagnostics to validate the correct operation of the hardware and software. In addition, test suites are available to ensure the correct operation of the operating system software.

The following tools can be run automatically to detect inconsistencies in the security software and databases:

- `fverify`

  This utility reads subset inventory records from standard input and verifies that the attributes for the files on the system match the attributes listed in the corresponding records. Missing files and inconsistencies in file size, checksum, user ID, group ID, permissions, and file type are reported.

- `authck` utility

  This utility checks both the overall structure and internal field consistency of all components of the authentication database and reports problems that it finds.

9.9 Other Security Features

The following sections describe additional security features that are integrated into the Tru64 UNIX software and separately available programs that provide addtional security features to the operating system.

9.9.1 Features Included with the Operating System

Tru64 UNIX supports features that are unavailable in other OSF-based UNIX operating systems.

9.9.1.1 The Security Integration Architecture Layer

All security mechanisms that run on the Tru64 UNIX operating system run under the Security Integration Architecture (SIA) layer. The SIA allows a suitably privileged administrator to layer various local and distributed security authentication mechanisms onto Tru64 UNIX with no modification to the security-sensitive Tru64 UNIX commands, such as `login`, `su`, and `passwd`. The SIA isolates the security-sensitive commands from the specific security mechanisms, thus eliminating the need to modify them for each new security mechanism.

Because of the presence of the SIA, administrators can use the `secconfig` command to move back and forth between the secure and nonsecure commands and utilities. The SIA also supports the following:

- DECnet interoperability

  The SIA interface provides support for the DECnet/OSI networking software.

- Distributed Computing Environment interoperability

  When Tru64 UNIX is configured for enhanced security, the SIA allows you to enter both your system password and your Distributed Computing Environment (DCE) password at login time. You do not have to log in to the Tru64 UNIX secure system and then log in again to DCE.

9.9.1.2 Network Information Service Compatibility

Tru64 UNIX provides support for accessing Network Information Service (NIS) distributed databases while running enhanced security. For example, administrators can use the `ypcat passwd` utility to gather information about users on the network. However, the user's encrypted password in the NIS distributed password database is not the same as the encrypted password on the secure system, which cannot be viewed by unprivileged users.

In addition, on a Tru64 UNIX system running enhanced security, NIS can be used to distribute the enhanced security protected password database.

9.9.1.3 Configuration and Setup Script

Tru64 UNIX supports the `secsetup` configuration and setup script, which allows you to select the security level you want to run, permits you to move back and forth between secure and nonsecure commands and utilities, and configures security at boot time, depending on the value of the `SECURITY` variable in the `/etc/rc.config` file.

9.9.1.4 Graphical User Interfaces

Tru64 UNIX provides the `dxaccounts`, `dxaudit`, and `dxdevices` graphical interfaces that administrators can use to create and modify user accounts, modify system defaults, modify the audit interfaces and devices. The `dxsetacl` graphical interface lets users manipulate ACLs on system objects.

9.9.1.5 Division of Privileges

The division of privileges (`dop`) utility lets a system administrator with `root` privileges assign access to certain classes of administrative tasks to other users or groups of users, thereby minimizing access to the powerful `root` account.

With the assigned privileges, users and groups can execute the selected privileged programs without knowing the root password. For example, a user granted the `AccountManagement` privilege can run the tasks listed under the Accounts branch of the SysMan Menu.

The `dop` utility is available from the command line, from the SysMan Menu, and from the CDE Application Manager.

9.9.2 Programs to Augment Tru64 UNIX Security

The following sections describe programs that are available on the Tru64 UNIX Associated Products CD–ROMS and from Compaq on the Internet.

9.9.2.1 Common Data Security Architecture

The Common Data Security Architecture (CDSA) is a multiplatform, industry-standard security infrastructure that is available for Tru64 UNIX as an advanced developers kit (ADK). It provides a standards-based, stable programming interface that applications can use to access operating system security services, allowing developers to create cross-platform, security-enabled applications.

The CDSA kit is available at the following Compaq Web site:

`http://tru64unix.compaq.com/internet/download.htm`

9.9.2.2 Single Sign On

Single Sign On (SSO) is an optional user authentication security feature included on a Tru64 UNIX Associated Products CD-ROM. Based on Kerberos technology, the Tru64 UNIX SSO software increases an organization's level of security and decreases user account maintenance and administration in a heterogeneous intranet.

9.9.2.3 Security Products on the Internet Express for Tru64 UNIX CD-ROM

Several security products are included on the Internet Express for Tru64 UNIX CD-ROM (see Section 1.3.7.1). Those products include a secure socket layer (SSL), the FireScreen firewall product, the RID Denial of Service Scanner, and `tcpwrapper`.

Other security utilities such as `tripwire`, `wuftpd`, `lsof`, and `crack` are available on the Tru64 UNIX Open Source Software Collection CD–ROM (see Section 1.2.10) or from public domain sites on the Internet.

10

Internationalization

This chapter describes the internationalization features of Tru64 UNIX. The first section provides a brief internationalization overview (Section 10.1), after which the following topics are discussed:

- Supported languages (Section 10.2)
- Using the `localedef` utility to create locales (Section 10.3)
- The enhanced terminal subsystem for Asian languages (Section 10.4)
- Enhanced sorting for Asian languages (Section 10.6)
- The Multilingual Emacs editor (Section 10.7)
- Support for user-defined characters in Chinese, Japanese, and Korean (Section 10.8)
- Converting text from one codeset to another (Section 10.9)
- Support for the Unicode and ISO 10646 standards (Section 10.10)
- Support for the euro currency symbol (Section 10.11)
- The internationalized Curses library (Section 10.12)
- Internationalized printing (Section 10.13)
- The graphical internationalization configuration tool (Section 10.14)
- Support for 8–bit character encoding in mail programs (Section 10.15)
- Enhancements to the `file` command (Section 10.16)
- Internationalization for graphical applications (Section 10.17)

10.1 Overview

The term “internationalization” is formally defined by The Open Group as a

> “provision within a computer program of the capability of making itself adaptable to the requirements of different native languages, local customs, and coded character sets”

This essentially means that internationalized programs can run in any supported locale without having to be modified. A locale is a software environment that correctly handles the cultural conventions of a particular geographic area, such as China or France, and a language as it is used in that area. So by selecting a Chinese locale, for example, all commands,

system messages, and keystrokes can be in Chinese characters and displayed in a way appropriate for Chinese.

Tru64 UNIX is an internationalized operating system that not only allows users to interact with existing applications in their native language, but also supports a full set of application interfaces, referred to as the Worldwide Portability Interfaces (WPI), to enable software developers to write internationalized applications. The original code for these interfaces came from the Open Software Foundation (OSF) and has been enhanced.

The internationalization support in the operating system conforms to The Open Group's CAE specifications for system interfaces and headers (XSH Issue 5), curses (XCURSES Issue 4.2), and commands and utilities (XCU Issue 5). These specifications align with current POSIX and ISO C standards. This conformance ensures that commands, utilities, and libraries have been internationalized, and their corresponding message catalogs have been included in the base system.

In addition, the operating system supports the X Input Method (XIM) and X Output Method(XOM) to facilitate input of local language characters, text drawing, measurement, and interclient communication. These functions are implemented according to the X11R6.3 specification and include some problem corrections specified by X11R6.4.

Note that the operating system also supports a 32-bit `wchar_t` datatype which in turn enables support for a wide array of codesets, including the one defined by the ISO 10646 standard.

For more comprehensive information, see *Writing Software for the International Market*.

10.2 Supported Languages

Table 10–1 lists the languages supported by the operating system and their corresponding locales. Most locales are included in Worldwide Language Support (WLS) subsets that are optionally installed. Some, as indicated in the table, are part of the mandatory base operating system.

Locale names that include `@ucs4` and `UTF-8` support character encoding as defined by the ISO 10646 standard. The `wchar_t` data type is 32 bits in length, with zero padding of leading bits for those character values that do not require all 32 bits. The first 256 values of the Universal Character Set (UCS) define the same characters as defined for the ISO 8859–1 (Latin-1) character set. Therefore, the Tru64 UNIX implementation of the `wchar_t` data type is identical to UCS-4 process code for all `ISO8859-1` locales. `ISO8859-1` locales differ from `UTF-8` locales with the same base name in terms of data file encoding and the fact that only the `UTF-8` locales support the euro currency symbol.

The English locale name that includes `cp850` supports character encoding in PC code-page format.

For the most up-to-date list of supported languages and locales, refer to the `l10n_intro`(5) reference page. This book may not be updated for minor functional releases of the operating system and locales are sometimes added for such releases.

Table 10–1: Languages and Locales

Language	Locale Name
Catalan	ca_ES.ISO8859-1 [a] ca_ES.ISO8859-15 ca_ES.UTF-8
Simplified Chinese (PRC)	zh_CN.dechanzi zh_CN.dechanzi@ucs4 zh_CN.dechanzi@pinyin zh_CN.dechanzi@pinyin@ucs4 zh_CN.dechanzi@radical zh_CN.dechanzi@radical@ucs4 zh_CN.dechanzi@stroke zh_CN.dechanzi@stroke@ucs4 zh_CN.GBK zh_CN.GB18030
Traditional Chinese (Hong Kong)	zh_HK.big5 zh_HK.dechanyu zh_HK.dechanyu@ucs4 zh_HK.dechanzi zh_HK.dechanzi@ucs4 zh_HK.eucTW zh_HK.eucTW@ucs4

Table 10–1: Languages and Locales (cont.)

Language	Locale Name
Traditional Chinese (Taiwan)	zh_TW.big5 zh_TW.big5@chuyin zh_TW.big5@radical zh_TW.big5@stroke zh_TW.dechanyu zh_TW.dechanyu@ucs4 zh_TW.dechanyu@chuyin zh_TW.dechanyu@chuyin@ucs4 zh_TW.dechanyu@radical zh_TW.dechanyu@radical@ucs4 zh_TW.dechanyu@stroke zh_TW.dechanyu@stroke@ucs4 zh_TW.eucTW zh_TW.eucTW@ucs4 zh_TW.eucTW@chuyin zh_TW.eucTW@chuyin@ucs4 zh_TW.eucTW@radical zh_TW.eucTW@radical@ucs4 zh_TW.eucTW@stroke zh_TW.eucTW@stroke@ucs4
Czech	cs_CZ.ISO8859-2 cs_CZ.ISO8859-2@ucs4
Danish	da_DK.ISO8859-1 [a] da_DK.ISO8859-15 da_DK.UTF-8
Dutch	nl_NL.ISO8859-1 [a] nl_NL.ISO8859-15 nl_NL.UTF-8
Belgian Dutch	nl_BE.ISO8859-1 [a] nl_BE.ISO8859-15 nl_BE.UTF-8
US English/ASCII	C (POSIX) [a]
US English	en_US.ISO8859-1 [a] en_US.ISO8859-15 en_US.cp850. en_US.UTF-8, en_US.UTF-8@euro [b]
GB English	en_GB.ISO8859-1 [a] en_GB.ISO8859-15 en_GB.UTF-8
European	en_EU.UTF-8@euro [c]

Table 10–1: Languages and Locales (cont.)

Language	Locale Name
Finnish	fi_FI.ISO8859-1 [a] fi_FI.ISO8859-15 fi_FI.UTF-8
French	fr_FR.ISO8859-1 [a] fr_FR.ISO8859-15 fr_FR.UTF-8
Belgian French	fr_BE.ISO8859-1 [a] fr_BE.ISO8859-15 fr_BE.UTF-8
Canadian French	fr_CA.ISO8859-1 [a] fr_CA.ISO8859-15 fr_CA.UTF-8
Swiss French	fr_CH.ISO8859-1 [a] fr_CH.ISO8859-15 fr_CH.UTF-8
German	de_DE.ISO8859-1 [a] de_DE.ISO8859-15 de_DE.UTF-8
Swiss German	de_CH.ISO8859-1 [a] de_CH.ISO8859-15 de_CH.UTF-8
Greek	el_GR.ISO8859-7, el_GR.ISO8859-7@ucs4 el_GR.UTF-8
Hebrew	he_IL.ISO8859-8 he_IL.ISO8859-8@ucs4
Hungarian	hu_HU.ISO8859-2 hu_HU.ISO8859-2@ucs4
Icelandic	is_IS.ISO8859-1 [a] is_IS.ISO8859-15
Italian	it_IT.ISO8859-1 [a] it_IT.ISO8859-15 it_IT.UTF-8
Japanese	ja_JP.eucJP ja_JP.SJIS ja_JP.SJIS@ucs4 ja_JP.deckanji ja_JP.deckanji@ucs4 ja_JP.sdeckanji

Table 10–1: Languages and Locales (cont.)

Language	Locale Name
Korean	ko_KR.deckorean ko_KR.deckorean@ucs4 ko_KR.eucKR ko_KR.KSC5601
Lithuanian	lt_LT.ISO8859-4 lt_LT.ISO8859-4@ucs4
Norwegian	no_NO.ISO8859-1 [a] no_NO.ISO8859-15 no_NO.UTF-8
Polish	pl_PL.ISO8859-2 pl_PL.ISO8859-2@ucs4
Portuguese	pt_PT.ISO8859-1 [a] pt_PT.ISO8859-15 pt_PT.UTF-8
Russian	ru_RU.ISO8859-5 ru_RU.ISO8859-5@ucs4
Slovak	sk_SK.ISO8859-2 sk_SK.ISO8859-2@ucs4
Slovene	sl_SI.ISO8859-2 sl_SI.ISO8859-2@ucs4
Spanish	es_ES.ISO8859-1 [a] es_ES.ISO8859-15 es_ES.UTF-8
Swedish	sv_SE.ISO8859-1 [a] sv_SE.ISO8859-15 sv_SE.UTF-8
Thai	th_TH.TACTIS
Turkish	tr_TR.ISO8859-9 tr_TR.ISO8859-9@ucs4

[a] This locale is included in the base installation rather than in an optional worldwide language support subset.
[b] The en_US.UTF-8@euro locale defines the local currency sign to be the euro character and the international currency sign to be EUR.
[c] The en_EU.UTF-8 locale defines the local currency sign to be the euro character, the international currency sign to be EUR, the decimal point to be a comma (,), and the thousands separator to be a period (.). When assigned specifically to the LC_MONETARY locale category or environment variable, this locale is useful in many European countries, not just those in which English is spoken.

Note that you can switch languages or character sets as necessary and can even operate multiple processes in different languages or codesets in the same system at the same time.

For more information on a particular coded character set, such as `ISO8859-9`, see the reference page with the same name. For more information about `UCS-4` and `UTF-8` encoding, see `Unicode`(5). For more information about PC code pages, see `code_page`(5).

10.3 Locale Creation

The `localedef` utility allows programmers to create their own locales, compile their source code, and generate a unique name for their new locale.

For more information on creating locales, see *Writing Software for the International Market*.

10.4 Enhanced Terminal Subsystem for Asian Languages

The base `tty` terminal driver subsystem is extended to include additional BSD line disciplines and STREAMS terminal driver modules for processing data in Chinese, Japanese, Korean, and Thai. For example, the enhanced terminal subsystem supports the following capabilities for these languages:

- Japanese Kana-Kanji conversion input method
- Character-based line processing in cooked mode
- Input line history and editing (BSD line discipline only)
- Software on-demand-loading for user-defined characters
- Conversion between terminal code and application code

For more information about the Asian and Thai terminal subsystems, see the `atty`(7), `ttty`(7), and `stty`(1) reference pages.

10.5 Enhanced wwconfig Command

The `wwconfig` command configures terminal (tty) options for Asian countries. The command has been enhanced to include the following:

- The `-vmunix` and `-kernel` options, which display I18N modules that are statically linked into `/vmunix` or currently used by the running kernel
- The `-pty` option, which specifies the use of a BSD or streams pseudo-driver for remote logins and `telnet` sessions
- The `-config` option, which specifies a kernel configuration file
- The`-utx` option, which specifies the addition of Kana-Kanji, On Demand Loading, or Software Phrase Input as Asian tty driver options
- The `-code` option, which specifies the addition of BIG5, Mitac, Chinese mapping support, and UTF-8 character set support as Asian tty driver options

- The -[no]thai option, which includes or excludes the Thai tty driver
- The -utxnum option, which specifies the number of utx pseudo-devices to be created

For information on the use and specification of these options, see the wwconfig(8) reference page.

10.6 Enhanced Sorting for Asian Languages

The operating system supports the asort utility, an extension of the sort command, which allows characters of ideogrammatic languages, like Chinese and Japanese, to be sorted according to multiple collation sequences. For more information on the asort utility, see asort(1).

10.7 Multilingual Emacs Editor

The operating system supports the Multilingual Emacs editor (MULE) for Asian languages. See mule(1) for more information.

10.8 Support for User-Defined Characters

The operating system provides support for creating user-defined characters (UDCs) in Chinese, Japanese, and Korean, so that users can create and define character fonts and their attributes, including bitmap fonts, with the cedit and cgen utilities.

Font rendering facilities are available so that X clients can use UDC databases through the X server or font server to obtain bitmap fonts for user-defined characters.

For more information on user-defined characters, see *Writing Software for the International Market*, cedit(1) and cgen(1).

10.9 Codeset Conversion

The operating system includes the iconv utility and the iconv_open(), iconv(), and iconv_close() functions, which convert text from one codeset to another, thereby assisting programmers in the writing of international applications. For use with these interfaces, the operating system includes a large set of codeset converters.

A new en_US.UTF-8 X locale database file contains font definitions that include all the various fonts used with the operating system. Thus, applications running under the en_US.UTF-8 locale can display all the font characters installed with Worldwide Language Support (WLS). Applications running under the Asian locales display all of the WLS installed fonts, except for ISO8859-2, -4, -5, -7, -8, -9, and TACTIS.

In addition to conversion between different codesets for the same language, these converters support conversion between different Unicode formats, such as UCS-2, UCS-4, and UTF-8. There are also codeset converters that handle the most commonly used PC code-page formats.

Codeset conversion is also used by the printing subsystem and utilities, such as man, to allow processing of files in different languages and encoding formats. Additionally, codeset conversion is implemented in mail utilities for mail interchange with systems using different codesets and in the X Windows System Toolkit for text input, drawing, and interclient communication. For more information on codeset conversion, see the iconv_intro(5) reference page. See the Unicode(5) and code_page(5) reference pages for a discussion of converters for Unicode encoding formats and PC code-page formats, respectively.

10.10 Unicode Support

The operating system provides both codeset converters and locales that support the Unicode and ISO 10646 standards. The codeset converter modules convert between other supported codesets and UCS-2, UCS-4, and UTF-8 formats. In addition to the country-specific and language-specific locales listed in Section 10.2, programmers can use the universal.UTF-8 locale to process characters in all languages by using UCS-4 encoding format.

The operating system provides a function called fold_string_w(), which maps one Unicode string to another and performs the specified Unicode transformations. For more information on the fold_string_w() function, see fold_string_w(3). For more information on Unicode support, see Unicode(5).

10.11 Support for the Euro Character

The operating system supports the new euro currency now being used by member countries of the European Economic and Monetary Union (EMU).

Locales that use the UTF-8 or Latin-9 (ISO8859-15) codesets support the euro character. Those locales whose names include the @euro suffix also define the local currency sign to be the euro character and the international currency sign to be EUR. See Section 10.2 for more information about locales.

The ISO Latin-9 codeset forms the basis of euro font support. The operating system includes both screen and PostScript outline fonts for this codeset. See ISO8859-15(5) for a list of these fonts.

The operating system does not provide native Unicode fonts that support the euro character. However, the X font library has been extended to combine

a number of fonts together to form logical Unicode fonts for applications to use. The names of these logical fonts include the string `ISO10646-1`.

Printer support for the euro character is enabled by a generic PostScript print filter, `wwpsof`, which supports printing of file data in UTF-8 or Latin-9 format. See `wwpsof`(8) for information on setting up printers with this print filter.

In the UTF-8 and Latin-9 locales, keyboard entry of the euro character is supported by language-specific and keyboard-specific key sequences that are defined in keymaps (XKB format). The euro character also can be entered by using a Compose key sequence on those keyboards that support a Compose key. The `euro`(5) reference page lists these key sequences.

Finally, the operating system provides codeset converters to convert file data between the various encoding formats that support the euro character. Specifically, codeset converters can convert file data between:

- Unicode encoding formats and PC code-page formats that support the euro
- Unicode encoding formats and `ISO8859-15` encoding

See `euro`(5) for a more detailed discussion of the information in this section.

10.12 Internationalized Curses Library

The operating system supplies an internationalized Curses library in conformance with X/Open Curses, Issue 4 Version 2. This library provides functions for processing characters that span one or multiple bytes. These characters may be in either wide-character (`wchar_t`) or complex-character (`cchar_t`) formats. The complex-character format provides for a single logical character made up of multiple wide characters. Some of the components of the complex character may be nonspacing characters.

For information on the syntax and effect of Curses interfaces, see `curses`(3). For a description of the enhancements provided by the internationalized Curses routines, and their relationship to previous Curses routines, see *Writing Software for the International Market*.

10.13 Internationalized Printing

The operating system supports the printing of plain text and PostScript files for a variety of languages and provides outline fonts for high quality printing on PostScript printers. In addition to print filters for a variety of local-language printers, generic internationalized print filters are available for use with both Compaq and third-party printers.

One of these filters, wwpsof, supports printing of local-language files on PostScript printers that do not include the required fonts. For more information on internationalized printing features, see the i18n_printing(5), pcfof(8), and wwpsof(8) reference pages.

10.14 Graphical Internationalization Configuration Tool

The I18N Configuration tool, available through the Application Manager, is one of the CDE System Administration Configuration applications.

The I18N Configuration tool provides a graphical interface for the system administrator to configure I18N-specific settings. It also provides a convenient way to see which countries, locales, fonts, and keymaps are supported on the host system. System administrators can also use this tool to remove unused fonts and country support from the system.

10.15 Mail and 8-Bit Character Support

By default, the operating system provides support for 8–bit character encoding in mailx, dtmail, MH, and comsat.

For more information on these mail utilities, see mailx(1), dtmail(1), mh(1), and comsat(8).

10.16 Enhanced file Command

The filecommand is enhanced to recognize UCS-2 and UCS-4 encoding in any locale setting. For other encoding formats, the command recognizes file data encoding if it is valid for the current locale setting. This command also has a jfile alias that, in any locale, can recognize DEC Kanji, Japanese EUC, Shift JIS, and 7–bit JIS encoding.

10.17 Internationalization for Graphical Applications

Motif Version 1.2.3 takes advantage of many of the internationalization features of X11R6 and the C library to support locales. Motif Version 1.2.3 also supports the use of alternate input methods, which allows input of non-ISO Latin-1 keystrokes, and delivers an extensively rewritten XmText widget, which supports multibyte and wide-character format and on-the-spot input style.

Motif supports multibyte and wide-character encoding through the use of the internationalized X Library functions, and C Library functions. In addition, the compound string routines include the X11R6 XFontSet component to allow for the creation of localized strings.

The User Interface Language (UIL) supports the creation of localized UID files through the `-s` compile-time option on the UIL compiler, which causes the compiler to construct localized strings.

Alternate input methods can be specified by a resource on the `VendorShell` widget. Widgets that are parented by a `Shell` class widget can take advantage of this resource and register themselves as using a specific method for input.

The following sections discuss internationalization features of Motif widgets and internationalized client applications.

10.17.1 Internationalized Motif Widgets

The following lists contain the widgets in the Motif Toolkit and in the Extensions to the Motif Toolkit that support local language characters, I/O capabilities, and local language message displays.

Note that the Motif UIL compiler is extended to support local language characters in UIL files.

- Motif Toolkit
  - Command
  - FileSelectionBox
  - Label
  - MessageBox
  - SelectionBox
  - Text
  - TextField
- Extensions to Motif Toolkit
  - ColorMix
  - CSText
  - Help
  - Print
  - Structured Visual Navigation (SVN)

10.17.2 Internationalized CDE Clients

CDE is the default desktop for Tru64 UNIX. The following CDE clients are internationalized:

- Application Manager

- Calculator
- Calendar
- Create Action
- File Manager
- Front Panel
- Help Viewer
- Icon Editor
- Login Screen
- Message
- Mailer
- Print Manager
- Style Manager
- Terminal Emulator
- Trash Can

By default, client applications run in the language set by the user at the start of a CDE session. However, users can also change locale in a terminal emulation window and invoke an application in a language different from the session default.

10.17.3 Additional Internationalized Motif Clients

The operating system includes the following internationalized clients in addition to those common to all CDE implementations:

- Differences
- Keycap
- DECterm
- X Display Manager

A

Internet Standards

This appendix lists Internet RFC (Request for Comment) standards and non-RFC standards to which the Tru64 UNIX operating system conforms.

A.1 RFC Standards

Table A–1 lists RFC standards to which Tru64 UNIX complies. When viewing this list on line on a system connected to the Internet, you can click on an RFC to display the text of that RFC in your Web browser.

Table A–1: RFC Standards

RFC	Protocol	Name
678		Standard File Formats
768	UDP	User Datagram Protocol
791		Internet Protocol
792	ICMP	Internet Control Message Protocol
793	TCP	Transmission Control Protocol
819	SMTP	The Domain Naming Convention for Internet User Applications
821	SMTP	Simple Mail Transfer Protocol
822	MAIL	Standard for the Format of ARPA Internet Text Messages
826	ARP	Address Resolution Protocol
854	TELNET	TELNET Protocol
855	TELNET	TELNET option specifications
856	TELNET	TELNET binary transmission
857	TELNET	TELNET echo option
858	TELNET	TELNET Suppress Go Ahead option
859	TELNET	TELNET status option
868	Time	Time Protocol
893		Trailer Encapsulations

Table A–1: RFC Standards (cont.)

RFC	Protocol	Name
894	IP-E	Internet Protocol on Ethernet Networks
903	RARP	Reverse Address Resolution Protocol
904	EGP	Exterior Gateway Protocol
919		Broadcasting Internet Datagrams
920		Domain Requirements
922		Broadcasting Internet Datagrams in the Presence of Subnets
950		Internet Standard Subnetting Procedure
951	BOOTP	The Bootstrap Protocol
954		NICNAME/WHOIS (obsoletes RFC 812)
959	FTP	File Transfer Protocol
974		Mail Routing and the Domain System
1014	XDR	External Data Representation
1032		Domain Administrators Guide
1033		Domain Administrators Operations Guide
1034		Domain Names (Concepts and Facilities) (obsoletes RFCs 882, 883, and 973)
1035	IP-IEEE	Domain Names (Implementation and Specification) (obsoletes RFCs 882, 883, and 973)
1042	IP-IEEE	Internet Protocol on IEEE 802
1049		Content-Type Field for Internet Messages
1050	RPC	Sun Remote Procedure Calls
1055	SLIP	Serial Line Internet Protocol
1057		RPC Protocol Specification Version 2 (obsoletes RFC 1050)
1058	RIP	Routing Information Protocol
1094	NFS	Network File System Protocol
1101		DNS Encoding of Network Names and Other Types (updates RFCs1034 and 1035)
1112		Host Extensions for IP multicast
1116		Telnet Line Mode Option

Table A–1: RFC Standards (cont.)

RFC	Protocol	Name
1119	NTP	Network Time Protocol minus authentication
1122		Requirements for Internet Hosts – Communication Layers
1123		Requirements for Internet Hosts – Application and Support
1144	CSLIP	Compressing TCP/IP Headers for Low-Speed Serial Links
1155	SMI	Structure of Management Information
1156	MIB	Management Information Base
1157	SNMP	Simple Network Management Protocol
1163	IP-FDDI	A Border Gateway Protocol (BGP) (obsoletes RFC 1105)
1164		Application of the Border Gateway Protocol in the Internet, and OSPF/BGP Interaction
1183	IP-FDDI	New DNS RR Definitions (updates RFCs1034 and 1035)
1188	IP-FDDI	Transmission of IP over FDDI (obsoletes RFC 1103)
1191		Path MTU Discovery (router specification, host specification)
1212		Concise MIB definitions
1213	MIB-II	Management Information Base II (obsoletes RFC 1158)
1231		IEEE 802.5 Token Ring MIB (set operations are not supported)
1253		OSPF Version 2 Management Information Base (obsoletes RFC 1252)
1256	ICMP	Router Discovery Messages
1267		A Border Gateway Protocol 3 (BGP-3) (obsoletes RFCs 1105 and 1163)
1269		Definitions of Managed Objects for the Border Gateway Protocol (Version 3), AS path pattern matching
1282		BSD rlogin
1285		FDDI Management Information Base (set operations are not supported)
1288	FINGER	Finger Protocol (obsoletes RFCs 1196, 1194, and 742
1305	NTP	Network Time Protocol Version 3.0
1321	MD5	The MD5 Message Digest Algorithm

Table A–1: RFC Standards (cont.)

RFC	Protocol	Name
1323	TCP-HIPER	TCP Extensions for High Peformance (Window scale option, time stamp option, and PAWS)
1332	IPCP	The PPP Internet Protocol Control Protocol (obsoletes RFC 1172)
1334	PAP/CHAP	PPP Authentication Protocols
1350	TFTP	Trivial File Transfer Protocol (obsoletes RFC 783)
1388	RIP	RIP Version 2 Carrying Additional Information
1398		Definitions of Managed Objects for the Ethernet-like Interface Types (obsoletes RFC 1284)
1403		BGP OSPF Interaction (obsoletes RFC 1364)
1483		Multiprotocol Encapsulation over ATM Adaptation Layer 5 (routed protocol encapsulation only)
1514		Host Resources MIB (set operations are not supported)
1518	CIDR	An architecture for IP Address Allocation with CIDR
1521		MIME support as stated in Appendix A or this RFC
1533	DHCP	DHCP Options and BOOTP Vendor Extensions (obsoletes RFCs 1497, 1395, 1084, and 1048)
1534		Interoperation between DHCP and BOOTP
1535		A Security Problem and Proposed Correction With Widely Deployed DNS Software
1536		Common DNS Implementation Errors and Suggested Fixes
1542		Clarifications and Extensions for the Bootstrap Protocol (obsoletes RFC 1532; updates RFC 951)
1547	IS-PPP	Requirements for an Internet Standard Point-to-Point Protocol
1571		Telnet Environment Option Interoperability Issues (updates RFC 1408)
1572		Telnet Environment Option
1577		Classical IP over ATM
1583	OSPF	OSPF V2 (obsoletes RFC 1247)
1589		A Kernel Model for Precision Time-keeping (The support to discipline the system clock to an external precision timing source is not supported.)
1591		Domain Name System Structure and Delegation

Table A–1: RFC Standards (cont.)

RFC	Protocol	Name
1597		Address Allocation for Private Internets
1626		Default MTU for IP over ATM
1627		Network 10 Considered Harmful
1633		Integrated Services
1637		DNS NSAP Resource Records (obsoletes RFC 1348)
1652	SMTP	Service Extension for 8-bit MIME transport
1654	PPP	A Border Gateway Protocol 4 (BGP-4)
1661	PPP	The Point-to-Point Protocol (PPP) (obsoletes RFC 1548) (asynchronous IP only)
1700		Assigned Numbers (obsoletes RFC 1340)
1713		Tools for DNS debugging
1748		Token Ring interface information (obsolets RFC 1743)
1755		Signaling for IP of ATM
1794		DNS Support for Load Balancing
1813	NFS	Network File System Version 3 Protocol
1869	SMTP	Service Extensions
1870	SMTP	Service Extensionsfor Message Size Declaration (obsoletes RFC 1653)
1876		A Means for Expressing Location Information in the Domain Name System (updates RFCs 1034 and 1035)
1884		IP Version 6 Addressing Architecture
1886		DNS Extensions to Support IP Version 6
1891	SMTP	Service Extension for Delivery Status Notification
1892		Multipart/Report Content Type for the Reporting of Mail System Administrative Messages
1893		Enhanced Mail System Status Codes
1894		Extensible Message Format for Delivery Status Notifications
1912		Common DNS Operational and Configuration Errors (obsoletes RFC 1537)
1939	POP3	Post Office Protocol, Version 3 (obsoletes RFC 1725)
1953	IFMP	Ipsilon Flow Management Protocol Specification for IPv4

Table A–1: RFC Standards (cont.)

RFC	Protocol	Name
1954		Transmission of Flow Labeled IPv4 on ATM Data Links
1981		Path MTU Discovery for IPv6
1985	SMTP	Service Extension for Remote Queue Starting
1995		Incremental Zone Transfer in DNS (updates RFC 1035)
1996		A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY) (updates RFC 1035)
2001		TCP Slow Start, Congestion Avoidance, Fast Retransmit, Fast Recovery Algorithms
2010		Operational Criteria for Root Name Servers
2018		TCP Selective Acknowledgement Options
2052		A DNS RR for specifying the location of services (DNS SRV) (updates RFCs 1035 and 1183)
2060	IMAP4	Internet Message Protocol, Version 4 Rev. 1 (obsoletes RFC 1730)
2080	RIPng	RIPng for IPv6
2131	DHCP	Dynamic Host Configuration Protocol (obsoletes RFC 1541)
2132	DHCP	DHCP options and BOOTP vendor extensions (obsoletes RFC 1533)
2136		Dynamic Updates in the Domain Name System (DNS UPDATE) (updates RFC 1035)
2137		Secure Domain Name System Dynamic Update (updates RFC 1035)
2181		Clarifications to the DNS Specification (updates RFCs 1034, 1035, and 1123)
2182		Selection and Operation of Secondary DNS Servers
2205	RSVP	Resource Reservation Protocol for FDDI and Ethernet
2210		Use of RSVP with IETF Integrated Services
2211		Controlled Load Services
2308		Negative Caching of DNS Queries (DNS NCACHE) (updates RFCs 1034 and 1035)
2373		IPv6 Addressing Architecture (anycast addressing not currently implemented)
2374		An IPv6 Aggregatable Global Unicast Address Format (obsoletes RFC 2073)

Table A–1: RFC Standards (cont.)

RFC	Protocol	Name
2460	IPv6	Internet Protocol, Version 6 (IPv6) Specification (obsoletes RFC 1883)
2461		Neighbor Discovery for IPv6 (obsoletes RFC 1970)
2462		IPv6 Stateless Address Autoconfiguration (obsoletes RFC 1971)
2463	ICMPv6	Internet Control Message Protocol (ICMPv6) for IPv6 (obsoletes RFC 1885)
2464		Transmission of IPv6 Packets over Ethernet Networks (obsoletes RFC 1972)
2467		Transmission of IPv6 Packets over FDDI Networks (obsoletes RFC 2019)
2471		IPv6 Testing Address Allocation (obsoletes RFC 1897)
2472		IPv6 over PPP (obsoletes RFC 2023)
2535		Domain Name System Security Extensions (obsoletes RFC 2065, updates RFCs , 1035, 2181, and 2534)
2536		DSA KEYs and SIGs in the Domain Name System (DNS)
2537		RSA/MD5 KEYs and SIGs in the Domain Name System (DNS)
2538		Storing Certificates in the Domain Name System (DNS)
2539		Storage of Diffie-Hellman Keys in the Domain Name System (DNS)
2541		DNS Security Operational Considerations
2553		Basic Socket Interface Extensions for IPv6 (obsoletes RFC 2133)
2671		Extension Mechanisms for DNS (EDNS0)
2672		Non-Terminal DNS Name Redirection
2673		Binary Labels in the Domain Name System
2710		Multicast Listener Discovery (MLD) for IPv6 (multicast router not currently implemented)
2741		Agent Extensibility (AgentX) Protocol Version 1 (obsoletes RFC 2257)

Table A–1: RFC Standards (cont.)

RFC	Protocol	Name
2874		DNS Extensions to Support IPv6 Address Aggregation and Renumbering
2893		Transition Mechanisms for IPv6 Hosts and Routers (unidirectional configured tunnels not currently implemented) (obsoletes RFC 1933)

A.2 Non-RFC Standards

The Tru64 UNIX operating system implements the following non-RFC (Request for Comment) standards:

- 4.3BSD and 4.4BSD Socket Interface
- 4.3BSD `inetd`
- 4.3BSD `lpd`
- 4.3BSD `netstat`
- 4.3BSD `ping`
- 4.3BSD `rcp`
- 4.3BSD `rexecd`
- 4.3BSD `rlogin`
- 4.3BSD `rmt`
- 4.3BSD `rsh`
- 4.3BSD `syslog`
- 4.4BSD Sendmail V8.9.3
- UUCP Basic Networking Utilities (HoneyDanBer)
- X/Open Transport Interface (XTI)
- Sun Open Network Computing (ONC) 4.2
- New `rdist` command packaged as optional `nrdisk`
- BSD Packet Data Compression (for PPP)

Glossary of Common UNIX and General Computer Terms

This glossary provides definitions for many of the terms you may see while using the Tru64 UNIX documentation. Although the majority of terms deal with the UNIX environment, you will also find other common terms you will encounter; for example, words related to the Internet.

Special Characters

/

See *root*

. (dot)

A shorthand expression representing the user's working directory.

See also *working directory*

.. (dot-dot)

A shorthand expression representing the immediate parent of the user's working directory.

A

absolute pathname

A pathname that begins at the root directory; a pathname that always begins with a slash (/). For example, `/usr/games` is an absolute pathname. Also called a full pathname.

See also *relative pathname*

active user

In an XTI transport connection, the transport user that initiated the connection.

See also *client process*, *passive user*, *XTI (X/Open Transport Interface)*

Address Resolution Protocol

See *ARP (Address Resolution Protocol)*

alias

A name or symbol used in place of another name, symbol, or group of symbols; usually shorter or easier to use than what it represents. For example, if you often access a certain directory, you could

set up an alias so that the word `work` would be an alias for "`cd /share/tomb/tools/tools/work`". Thereafter, typing `work` would put you in the `/share/tomb/tools/tools/work` directory. For more information see the `alias`(1) reference page.

API (Application Program Interface)

A method prescribed by a specific program (application) or by the computer operating system by which a programmer writing an application program can make requests of the specific program or the operating system.

application

A program or set of programs designed to perform a particular useful function or set of functions; for example, the Source Code Control System (SCCS) is an application for managing program source code.

Application Program Interface

See *API (Application Program Interface)*

apropos

A command that displays the reference page names and summary lines that contain a specified word or string of characters. The `apropos` command is the same as the `man -k` command.

See also *reference page*, *man*

archive

1. To store programs, data files, text files, and other types of files for safekeeping.

2. A repository for such files.

argument count

The number of arguments passed by a command interpreter to a command, or from a routine in a program to a subroutine, procedure, or function.

argument list

The actual information (arguments) passed by a command interpreter to a command, or from a routine in a program to a subroutine, procedure, or function.

ARP (Address Resolution Protocol)

1. The Internet (TCP/IP) Protocol that can dynamically bind a high-level Internet address to a low-level, physical hardware address. ARP can be used only across a single physical network and in networks that support the hardware broadcast feature.

2. The Internet (TCP/IP) Protocol that dynamically maps between Internet addresses, Baseband Adapter addresses, and Token-Ring Adapter addresses on a local area network (LAN).

array
A collection of data elements (variables) identified by a common name and distinguished from one another by numbers representing their positions in the collection. The distinguishing numbers are called subscripts.

assignment statement
A statement that sets a value for a particular field or parameter. In program source files and scripts, assignment statements often have the form *parameter=value.*

ASCII
A standard character set that defines 128 characters (including control and graphic characters). ASCII (American Standard Code for Information Interchange) assigns a 7–bit binary value to each letter, number, and selected control character. The terms *ASCII file* and *text file* are used interchangeably.

asynchronous event
See *event*

asynchronous execution
1. The execution of processes or threads in which each process or thread does not await the completion of the others before starting.
2. In XTI, a mode of execution that notifies the transport user of an event without forcing it to wait.

Asynchronous Transfer Mode
See *ATM (Asynchronous Transfer Mode)*

ATM (Asynchronous Transfer Mode)
A 25 M/bps to 622 M/bps network standard that uses cell switching. It is connection oriented, providing switched, full-duplex communication circuits between nodes.

attribute-value pair
In the key file of a software product kit, a line specifying the name and value for a single attribute of the kit. Controls how the kit is built by the `kits` command and how it is installed by the `setld` utility.

awk
The command for executing programs written in the `awk` programming language. An `awk` program is a sequence of patterns and corresponding actions that are carried out when a pattern is read. The `awk` utility is a more powerful tool for pattern matching and text manipulation than either `grep` or `sed`.

See also *grep*, *sed*

B

background job
See *background process*

background process
A job that runs without interfering with normal command-line entries. A process runs in the background when the command to begin the process is entered with an ampersand (&) character following it. For example, to run the X Window System clock program in background, a user would enter the command `xclock &`. As a result, the clock would be invoked in one window, while the command line on which the `xclock` command was entered would be ready to accept new commands.

See also *foreground process*

Berkeley Internet Name Domain
See *BIND (Berkeley Internet Name Domain)*

Berkeley Software Distribution
See *BSD (Berkeley Software Distribution)*

Berkeley UNIX
See *BSD (Berkeley Software Distribution)*

/bin directory
A directory that contains executable programs and scripts. For example, the `/usr/bin` directory contains programs that nonprivileged users can run, and the `/sbin` directory contains programs that only privileged users can run.

See also *binary file*, *path*, *script*

binary
1. Referring to the number 2 or the system of binary numeration.
2. Referring to an executable file created by a compilation process.
3. Referring to a situation that can assume one of two possible states.

binary file
A file created by a compilation process. Binary files contain codes that are not part of the ASCII character set and utilize all 256 possible byte values. Binary files cannot be read using programs such as `more`, nor can they be edited using editors such as `vi`.

See also *text file*

binary operator
1. A symbol that represents an operation to be performed on two arrays, data items, or expressions. The four types of binary operators are character, logical, numeric, and relational.

2. An arithmetic operator that has two terms.

BIND (Berkeley Internet Name Domain)

A name service available on internet networks.

bit bucket

A term for any receptacle into which data is placed without the possibility of retrieval. It is often used to refer to the null device `/dev/null`.

block device

A data storage or transfer device that manipulates data in groups of a fixed size; for example, a disk, whose data storage size is usually 512 bytes.

See also *character device*

block device switch table

The method used by the Tru64 UNIX operating system to select the entry points associated with a particular block device.

See also *character device switch table*

blocking mode

See *synchronous execution*

block special file

A device special file that provides access to an input or output device and is capable of supporting a file system.

See also *device special file*

Boolean

1. An algebra (named for George Boole) that is similar in form to ordinary algebra, but in which the values of the variables are restricted to the two possible values true and false. The logic of Boolean algebra works well with the binary logic of computers, where values are represented by the digits 0 and 1.

2. A term sometimes used to refer to Boolean operators, including AND, OR, NOT, EXCEPT, IF, THEN, TRUE, and FALSE.

Bourne shell

The command interpreter and interpreted programming language originally developed by Steve Bourne.

See also *shell*

breakpoint

A place in a source code program that stops the debugger during program execution. Breakpoints aid in the testing and debugging of programs.

See also *tracepoint*

break statement

In a programming language, a statement that causes the program to exit immediately from the current control structure (such as a case statement or a for loop). A break statement is often used to terminate execution of a loop before the programmed number of iterations has been performed.

BSD (Berkeley Software Distribution)

The UNIX software release of the Computer System Research Group of the University of California at Berkeley — the basis for some features of the Tru64 UNIX operating system.

BSD socket interface

A transport-layer interface provided for applications to perform interprocess communication between two unrelated processes on a single system or on multiple connected systems. This interprocess communications facility allows programs to use sockets for communications between other programs, protocols, and devices.

built-in

A command that is built into a shell, as opposed to a command that stands alone as a separate executable file and is invoked by a shell.

C

C

A structured, procedural programming language that is widely used both for operating systems and applications and that has a wide following in colleges and universities. The Tru64 UNIX operating system is written in C, which has been standardized as part of the Portable Operating System Interface (POSIX).

See also *C++*, *compiler*, *Java*

C++

An object-oriented programming language that is now generally viewed as the best language for creating large-scale application programs. C++ (pronounced *C plus plus*) is a superset of the C programming language.

See also *compiler*, *Java*

call

In a programming language, a statement that invokes a subroutine, function, or procedure.

call by reference

In a programming language, a method of passing an argument to a subroutine, a function, or a procedure by supplying the address of the data rather than its actual value.

See also *call by value*

call by value

In a programming language, a method of passing an argument to a subroutine, a function, or a procedure by supplying the actual value of the data.

See also *call by reference*

CAM (Common Access Method)

The ANSI standard that defines the software interface between device drivers and the Host Bus Adapters, as well as other means by which SCSI peripherals are attached to a host processor.

See also *SCSI (Small Computer System Interface)*

CAM Control Block

See *CCB (CAM Control Block)*

carriage return

A character that forces all following text to the left margin of the next line or that signals the end of user input. The Return key is usually used to produce a carriage return.

case insensitive

Unable to distinguish between uppercase and lowercase letters. A case-insensitive device or program considers `A` and `a` to be the same character.

See also *case sensitive*

case sensitive

Able to distinguish between uppercase and lowercase letters. A case sensitive device or program considers `A` and `a` to be different characters. Devices and programs that are part of the Tru64 UNIX operating system are case sensitive.

See also *case insensitive*

case statement

In a programming language, a control structure that can take any of several possible paths depending on the evaluation of its argument.

cbreak mode

A terminal driver operation mode that allows processes to read input as it is being typed. This mode eliminates the character, mode, and line editing input facilities.

CCB (CAM Control Block)
The data structure provided by SCSI peripheral drivers to the XPT transport level to control the execution of a function by the SCSI Interface Module (SIM).

CDB (Command Description Block)
A data structure that contains the SCSI operation code, parameters, and control bits for a specific operation.

CDE (Common Desktop Environment)
A graphical user interface for interacting with the Tru64 UNIX operating system. The CDE interface was jointly developed and is based on industry standards, including the X Consortium's X Window System and the Open Software Foundation's Motif interface.

character device
A data storage or transfer device that manipulates data in increments of a single character; for example, a terminal.

See also *block device*

character device switch table
The method used by the Tru64 UNIX operating system to select the entry points associated with a particular character device.

See also *block device switch table*

character special file
A file through which processes can access either a character-stream oriented I/O interface or an unstructured (raw) device, such as a communication line or an unbuffered magnetic tape or disk.

child process
See *parent process*

client
A computer system that uses resources provided by another computer system called a server.

client process
In the client/server model of communication, a process that requests services from a server process.

clist
A data structure used by a BSD-type of terminal driver to store data coming from, or going to, terminals.

See also *STREAMS*

Command Description Block
See *CDB (Command Description Block)*

command history
See *history list*

command interpreter
A program that understands and executes programs written in a particular source language. Interpreted programs execute more slowly than compiled programs because the interpreter is performing two operations at once.

Perl and JavaScript are examples of popular scripting languages that rely on command interpreters. The UNIX shells are command interpreters.

See also *compiler*, *shell*

command interpreter
See *shell*

command mode
A state of a system or device in which the user can enter commands.

command substitution
The ability to capture the output of any command as an argument to another command by placing that command line within grave accents (` `). The shell first executes the command or commands enclosed within the grave accents and then replaces the whole expression, including grave accents, with their output. This feature is often used in assignment statements.

comment out
To selectively disable interpretation of a portion of a program or document source file.

Common Access Method
See *CAM (Common Access Method)*

Common Desktop Environment
See *CDE (Common Desktop Environment)*

common internet address notation
On internet networks, the decimal for the 32-bit internet address. Also called dotted-decimal notation.

communication domain
An abstraction used by the interprocess communication facility of a system to define the properties of a network. Properties include a set of communication protocols, rules for manipulating and interpreting names, and the ability to transmit access rights.

compile
To process one or more program source files in order to produce an executable binary file or an object file.

compiler
A program that translates programs written in a particular source language into executable binary files (or into intermediate binary files referred to as object files). The input can include one or more source-language files together with one or more object files. Compiled programs execute faster than interpreted programs because the compiler has already performed the interpretation. The `cc` program is a C compiler.

See also *binary file*, *compile*, *command interpreter*, *object file*

compile time
Refers to actions that are taken by a compiler during the compilation of a program.

See also *run time*

computer virus
See *virus*

computer worm
See *worm*

concatenate
To place together. Data elements such as strings can be concatenated to produce a string that contains all of the characters of the first original string, followed by the characters of the next original string, and so on. Files can be concatenated by combining their contents in a similar manner, either into a new file or into one of the original files.

conditional compilation
During the compilation of a program, a portion of the process (code block) that is enabled or disabled by a variable or condition external to the code block under consideration. For example, a certain program might contain a block that is to be compiled only if the compilation is performed on a Tru64 UNIX system.

conditional execution
During the execution of a program, a portion of the program's behavior or output that is enabled or disabled by a variable or condition. For example, a certain program might contain code that asks the user questions only if the user initiates the program to run in a menu mode.

conditional statement
In a programming language, a statement (for example, the if statement) that evaluates one or more variables or conditions and uses the result to choose one of several possible paths through the subsequent code.

configuration
1. The machines, devices, and programs that make up a data processing system or network.
2. The act of making a subsystem, or a set of subsystems, available for use by a running operating system.
3. The set of configured subsystems in an operating system.

configuration file
A file that specifies the characteristics of a system or subsystem.

connectionless mode
A mode of service supported by a transport endpoint that requires no established connection for transmitting data. Data is delivered in self-contained units, called **datagrams**.

connection-oriented mode
A mode of service supported by a transport endpoint for transmitting data over an established connection.

construct
A data structure used for a particular purpose.

context search
See *global search*

control statement
In a programming language, a statement that can cause different actions to ensue, depending on the results of an evaluation or test.

cooked mode
The condition of a device driver in which the driver interprets the data passing through it. For example, a UNIX terminal driver operating in cooked mode translates a Return character from the terminal into a Line Feed character to be passed to the system.

See also *raw mode*

cron
A daemon that executes commands at specified times and dates, according to instructions in the `crontab` file.

See also *daemon*

crontab file
A file that specifies the dates and times at which specified commands are to be executed. The `cron` daemon examines the `crontab` file at specified intervals, and executes the indicated commands at the specified dates and times.

csh
The command that invokes the C shell.

See also *C shell*, *shell*

C shell
A command interpreter and interpreted programming language developed at the University of California at Berkeley; so named because many of its constructs resemble the equivalent C language constructs.

See also *shell*

current directory
See *working directory*

cursor
For video display screens, a symbol that shows the location of keyboard input. The cursor shows the position at which the next character to be displayed will be placed.

See also *pointer*

cursor movement keys
A set of keys, usually labeled with arrows pointing up, down, left, and right, that position the cursor on a video display screen.

D

daemon
A process that performs a system management function that is transparent to the user. A daemon can perform its task automatically or periodically. For example, the `cron` daemon periodically performs the tasks listed in the `crontab` file. Daemons can be generated by the system and by applications. Some daemons can also be started manually; for example, the `binlogd` command starts a daemon that logs binary event records to specified files. The commands that manually start daemons usually end with a `d`.

data communications
The transmission of information between computers by means of a network such as an Ethernet, a telephone system, or a satellite link.

datagram
A unit of data that is transmitted across a network by the connectionless service of a transport provider. In addition to user data, a datagram includes the information needed for its delivery. It is self-contained, in that it has no relationship to any datagrams previously or subsequently transmitted.

datagram socket
A socket that provides datagrams consisting of individual messages for transmission in connectionless mode.

Dataless Management Services
See *DMS (Dataless Management Services)*

dbxd
The command that invokes the `dbx` program, which is used by developers to help debug other programs under development.

DCE (Distributed Computing Environment)
A de facto standard for distributed computing that defines a uniform set of services that share certain global properties for common naming, security, time synchronization, system availability, access to data, and system management. DCE enables applications and data on heterogeneous systems to work together.

delta
In an RCS or SCCS file, the set of changes that constitute a specific version of the file.

dependency file
See *dependent*

dependency subset
The condition in which a subset may or may not require the presence of other subsets in order to function properly. Evaluated by a subset's software control program (SCP) under control of the `setld` utility.

See also *SCP (Subset Control Program)*, *subset*

dependent
Also called a **dependency file**. In the `make` utility, an entity on which a file to be built (the target) depends. A source file is a dependent of an object module.

detached job
A job that continues processing after the user has logged out.

device driver
The software that controls a peripheral device such as a disk or a printer.

device special file
A file used by processes to access hardware devices. For example, a printer is accessed through a device special file.

See also *block special file*

DFS (Distributed File System)
A distributed DCE application that provides a unified, globally distributed file system. Under this file system, a DFS file is accessible from any DCE DFS machine using the same name, regardless of the server currently storing the file.

DHCP (Dynamic Host Configuration Protocol)
An Internet (TCP/IP) Protocol that enables the automatic assignment of Internet addresses to clients on the network from a pool of reusable addresses. Address assignment occurs automatically whenever client systems such as portable computers are attached to the network.

directory
A type of file containing the names and controlling information for other files or other directories.

directory hierarchy
The arrangement of directories in a file system. The root directory is at the top of the directory hierarchy and contains pointers to all file systems and all directories on the system.

directory stack
A data structure that stores directories for later recall.

disk label
The disk information, usually located in sector 0 (zero), that includes the disk geometry and partition divisions. This information is used by the system disk driver and the boot program to identify a drive, and to determine how to program a drive and where to find the file systems.

See also *geometry*, *partition*

disk partition
See *partition*

Distributed Computing Environment
See *DCE (Distributed Computing Environment)*

Distributed File System
See *DFS (Distributed File System)*

DMS (Dataless Management Services)
A service provided by Compaqwhereby a server computer system maintains the `root`, `/usr`, and `/var` file systems for client computer systems connected to the server via a local area network (LAN).

DMS area
A reserved disk area that is physically connected to a DMS server and that contains multiple copies of the DMS root area, one for each DMS client.

DMS client
A computer system whose system disk area is physically connected to a DMS server rather than to the client itself and is accessed across the network by the client.

domain
See *domain name system*

domain name system
A tree-structured system for organizing hosts names for an entire internet.

See also *communication domain*, *Internet domain name system*

down time
The period during which a machine is unavailable for use.

See also *up time*

dupatch
A utility included in a patch kit that installs, removes, and manages patches for Tru64 UNIX and the TruCluster products. This utility is installed and left on the system through the successful installation of a Tru64 UNIX patch kit.

Dynamic Host Configuration Protocol
See *DHCP (Dynamic Host Configuration Protocol)*

E

ed editor
A line-oriented program for modifying the contents of text files. The program operates by accepting commands from the user; for example, issuing the command `s/Unix/UNIX/g` would cause the editor to replace each instance of the string “Unix” on the current line with “UNIX.”

editor
A program for modifying the contents of text files. Full-screen editors, such as `vi`, use video display terminals to display several lines of the file being manipulated; they allow the user to move the cursor to a specific location and change the text there. Line editors, such as `ed`, work on a line-by-line

basis. Stream editors, such as `sed`, work by applying commands from a previously prepared list (called a script) instead of by accepting commands from the user.

effective root directory

The point where a system starts when searching for a file. Its pathname begins with a slash (/).

effective user ID

The current user ID, but not necessarily the user's ID. For example, a user logged in under a login ID may change to another user's ID. The ID to which the user changes becomes the effective user ID until the user switches back to the original login ID.

EGP (External Gateway Protocol)

A type of routing protocol that allows individual networks to communicate with the Internet backbone.

See also *Internet*

Emacs

A text editor developed by the Free Software Foundation that is available for all UNIX systems, although it is not a standard part of Berkeley UNIX or System V. It is included with the Tru64 UNIX operating system.

email (electronic mail)

A system that allows the exchange of written messages with other users over a network.

environment

The set of conditions under which a user is working on the computer. The environment includes such information as the name of the working directory, the name of the command interpreter, the identity of the user's terminal, and so on.

environment variable

A symbol containing information that can be used by shells or commands. Environment variables are available to all processes in a given process group; they are propagated by the creation of a child process.

See also *process variable*

EOF (end of file)

1. A condition indicating that the end of a data file has been reached by a program reading the file.

2. A specific sequence of characters written on a magnetic tape.

See also *file mark*

equivalence class
A grouping of characters or character strings that are considered equal for purposes of collation. For example, many languages place an uppercase character in the same equivalence class as its lowercase form, but some languages distinguish between accented and unaccented character forms for the purpose of collation.

error
Any condition in which the expected results of an operation are not achieved. In XTI, an indicator that is returned by a function when it encounters a system or library error in the process of executing. The object is to allow applications to take an action based on the returned error code.

escape
1. To protect a character from interpretation by a program by preceding it with a backslash (`\`).
2. An ASCII character that is usually interpreted as a command to cease a certain activity or as the initial character of a sequence that performs a special function. Cursor control sequences for many terminals and workstations use the escape character.

See also *quote*

/etc
A catchall directory, which usually contains miscellaneous system data files (such as `termcap`, the terminal capabilities database).

Ethernet
A communications concept for local communication networks that interconnects different kinds of computers, information processing products, and office equipment. It is a 10-megabit-per-second baseband local area network (LAN) using carrier sense multiple access with collision detection (CSMA/CD). The network allows multiple stations to access the medium at will without prior coordination, and avoids contention by using carrier sense and deference, and detection and transmission.

event
An occurrence, or happening, that is significant to a transport user. Events are asynchronous, in that they do not happen as a result of an action taken by the user.

event executable image
An executable image located in physical memory.

event management
A mechanism by which transport providers notify transport users of the occurrence of significant events.

executable file
A data file created by a compiler that contains program information a computer can read, interpret, and execute. Also called an image or a binary file.

ex editor
A line-oriented program for modifying the contents of text files. The `ex` editor is an extended version of the `ed` editor.

expedited data
Data that is considered urgent. The semantics of this data are defined by the transport provider.

expression
1. A representation of a value; for example, variables and constants appearing alone or in combination with operators.
2. In programming languages, a language construct for computing a value from one or more operands, such as literals, identifiers, array references, and function calls.
3. A configuration of signs.

extended character
A character other than a 7-bit ASCII character. An extended character can be a 1-byte code point with the eighth bit set (ordinal 128-255).

External Gateway Protocol
See *EGP (External Gateway Protocol)*

F

Fibre Channel
A technology that provides the means for moving data between computer devices at as much as three times the speed of SCSI (currently up to a billion bits per second, with even faster speeds expected). Fibre Channel, which works using optical fiber, coaxial cable, and twisted pair, is specified by a set of standards, specifically the Fibre Channel Physical and Signalling standard, ANSI X3.230-1994, which is also ISO 14165-1.

field
1. The basic unit of information in a record.
2. In `awk`, one element of an input record.

See also *record*

field separator
One or more characters used to separate fields in a record.

file descriptor
A small unsigned integer that a UNIX system uses to identify a file. A file descriptor is created by a process through issuing an open system call for the file name. A file descriptor ceases to exist when it is no longer held by any process.

file mark
A sequence of characters written on a magnetic tape to signify the end of a data file.

See also *EOF (end of file)*

file name expansion
See *globbing*

file pointer
An identifier that indicates a structure containing the file name.

file system
The collection of files and file management structures on a physical or logical mass storage device.

filter
1. A command that reads standard input data, modifies the data, and sends it to standard output.
2. A device or program that separates data, signals, or materials in accordance with specific criteria.

flag
See *option*

foreground job
See *foreground process*

foreground process
A job that must be completed or interrupted before the shell will accept more commands; a job receiving input from a workstation or terminal.

See also *background process*

fork
1. The command used to create and start a child process.
2. The result of using the `fork` command.

See also *parent process*

full installation
A Tru64 UNIX installation that creates new file systems and loads a full copy of the operating system from the kit onto a system. Any other version of

the operating system, any layered products, and any patches that previously existed on the system are overwritten. A full installation does not preserve system customizations (for example, user or data files) because the root (/), /usr, and /var file systems are re-created during the process.

See also *update installation*

full pathname
See *absolute pathname*

full-screen editor
An editor that displays an entire screen at a time. Also called a visual editor.

See also *line editor*

G

geometry
The sizes (in bytes) of cylinders, tracks, and sectors for a particular disk device.

See also *disk label*

gid, GID
See *group ID (GID)*

global
In programming languages, pertaining to information defined in one subdivision of a program and used in at least one other subdivision of the program; pertaining to information available to more than one program or subroutine.

global character
See *wildcard character*

global search
In an editing environment, the process of having the system look through a document for specific characters, words, or groups of characters.

globbing
A UNIX term for the shell's process of wildcard file name expansion to develop a list of literal file names that the shell then passes to a command. The C shell permits the user to disable globbing by default; the Bourne, Korn, and POSIX shells require the user to quote or escape metacharacters in file names if globbing is not desired.

grep
The command that invokes the grep program, which is used to search specified files for lines containing characters that match specified patterns,

and then writes those matching lines to standard output. The name means Global Regular Expression Printer.

See also *regular expression*

group

1. A collection of users who can share access authorities for protected resources.

2. A list of names that are known together by a single name.

3. A set of related records that have the same value for a particular field in all records.

4. A series of records logically joined together.

See also *login group*

group ID (GID)

A unique number assigned to a group of related users. The group number can often be substituted in commands that take a group name as an argument.

See also *user ID (UID)*, *process ID (PID)*

H

hard link

1. A mechanism that allows the `ln` command to assign more than one name to a file. Both the new name and the file being linked must be in the same file system.

2. The default result of using the `ln` command.

See also *symbolic link*

hashed passwd database

An indexed database containing the contents of the `passwd` file. The indexed database minimizes the search time needed to retrieve information.

hashing

A method of transforming a search key into an address for the purpose of storing and retrieving items of data.

HBA (host bus adapter)

The hardware and microcode that provides the interface between system memory and a Small Computer System Interface (SCSI) bus.

header file

See *include file*

hidden character
A character in the ASCII character set that is not printable; for example, the DEL and ESC characters.

hidden file
A file whose name begins with a period. by default, the `ls` command omits such files from its listings.

history
In the C shell and the Korn shell, a command that displays the user's history list.

history list
In the C shell and the Korn shell, a listing of the most recent commands entered by the user. Commands in the history list are available for recall, modification, and reexecution.

$HOME
An environment variable containing the absolute pathname of the user's home directory.

See also *$home*, *environment variable*

$home
A process variable containing the absolute pathname of the user's home directory.

See also *$HOME*, *process variable*

home directory
A directory that is owned by a specific user and from which that user's other directories descend in a hierarchy. Also known as a **login directory.**

See also *working directory*

host
1. The primary or controlling computer in a communications network.
2. A computer attached to a network.

Host Bus Adapter
See *HBA (host bus adapter)*

host name
The name given to a computer on the network.

HTML (HyperText Markup Language)
The coding (markup) inserted in a file intended for display on a World Wide Web browser that tells the browser how to display a Web page's words. The markup is done with *tags*, which are command words enclosed in angle

brackets. For example, the tag **<P>** creates a new paragraph; the tag **<TABLE>** begins the formatting of a table. Although the World Wide Web Consortium (W3C) promotes the standardization of HTML, both Netscape and Microsoft browsers currently implement some features differently and provide nonstandard extensions.

See also *PDF file*

HyperText Markup Language

See *HTML (HyperText Markup Language)*

I

I18N

See *internationalization*

ICMP (Internet Control Message Protocol)

A host-to-host protocol from the Internet Protocol suite that controls errors and the operations of the Internet Protocol (IP).

See also *IP (Internet Protocol)*

#include

A C language precompiler directive specifying interpolation of a named file into the file being compiled. The interpolated file is a standard header file (indicated by placing its name in angle brackets) or any other file containing C language code (indicated by placing its name in double quotation marks). For example:

```
#include <header_file.h>
#include "myfile.c"
```

The absolute pathname of header files whose names are placed in angle brackets (<>) is `/usr/include/`*`file.h`*.

See also *include file*

include file

A text file that contains declarations used by a group of functions, programs, or users. Also known as a header file.

See also *#include*

incremental backup

The process of copying files that have been opened for reasons other than read-only access since the last backup was created and that meet the backup frequency criteria.

infinite loop
A source code error that causes the program to continually repeat the same set of instructions. For example, Instruction A sends the program execution to Instruction B, which in turn sends the program execution back to instruction A. Such a loop can only be interrupted by intervention from outside the program.

init
The command given by a UNIX system as the final step in the boot procedure after the root file system is mounted. The `init` program initializes the system by creating and controlling processes, which are defined in the `inittab` file.

init process
A process created by the system that performs system administration tasks, such as spawning login processes and handling the orderly shutdown from multiuser to single-user mode.

inline editing
A feature of some shells that allows users to edit a current or previously entered command line.

inode
The internal structure that describes the individual files in the operating system. There is one inode for each file. An inode contains the node, type, owner, and location of a file. A table of inodes is stored near the beginning of a file system.

inode number
A number specifying a particular inode file in the file system.

input
Data to be processed.

input redirection
The specification of an input source other than standard input.

instruction
The part of a computer program that tells the computer what function to perform at that stage.

internationalization
The capability of a computer program of making itself adaptable to the requirements of different native languages, local customs, and coded character sets, which means, essentially, that internationalized programs can run in any supported **locale** without having to be modified.

The term internationalization is often represented by the abbreviation I18N, with 18 representing the number of letters between the first and last letters of the word.

International Standards Organization

See *ISO (International Standards Organization)*

Internet

1. A collection of computing networks consisting of participants from major research institutions, universities, and government labs, including the National Science Foundation (NSF) and the NFSnet regional organizations. The Internet is not a commercial product, but rather a large project in support of research.

2. A collection of connected networks using the TCP/IP protocols.

internet address

A unique 32-bit number that identifies a host's connection to an internet network. An internet address consists of a network number and a host number.

Internet Control Message Protocol

See *ICMP (Internet Control Message Protocol)*

Internet domain name system

The domain name system of the Internet, which consists of the following categories of hosts: COM, EDU, GOV, MIL, NET, ORG, and ARPA.

See also *communication domain*, *domain name system*, *Internet*

Internet Protocol

See *IP (Internet Protocol)*

interrupt

1. An event that causes a computer to digress from its normal processing stream in order to respond to the condition that triggered the digression. Upon completion of the digression, the normal processing stream is resumed at the point of interruption. Interrupts can be caused either by software instructions or by hardware events such as the completion of an I/O operation.

2. To trigger an interrupt.

interrupt handler

Code in a program or operating system that performs actions in response to an interrupt.

IP (Internet Protocol)

The network layer protocol for the Internet protocol suite that provides the basis for the connectionless, best-effort packet delivery service. IP includes

the Internet Control Message Protocol (ICMP) as an integral part. The Internet protocol suite is referred to as TCP/IP because IP is one of the two most fundamental protocols.

IP gateway
See *IP router*

IP router
A host that connects two or more internet networks. The IP router knows how to reach all the hosts on the networks to which it is attached. Also known as an IP gateway.

ISO (International Standards Organization)
An international body composed of the national standards organizations of 89 countries. ISO issues standards on a vast number of goods and services, including networking software.

iterate
To perform the same function repeatedly on different data, often with the object of arriving at a result by successively closer approximation.

J

Java
A programming language developed by Sun Microsystems that is based on C++ but optimized for the distribution of program objects in a network such as the Internet. Java is somewhat simpler and easier to learn than C++ and has characteristics that give it other advantages over C++.

A Java applet is a small program, such as a window or a pull-down menu, written in Java.

JavaScript
An interpreted programming (script language) from Netscape. It is somewhat similar in capability to Microsoft's Visual Basic, Sun's Tcl, the UNIX-derived Perl, and IBM's REXX. In general, script languages are easier and faster to code in than the more structured and compiled languages such as C and C++. Script languages generally take longer to process than compiled languages, but are very useful for shorter programs.

job
1. A unit of work defined by a user to be done by a system. The term *job* sometimes refers to a representation of the job, such as a set of programs, files, and control statements to the operating system.
2. One or more related procedures or programs grouped into a procedure, identified by appropriate job control statements.

job control
Facilities for monitoring and accessing background processes.

job number
A number assigned to a job as it enters the system to distinguish the job from other jobs.

job queue
A list of the jobs that are waiting to be processed by the system.

job state
The status of the work being done by a system.

K

kdbx
The command that invokes the `kdbx` program, an interactive crash analysis and kernel debugging tool. The `kdbx` program serves as a front end to the `dbx` debugger.

kdebug program
A program that lets programmers control the execution of a running kernel.

kernel
The integral part of the operating system that provides fundamental services to users and applications. Traditional kernels do the following:

- Dispense such system resources as memory and execution time to tasks
- Provide input and output services that allow tasks to access mass storage and other devices (and users and the computer to engage in a dialogue through a monitor and keyboard)
- Manage communications to other computers residing on a network
- Provide authentication services for users and tasks
- Manage the organization and integrity of file system

The kernel has specialized knowledge of the hardware on which it runs.

See also *shell*, *microkernel*

keyword
1. A word that must be matched when retrieving information.
2. A reserved word whose presence is required in a file.

kill
1. To stop the operation of a process. In most cases, a user can kill a foreground process by pressing Ctrl/c.

2. The Tru64 UNIX command that a user can issue to stop a background or suspended process. A superuser can use this command to stop any process on the system.

Korn shell

A command interpreter and interpreted programming language developed by David Korn. The Korn shell (`ksh`) is semantically an extended version of the **Bourne shell**, with constructs and commands to implement enhanced features, including job control and command history recall. The **POSIX** shell is a superset of the Korn shell.

See also *shell*

L

L10N

See *localization*

label

See *disk label*

LAN (Local Area Network)

A device communications system that operates over a limited physical distance, offering high-speed communications channels optimized for connecting information-processing equipment.

LAPD (Lightweight Directory Access Protocol)

An Internet standard directory service protocol that runs over TCP/IP. An LDAP server manages entries in a directory, and makes the information available to users and applications across the network; it can be used as a central repository of user information to identify and authenticate individuals. When used in this way, an LDAP server is similar to Network Information Services (NIS).

See also *LAN (Local Area Network)*

LAT (Local Area Transport)

A protocol that supports communications between host computer systems and terminal servers with terminals, PCs, printers, modems, and other devices over local area networks (LANs).

See also *LAN (Local Area Network)*

layered product

An optional software product designed to be installed as an added feature of the Tru64 UNIX system.

lex
The command that invokes the Lexical Analyzer Generator, a program for generating other programs that can organize input into units of meaning (symbols) called lexemes.

See also *lexical analyzer*, *parser*, *yacc (Yet Another Compiler-Compiler)*

lexical analyzer
A program or program fragment for analyzing input and assigning elements of it to categories to assist in parsing the input. The `lex` program assists in the creation of lexical analyzers.

See also *parser*

Lexical Analyzer Generator
See *lex*

line editor
An interactive or noninteractive text editor that works on one line of text at a time.

See also *full-screen editor*

link
A directory entry referring to a file.

See also *hard link*, *symbolic link*

linking loader
A single program that loads, relocates, and links compiled and assembled programs, routines, and subroutines to create an executable file. Also known as link loader and linker loader.

lint
A program that checks C code for bugs, portability problems, and errors, such as mismatched argument types and uninitialized variables.

literal
1. A value expression representing a constant.
2. A specific symbol that cannot be modified during the translation of a program.

local area network
See *LAN (Local Area Network)*

local area transport
See *LAT (Local Area Transport)*

locale
A software environment that correctly handles the cultural conventions of a particular geographic area, such as China or France, and a language as it is used in that area. So by selecting a Chinese locale, for example, all commands, system messages, and keystrokes can be in Chinese characters and displayed in a way appropriate for Chinese.

See also *internationalization*

local host
The computer system to which a user's terminal is directly connected.

localization
The process of implementing local requirements within a computer system. Some of these requirements are addressed by locales. Each locale is a set of data that supports a particular combination of native language, cultural data, and codeset. The type of information a locale can contain and the interfaces that use a locale are subject to standardization.

The term "localization" is sometimes abbreviated as L10N, with 10 representing the number of letters between the first and last letters of the word.

See also *internationalization*

lock file
A file that indicates that operations on one or more other files are restricted or prohibited. The presence of the lock file can be used as the indication, or the lock file can contain information describing the nature of the restrictions. For example, the Tru64 UNIX `setld` utility creates a lock file for each product kit subset that it installs. If a given product includes subsets that require the presence of a previously installed subset, `setld` places in the earlier subset's lock file the names of the later subsets to prevent inadvertent deletion of the earlier subset.

logical unit number
See *LUN (Logical Unit Number)*

log in
To begin using a computer system, usually by entering one's login name and a password to gain access to and communicate with the operating system as an authorized user.

login directory
See *home directory*

login group
The primary classification that establishes the access permission for the files created by the user.

See also *group*

login name
The name that identifies a user to a computer system and to other users of the system. When logging in to the system, the user enters this name and (usually) a password. Also known as user name.

login shell
The shell that a user uses by default upon logging in to the system. It is specified by the user's entry in the `passwd` file.

log out, log off
To stop using a computer system, usually by entering a command that tells the operating system that the user is ending the current session.

loop
1. A sequence of instructions that is executed repeatedly until a specified condition is satisfied.
2. In the Tru64 UNIX virtual memory system, the page clusters in main memory that are repeatedly scanned for replacement.

See also *infinite loop*

LUN (Logical Unit Number)
The number assigned to a SCSI device to identify it to the SCSI driver.

M

macro
An instruction written as part of a source language, which when compiled into machine code will generate several machine code instructions.

See also *instruction*

mailbox
A file that contains new and unread mail messages. The mailbox file is usually in the `/usr/spool/mail` directory.

MAKDEV
A script that creates device special files for the devices on a Tru64 UNIX system. This script resides in the `/dev` directory.

make
A tool that builds programs and applications by testing to see whether the source files that produce a given application are newer than the target files produced from them. If any source or intermediate file is newer than its target, `make` performs the actions necessary to rebuild the target file by

following a set of rules. The rules can be standard (specified by default) or they can be explicit descriptions of the steps required.

makefile
The specification file used by the `make` tool. The makefile specifies the names of target programs and describes rules for their creation.

See also *make*

man
The command that displays reference pages on line; the name is a short form of *manual*.

See also *apropos*, *reference page*

Management Information Base
See *MIB (Management Information Base)*

manpage, manual page
See *reference page*

MANPATH
An environment variable whose value provides the default directory search path used by the `man, catman,` and `xman` commands.

See also *search path*

Memory File System
See *MFS (Memory File System)*

memory trolling
A process of reading a system's memory to proactively discover and handle memory errors.

metacharacter
A character that is interpreted by a computer system to mean something other than its obvious meaning. For example, the asterisk is often used to allow wildcard matching in file names.

MFS (Memory File System)
A UFS file system that resides only in memory. No permanent data or file structures are written to disk. An MFS can improve read/write performance, but it is a volatile cache. The contents of an MFS are lost after a reboot, unmount operation, or power failure.

MIB (Management Information Base)
The Management Information Base defines a set of data elements that relate to network management. Many of these are standardized in the RFCs that are produced as a result of the Internet Engineering Task Force working group standardization effort of the Internet Society.

microkernel
A type of **kernel** that delegates much of the work of memory management, task scheduling, and other services to nonkernel code. A microkernel provides only the basic primitives that control such operations on a specific processor architectures.

middleware
A term used to describe software that enables two other programs to work together. An example of middleware is the COM for Tru64 UNIX software.

mirroring
A way of duplicating information on a disk to ensure that the information is still available in the event of a disk failure.

See also *RAID (redundant array of independent disks)*, *striping*

mode
The set of permissions for a file. These permissions are often expressed as three numbers, which represent an octal notation to set each bit in the permission code. Users who are given or denied permissions are "owner," "group," and "other"; the most common permissions given are read, write, and execute. The command used to change permissions is `chmod`.

Motif
See *OSF/Motif*

mount
A command used to make a file system available.

See also *unmount*

mount point
A directory file that is the name of a mounted file system.

multiprocessor
A system with two or more processors sharing common physical memory.

N

name service
The service provided to client processes for identifying peer processes for communications purposes.

native software
Software that is written in a language that compiles either to assembly language or directly to the computer's standard machine representation (object files). Native software is more efficient and runs much faster than

translated or interpreted software; in addition, it can be tailored to make the most effective use of the machine's resources.

NetRAIN (Redundant Array of Network Adaptors)

An interface that provides a mechanism to protect against certain kinds of network connectivity failures. NetRAIN integrates multiple network interfaces on the same local area network (LAN) segment into a single virtual interface called a NetRAIN set. One network interface in the set is always active while the others remain idle. If the active interface fails, one of the idle set members comes on line with the same IP address within an adjustable failover time period.

network

Two or more computing systems that are linked for the purpose of exchanging information and sharing resources.

Network File System

See *NFS (Network File System)*

NFS (Network File System)

A service that allows a system (the server) to make file systems available across a network for mounting on other systems (clients). When a client mounts an NFS file system, the client's users see the file system as if it were local to the client.

NFS-mounted

Refers to a file system that is mounted over a network via NFS rather than being physically connected (local) to the system on which it is mounted.

See also *NFS (Network File System)*

NIS (Network Information Service)

A distributed data lookup service for sharing information on a local area network (LAN). NIS allows you to coordinate the distribution of database information throughout your networked environment. NIS was formerly known as Yellow Pages.

See also *NFS (Network File System)*

nonblocking mode

See *asynchronous execution*

(NUMA) non-uniform memory access

One of the two supported architectures for multiprocessor systems, the other being symmetric multiprocessors (SMP). The NUMA architecture overcomes a drawback of traditional SMP systems that provide one interconnect, either a bus or a switch, that links all system resources. With NUMA, systems can be scaled to large numbers of CPUs without the system switch or bus becomming a performance bottleneck.

nroff

The command that calls the `nroff` program, a member of the *roff* family of text formatters. The `nroff` program produces ASCII output suitable for display or printing on character-cell devices such as terminals and printers.

O

object file

A nonexecutable intermediate binary file created by a compiler. Object files are frequently used as libraries, to provide precompiled program elements for use in compiling a complete executable binary.

See also *binary file*, *compiler*

octal

A number system that uses 8 as a base (radix). The octal system uses the digits 0 through 7, and each digit position represents a power of 8.

OLAR (online addition and replacement

A system management process that is used to expand capacity, upgrade components, and replace failed components, while the operating system services and applications continue to run. This functionality (sometimes referred to as "hot-swap") provides the benefits of increased system uptime and availability during both scheduled and unscheduled maintenance.

open system

A system that supports the International Organization for Standardization (ISO) Reference Model for Open System Interconnection (OSI).

Open Systems Interconnection

See *OSI (Open Systems Interconnection)*

operator

In regular expressions, a character that is interpreted to mean something other than its literal meaning. For example, a pair of brackets (`[]`) form an operator that enables a single-character match on any one of the characters enclosed by the brackets.

optimization

The process of selecting the specific method by which a program is to perform a given task such that the most effective use is made of time, I/O, or other resources.

option

1. An argument that controls how the shell executes a command. Options are usually preceded by a hyphen and appear with the command name on a command line; for example, `ls -a`. An option is often referred to as a flag or a switch.

2. An indicator or parameter that shows the setting of a switch.

3. A character that signals the occurrence of some condition, such as the end of a word.

4. An internal indicator that describes a condition to the CPU.

OSF (Open Software Foundation)

A consortium of software vendors formed for the purpose of developing and marketing widely compatible UNIX systems based on a common set of features.

OSF/Motif

A graphical user interface developed and licensed by the Open Group (originally by the Open Software Foundation). OSF/Motif is based on the X Window System. Also called *Motif*.

OSI (Open Systems Interconnection)

A set of international standards developed by the International Organization for Standardization. The goal of the OSI is that different vendors' computer systems can interconnect.

owner

Usually, the user who creates a file. The owner has the right to change the list of users or groups who are permitted access to the file and the ways in which those users or groups may access the file. Ownership of a file can be reassigned by the system manager or superuser.

P

package

For the Tru64 UNIX operating system loader, a collection of object entities that share a common name space. Symbol names are unique within a package. Symbols from different packages may bear identical symbol names, because they are distinguished by their package names.

page

A fixed-size unit of physical memory.

PALcode (Privileged Architecture Library)

A set of subroutines that are specific to a particular Alpha operating system implementation. These subroutines provide operating-system primitives for context-switching interrupts, exceptions, and memory management.

parent directory

The directory in which another directory resides. The directory that is contained in the parent is called a subdirectory.

parent process
A process that has created other processes, called its children. In the Tru64 UNIX system, every command that is not a shell built-in command creates a child process.

See also *fork*

parser
A program or program fragment for interpreting input and determining how to act upon it. The `yacc` program assists in the creation of parsers.

See also *lexical analyzer*

parsing order
The sequence in which a program interprets information that is input to it. For example, a program using left-to-right parsing order interprets input reading "create a number; write the number" so that the number created by the first step is written. A program with right-to-left parsing order interprets the same input to mean that the program is to write a number that it created in some previous step and then to create a new number.

partition
A physical portion of a disk. Disks are divided into partitions that are then assigned to hold various file systems. For example, the root file system is usually on the first partition, named `a`. The `/usr` file system is on a different partition, often the `g` partition. The use of partitions provides flexibility and control of disk usage, but it is restricted in that it denies unlimited use of all the available space on a given disk for a given file.

See also *disk label*, *geometry*

passive user
In an XTI transport connection, the transport user that did not initiate the connection.

See also *client process*, *active user*, *XTI (X/Open Transport Interface)*

passwd
1. The command by which users change their login password.
2. The UNIX file in which user passwords and associated information are stored; the file's pathname is `/etc/passwd`.

patch
A file or a collection of files that contain fixes to problems. A patch may correct one or more problems. A collection of patches for a software product is sometimes referred to as a **patch kit**.

See also *dupatch*

$PATH
An environment variable containing the user's search path for commands. Directory names in the `$PATH` variable are separated with colons.

See also *$path*

$path
A process variable containing the user's search path for commands. Directory names in the `$path` variable are separated with spaces.

See also *$PATH*

path
An ordered list of the directories in which the shell searches for the executable files named by commands that are not entered with a pathname and are not shell built-in commands.

See also *$PATH*, *$path*

pathname
The name of a file, concatenated onto a list of the directories through which access to that file is achieved; hence, the complete name of the file. Absolute pathnames begin at the root directory and are written with an initial slash (for example, `/usr/users/rolf/myfile.txt`). Relative pathnames begin at the user's working directory and are written without the initial slash (for example, `rolf/myfile.txt`).

pathname qualifier
See *variable modifier*

pattern matching
The process of comparing input information (usually text) against a specified set of symbols (usually regular expressions) to find correspondences.

See also *regular expression*

Perl
A versatile scripting language that is often used to develop CGI (common gateway interface) programs for the World Wide Web, such as the forms you would use to order merchandise. Perl is similar in syntax to the C language and includes a number of popular UNIX facilities such as **sed** and **awk**. Perl is an interpreted language that can be compiled just before execution into either C code or cross-platform byte code. When compiled, a Perl program executes almost as fast as a fully precompiled C language program.

See also *script*

permission code
See *permissions*

permission field
See *permissions*

permissions
The constraints a user places on a file to control what other users or groups may read, write, or execute the file. There are three sets of permissions: those applied to the user, those applied to the user's group, and those applied to everyone else, called "other."

PDF file
A file type recognized by the Adobe Acrobat Reader, which provides an easy way to view and print documentation. As the next generation of Adobe's PostScript format, PDF files have become a standard way of distributing documentation, especially on CD–ROM and over the Internet. Most Tru64 UNIX documentation is provided in PDF and **HTML** formats. The Acrobat Reader is provided on the Tru64 UNIX Documentation CD–ROM of Version 4.0E and higher. It is also available at the Adobe Web site:

http://www.adobe.com/

pid, PID
See *process ID (PID)*

pipe
The construct that couples the output of one program directory to the input of another. Pipes are created by the use of a vertical bar (|) between commands on the command line. For example:

```
% nroff inputfile -ms | lpr
```

This pipeline processes the input file with the `nroff` command and sends the processed file directly to the printer with the `lpr` command.

See also *pipeline*

pipeline
A series of commands connected by pipes. The process of coupling the output of one command directly to the input of another with a pipe is called "pipelining" or "piping".

piping
See *pipeline*

pixel (picture element)
The smallest element of a display in a graphics application. On a video screen, pixels are the dots that produce the visual image. The number of pixels usually determines the resolution of the image; the more pixels, the better the resolution.

Point-to-Point Protocol
See *PPP (Point-to-Point Protocol)*

pointer
A symbol that specifies position by reflecting the motion of the mouse. The pointer can change shape to indicate the function of the area in which the pointer is positioned.

See also *cursor*

POSIX (Portable Operating System Interface)
A collection of standards proposed by the POSIX working groups of the Institute of Electrical and Electronics Engineers (IEEE). POSIX standards define system interfaces to support the source portability of applications.

See also *C*, *SVID (System V Interface Definition)*

POSIX shell
The shell that conforms to the POSIX standard. The POSIX shell (`sh`) is a subset of the **Korn** shell.

See also *shell*

PostScript
A language developed by Adobe Systems, Inc., for specifying the formatting of typeset documents or displays. An encapsulated PostScript file is a file that follows a standard for embedding PostScript files into other PostScript files.

See also *PDF file*

PPP (Point-to-Point Protocol)
A transmission line protocol that encapsulates and transfers IP datagrams over asynchronous serial lines. PPP is more efficient than SLIP.

See also *SLIP (Serial Line Internet Protocol)*

predefined variable
A shell variable defined and maintained by the C shell.

preprocessor
A program that translates some portion of information in a file into a form understandable to another program. For example, the `tbl` program is a preprocessor for the `nroff` text formatter.

printcap database
A file (`/etc/printcap`) containing descriptions of all the printers known to the system.

process ID (PID)
A unique number assigned to a process that is running.

See also *group ID (GID)*, *user ID (UID)*

process identification
See *process ID (PID)*

process table
A kernel data structure that contains relevant information about all processes in the system.

process variable
A symbol containing information that can be used by the current process only. Process variables are not automatically propagated to child processes.

See also *environment variable*

profile data
Information about how a program is spending its execution time.

See also *profiling*

profiling
The monitoring of how system resources are used in a given program. Profiling helps programmers improve the efficiency of their program code.

pseudodevice
A device that consists of a software simulation, rather than hardware; for example, a **pseudoterminal**.

pseudoterminal
A special file that effectively functions as a keyboard and display device. Also called a pty or pseudo-tty.

See also *pseudodevice*

pseudo-tty
See *pseudoterminal*

pty
See *pseudoterminal*

pwd
The command that causes the system to display the absolute pathname of the user's working directory.

See also *working directory*

Q

query
1. The action of searching data for desired information.

2. In data communications, the process by which a master station asks a slave station to identify itself and to give its status.

3. In interactive systems, an operation at a terminal or workstation that elicits a response from the system.

4. A request, based on specific conditions, for information from a file .

queue

A line of items waiting to be processed. For example, a print queue consists of jobs waiting to be printed.

queue daemon

The process that maintains a list of outstanding jobs and sends them to the specified device at the appropriate time.

See also *queue*, *daemon*, *job*

queued message

A system message that is added to a list of messages stored in a file for user viewing at a later time. Background processes usually produce queued messages. Programs interacting directly with users typically send messages to the screen for immediate user viewing.

queue element

An item in a queue.

quote

To protect a character from interpretation by a program by enclosing it in quotation marks or by preceding it with a backslash character (\); to mask the special meaning of certain characters, causing them to be taken literally.

See also *escape*

R

RAID (redundant array of independent disks)

A way of storing the same data in different places (thus, redundantly) on multiple hard disks. By placing data on multiple disks, I/O operations can overlap in a balanced way, thereby improving performance.

RAID technology is most used in storage configurations for high performance and high data availability, using the Logical Storage Manager (LSM) or a hardware-based RAID subsystem.

The four primary RAID levels are RAID 0, also known as data or disk **striping**; RAID 1, also known as data or disk **mirroring**; and RAID 3 and RAID 5, which are types of parity RAID.

raw mode
The condition of a device driver in which the driver does not interpret the data passing through it. For example, a UNIX terminal driver operating in raw mode passes a Return character from the terminal directly to the system.

See also *cooked mode*

raw socket
A socket that provides privileged users access to internal network protocols and interfaces. These sockets can be used to take advantage of protocol features not available through more normal interfaces or to communicate with hardware interfaces.

rc
An element of the name applied to files containing command scripts that control the process of booting a computer. The `rc` characters are also used in the names of files that contain user-customized startup information, such as the BSD mail utility `.mailrc` and the Motif window manager `.mwmrc`.

RCS (Revision Control System)
A set of programs for managing program and documentation source files so that any revision of a given file can be retrieved. Revisions to a file are stored as a series of incremental changes (deltas) applied to the original version instead of as complete copies of all the versions. The system provides locking mechanisms so that only one user can apply changes to a given file at any one time.

See also *SCCS (Source Code Control System)*

RCS file
A file stored in the Revision Control System (RCS) library containing the text of the original file and the list of changes that have been applied to it.

RCS library
The directory in which Revision Control System (RCS) files are stored.

record
1. A collection of related data items treated as a unit. A record contains one or more fields.
2. In `awk`, the information between two consecutive occurrences of the record separator. For most purposes, a record in `awk` can be thought of as a line from the input file.

recursive
In programming, pertaining to a procedure or function that accomplishes its task by repeatedly calling itself until a specified condition is reached. The process of using a recursive procedure or function is called *recursion*.

redirection
The specifying of one or more devices with which standard input, standard output, and standard error virtual files are to be associated during the execution of a given command.

Redundant Array of Network Adaptors
See *NetRAIN (Redundant Array of Network Adaptors)*

reference page
One of a collection of files containing documentation on commands, system calls, library routines, and so forth. Reference pages are often called man pages (short for manual pages).

regular expression
A pattern of one or more characters used to find text information and formed according to a set of rules that define how the characters are to be interpreted. For example, a period is interpreted as a valid match for any character in the input. The regular expression `a.c` matches any string containing the letter `a` and the letter `c` separated by a single intervening character, such as `abc, a?c, a9c,` and so on.

See also *pattern matching*

relative pathname
A pathname that begins at the user's working directory. A relative pathname is written without the initial slash; for example, `docs/myfile.txt` is a relative pathname.

See also *absolute pathname*

Request For Comments
See *RFC (Request For Comments)*

restricted shell
A security feature that provides a controlled shell environment with limited features.

Revision Control System
See *RCS (Revision Control System)*

RFC (Request For Comments)
A formal specification or document that describes a component of the Internet. RFCs are drafted by groups of engineers and submitted for review by their peers. Once approved, an RFC is numbered and is then considered to be a definitive specification. Although numbered RFCs cannot be changed, subsequent RFCs can supersede or elaborate on all or parts of previous RFCs.

RIS (Remote Installation Services)
A utility for installing software kits across a network instead of by using locally mounted distribution media.

RIS area
A reserved disk area physically connected to a RIS server, containing one or more product environments in which are stored installable software kits.

RIS client
A computer system that has permission to install software across the network by accessing kits stored in the server's RIS area.

RIS server
A computer system that serves other computers by providing software kits for them to install. The software is stored on disks belonging to the server and is accessed across the network by the RIS clients.

RISC (Reduced Instruction Set Computing)
A computer architecture that is based on a limited set of simple instructions instead of a larger and more varied set of more complex instructions.

rolling upgrade
A software upgrade of a cluster that is performed while the cluster is in operation. One member at a time is "rolled" and returned to operation while the cluster transparently maintains a mixed-version environment for the base operating system, cluster, and Worldwide Language Support (WSL) software. Clients accessing services are not aware that a rolling upgrade is in progress.

On clustered Tru64 UNIX Version 5.0A and higher systems, you perform a rolling upgrade to update the Tru64 UNIX operating system or TruCluster Server software or to patch the system.

root
1. The login name for the superuser (system administrator).
2. The name applied to the topmost directory in the UNIX system's tree-like file structure; hence, the beginning of an absolute pathname. The root directory is represented in pathnames by an initial slash (/); a reference to the root directory itself consists of a single slash.

See also *pathname*

root directory
See *root*

root file system
The basic file system, onto which all other file systems can be mounted. The root file system contains the operating system files that get the rest of the system running.

root login
See *root*

routing daemon
A program that provides a routing-management service. The routing daemon, `routed`, is invoked when the system is booted to manage the network routing tables.

See also *daemon*

run time
Refers to actions that are taken by a program or system during execution.

See also *compile time*

S

SCCS (Source Code Control System)
A set of programs for managing program and documentation source files so that any revision of a given file can be retrieved. Revisions to a file are stored as a series of incremental changes (deltas) applied to the original version instead of as complete copies of all the versions. The system provides locking mechanisms so that only one user can apply changes to a given file at any one time.

See also *RCS (Revision Control System)*

SCCS library
The directory in which Source Code Control System (SCCS) s-files and p-files are stored.

SCP (Subset Control Program)
Files included in software kits to control the installation of the product. The SCP is written by the kit's developer and is invoked by the **setld** utility during the installation of the kit. System control program files have the extension `.scp`.

script
A program that is interpreted (as opposed to **compiled**) and executed by a specified shell. In general, script languages are easier and faster to code in than the more structured and compiled languages such as C and C++. Script languages generally take longer to process than compiled languages, but are useful for shorter programs.

SCSI (Small Computer System Interface)
An industry-standard bus for small systems such as personal computers, small multiuser systems, or workstations. SCSI-based devices can be configured in a series, with multiple devices on the same bus. SCSI is pronounced "scuzzy."

SCSI Interface Module
See *SIM (SCSI Interface Module)*

search path
A list of full pathnames (usually separated by colons) of directories to be searched for executable files and other kinds of files. Users can create search paths by defining variables, such as `path`, `$PATH`, and `MANPATH`.

security
The protection of data, system operations, and devices from accidental or intentional ruin, damage, or exposure.

sed
The command that invokes the `sed` utility, the standard stream editor. The `sed` editor reads one or more text files, makes editing changes according to a script of editing commands, and writes the results to standard output.

Serial Line Internet Protocol
See *SLIP (Serial Line Internet Protocol)*

server
A computer system that serves one or more other computers, called clients, by providing a resource to them.

server process
In the client/server model of communication, a process that provides services to client processes.

session
See *terminal session*

setld
A utility for installing, managing, updating, and removing software subsets.

See also *subset*

sh
The command that invokes either the **Bourne shell** or the **POSIX shell**, depending on the user set up in the `passwd` file.

shell
A component of the operating system that understands and executes commands entered by the user or by another program. The most popular

UNIX shells (all of which are included in the Tru64 UNIX operating system) are the **Bourne**, **C**, **Korn**, and **POSIX** shells.

See also *script*

sign-extend
To increase the data size of an operand smaller than the computer's data path by appending high-order bits to the operand. If the sign bit of the operand is a one, the added bits are ones; if a zero, they are zeros. This operation preserves the twos-complement numerical value of the operand.

silent character
See *hidden character*

SIM (SCSI Interface Module)
A subprogram designed to accept CAM Control Blocks routed through the XPT transport layer in order to execute SCSI commands.

Simple Mail Transfer Protocol
See *SMTP (Simple Mail Transfer Protocol)*

Simple Network Management Protocol
See *SNMP (Simple Network Management Protocol)*

SLIP (Serial Line Internet Protocol)
A transmission line protocol that encapsulates and transfers IP datagrams over asynchronous serial lines. SLIP is less efficient than PPP.

See also *PPP (Point-to-Point Protocol)*

SMTP (Simple Mail Transfer Protocol)
The Internet standard protocol for exchanging electronic mail.

SNMP (Simple Network Management Protocol)
The Internet standard protocol for exchanging network management information.

socket
In interprocess communications, an endpoint of communication. Also, the system call that creates a socket and the associated data structure.

socketpair
A pair of sockets that can be created in the UNIX domain for two-way communication. Like pipes, socketpairs require communicating processes to be related.

See also *pipe*

soft link
See *symbolic link*

sort
To organize the information in a file into the desired order based on specifiable criteria.

Source Code Control System
See *SCCS (Source Code Control System)*

source hierarchy
For building software kits, the directory tree and files that are to be compiled by the `kits` command into subsets for a kit.

See also *target hierarchy*

special file
See *device special file*

spooling
The process of copying files into a reserved disk area and then delivering the temporary copies to a serially accessed device as the device becomes ready to receive each new file. The temporary copies are delivered to the device in the order of their creation and are deleted as their delivery is completed; hence, spooling is a form of FIFO (first in, first out) buffering. The most common use of spooling is for printing. Rather than require a user to wait until the printer becomes available, the system spools the file to be printed. The user can then edit or delete the original copy.

standard error
The file to which programs write error messages. The standard error file (commonly called `stderr`) is a virtual file that by default is assigned to the user's screen but can be reassigned (redirected) to any device or file available to the user.

standard input
The file from which most programs receive input data or commands. The standard input file (commonly called `stdin`) is a virtual file that is by default assigned to the user's keyboard but can be reassigned (redirected) to any device or file available to the user.

standard output
The file to which programs write output data. The standard output file (commonly called `stdout`) is a virtual file that is by default assigned to the user's screen but can be reassigned (redirected) to any device or file available to the user.

statement
An instruction in a source language, shell script, command language, and the like.

status
The state in which a program exists.

stderr
See *standard error*

stdin
See *standard input*

stdout
See *standard output*

store-and-forward
A type of network connection in which a complete transmission is passed to one intermediate host before transmission to the next intermediate host begins.

stream
The TCP/IP definition developed for System V systems and now in wide use across UNIX systems.

stream editor
A program that manipulates the data in a text file by applying commands from a previously prepared list called a script instead of by accepting commands from the user. Powerful stream editors, such as **sed,** can perform any operation available to a full-function interactive line editor.

STREAMS
A kernel mechanism that supports the implementation of device drivers and networking protocol stacks.

See also *clist*, *STREAMS framework*

STREAMS framework
STREAMS components that define the interface standards for character I/O within the kernel and between the kernel and user levels. These components include functions, utility routines, kernel facilities, and data structures.

stream socket
A socket that provides two-way byte streams across a transport connection.

striping
A way of distributing disk I/O and to improve throughput. The striped data is divided into blocks (sometimes called chunks or stripes) and distributed across multiple disks in an array. Striping enables parallel I/O streams to operate concurrently on different devices, so that I/O operations can be handled simultaneously by multiple devices.

See also *RAID (redundant array of independent disks)*, *mirroring*

stty
A command that sets or reports certain characteristics of the user's terminal.

su
See *superuser*

subdirectory
A directory that is contained (nested) in another directory. The containing directory is called the parent directory.

subset
A software kit module that is installed or removed with the Tru64 UNIX `setld` utility. A subset usually consists of a collection of related files, such as an application and its support files.

subset control program
See *SCP (Subset Control Program)*

subset dependency
The condition in which a given subset requires the presence, or lack thereof, of other subsets in order to function properly. Evaluated by a subset's subset control program (SCP) under control of the `setld` utility.

superuser
A user possessing privileges to override the normal restrictions on file access, process control, and so forth. A user who possesses these privileges becomes a superuser by issuing the `su` command or by logging in to the system as root.

suspended
The condition of a process that is stopped but not killed. C shell, Korn shell, and POSIX shell users have the ability to suspend and reactivate processes by using the `fg` and `bg` commands, or by pressing Ctrl/z. A process that is suspended is called a "suspended job."

See also *terminate*

SVID (System V Interface Definition)
The specification that defines subroutine calls, system calls, commands, utilities, and services under System V.

See also *POSIX (Portable Operating System Interface)*

SVVS (System V Verification Suite)
A set of programs used to test adherence to the System V Interface Definition.

switch
Another name for an **option**.

symbolic link
A file that contains the pathname of another file or directory and acts as a pointer to that file or directory. The symbolic link can occur within the same file system or across file systems; also called a soft link.

See also *hard link*

sync
A command that forces all cached disk write operations to be completed before the system is halted.

synchronous execution
A mode of execution that forces transport primitives to wait for specific events before returning control to the transport user.

system call
A function that accesses the file system and communication facilities of the kernel.

system load
The demand that all processes place on the computer. System load is usually expressed as a number, with 1.0 representing 100 percent utilization and 0.1 representing 10 percent utilization of system resources.

System V
A version of the UNIX system developed by AT&T.

System V Interface Definition
See *SVID (System V Interface Definition)*

System V Verification Suite
See *SVVS (System V Verification Suite)*

T

tar file
A file created with the `tar` command that saves and restores multiple files in a single file. Tru64 UNIX patch kits are provided as tar files (except for kits included on the Tru64 UNIX CD–ROM).

target
In the `make` utility, an entity to be built from its dependents. An executable program is a target that is built from one or more object modules. Also called a target file.

target hierarchy
For building software kits, the directory tree into which a software kit is placed by the `kits` command.

See also *source hierarchy*

task

1. A defined activity; a unit of work to be performed, for example, a user task, a server task, and a processor task.

2. A process and the procedures that run the process.

TCP (Transmission Control Protocol)

The Internet transport-layer protocol that provides a reliable, full-duplex, connection-oriented service for applications. TCP uses the IP protocol to transmit information through the network.

TCP/IP

The two fundamental protocols of the Internet Protocol suite, and an acronym that is frequently used to refer to the Internet Protocol suite. TCP provides for the reliable transfer of data, while IP transmits the data through the network in the form of datagrams.

See also *TCP (Transmission Control Protocol)*, *IP (Internet Protocol)*

termcap database

A file containing descriptions of terminal types and capabilities; used by the `tset` command and BSD Curses library routines to determine how a given physical terminal is to be controlled.

See also *terminfo database*

terminal session

A user's interaction with a computer between the time the user logs in and logs out.

terminate

1. To place special electrical resistors (terminators) at the endpoints of a SCSI bus. With some SCSI devices you must manually insert or remove the terminators. Other devices have built-in terminators that are enabled or disabled with switches or software commands. A SCSI host adapter and the SCSI devices attached to it must be properly terminated or they will not work reliably.

2. To permanently end (kill) a process. A process that is terminated is called a "terminated job."

See also *suspended*

terminfo database

A file containing descriptions of terminal types and capabilities; used by the system and X/Open Curses library routines to determine how a given terminal is to be controlled.

See also *termcap database*

text file
A file that can be read using a command such as `more` and edited using a program such as `vi`. Text files are comprised of the ASCII character set.

See also *binary file*

tilde substitution
In the **POSIX**, **Korn**, and **C** shells, use of a tilde (~) as the first character in a pathname. By default, the shell interprets the tilde as the pathname of the user's home directory; for example, if a user whose login name is `rolf` enters `~/docs/figure_1` as a pathname, the system might expand the entry to be `/usr/users/rolf/docs/figure_1`. If the tilde is followed immediately by a user's login name, the shell interprets the combination as a reference to the named user's home directory; for example, `~willy` represents the path to Willy's home directory when entered by any user on the system.

tracepoint
A specific place in a source code program where the value of a variable is printed, without pausing the program's execution. Used to test and debug a program.

See also *breakpoint*

Transmission Control Protocol
See *TCP (Transmission Control Protocol)*

transport endpoint
A communication path over which a transport user can exchange data with a transport provider.

transport provider
A transport protocol that offers transport layer services in a network.

transport services
The support given by the transport layer in a network to the session layer for the transfer of data between user processes. The two types of services provided are connection oriented and connectionless.

transport user
A program needing the services of a transport protocol to send data to or receive data from another program or point in a network.

trap
1. In data communications, an unprogrammed, hardware-initiated, conditional jump to a specific address. Similar to an interrupt, but triggered by direct action of an executing program rather than by an external event.

2. In programming languages, the process of branching or jumping to a subroutine that provides the desirable operation when a specific condition occurs.
3. In the UNIX system, a special statement used to catch signals in a shell script and transfer control to a handler routine within the script.

trap handler

A system-defined routine used when an abnormal situation arises during a program's execution.

tree structure

1. The organization of disk directories in most operating systems. Any given directory can contain files or other directories (called subdirectories), or both. By extension, any subdirectory can contain subdirectories of its own; when diagrammed, the resulting structure resembles the branching of a tree.
2. The organization of data in a manner similar to that described for disk directories. Common tree structures in files are the binary tree, in which each data element has zero, one, or two elements beneath it (called children); and the B+ tree, in which each data element can have more than two children, with the distribution of elements in the tree being balanced so that all of the elements at a given level have the same or similar numbers of children.

Trojan Horse

A computer program that appears to do something useful but is also designed to damage or destroy other files or programs or the system itself, without the user's knowledge. An example of a Trojan Horse would be a game program that secretly erased disk files while the game was being played.

See also *virus*, *worm*

trusted host

A computer within a network that permits access without the need to supply password information.

tty

A shorthand term for a terminal.

U

UBC (Unified Buffer Cache)

Functions as a layer between the operating system and a storage disk by temporarily holding recently accessed file system data for reads and writes from conventional file activity and holding page faults from mapped file sections.

Processes and the UBC compete for a limited amount of physical memory, and the virtual memory subsystem allocates physical pages according to the process and file system demand. To be able to meet the demands of competing claims on memory resources, the virtual memory subsystem periodically reclaims the oldest pages by writing their contents to swap space. Under heavy loads, entire processes may be suspended (swapped out) to free memory.

See also *VFS (Virtual File System)*

UDP (User Datagram Protocol)
The Internet protocol that allows application programs on remote machines to send datagrams to one another. UDP uses IP to deliver the datagrams.

uid, UID
See *user ID (UID)*

ULTRIX
A forerunner to the Tru64 UNIX operating system. Originally produced by Digital Equipment Corporation, the ULTRIX operating system runs on VAX and RISC computers, whereas Compaq Tru64 UNIX runs on Alpha systems.

umask
A three-digit octal number that specifies the default permissions given to a file when it is created. The `umask` command sets or changes this number.

unified buffer cache
See *UBC (Unified Buffer Cache)*

UNIX
An operating system that was originally developed at the Bell Laboratories of AT&T in the late 1960s and early 1970s and subsequently enhanced by the University of California at Berkeley, AT&T, the Open Software Foundation (OSF), and others.

UNIX-to-UNIX Copy Program
See *UUCP (UNIX-to-UNIX Copy Program)*

unlink
The system call used to sever the connection between files that had been created with the `link` system call.

unmount
To announce to the system that a file system previously mounted on a specified directory is to be removed. Only the person who mounted the particular file system or a superuser can unmount it. A file system is unmounted with the `umount` command.

update installation
A type of installation that preserves disk partitions; file systems; file customizations; the network, print, and mail environments; user accounts; user created files; and any other system setup you may have done.

See also *full installation*

up time
The period during which a machine is available for use.

See also *down time*

upward compatible
Pertaining to that which is designed for use on small machines but capable of running without change on larger machines.

URL (Uniform Resource Locator)
The address of a file or other resource accessible on the Internet. The type of file or resource depends on the Internet application protocol. For example, using the HyperText Transfer Protocol (HTTP), the file can be an HTML page, an image file, or a program such as a CGI application or **Java** applet. Such an address would look like this: **http://www.tru64unix.compaq.com/**, which is the URL for the Compaq Tru64 UNIX Web site.

User Datagram Protocol
See *UDP (User Datagram Protocol)*

user ID (UID)
The number associated with each login name. This number is stored in the `/etc/passwd` file.

See also *group ID (GID)*, *process ID (PID)*

user name
See *login name*

/usr
A read-only file system in which some components of the operating system and of applications are stored. Users' home directories are sometimes also located in a subdirectory of `/usr`.

UUCP (UNIX-to-UNIX Copy Program)
A set of programs and protocols for connecting computers by means of dial-up lines. The programs include facilities for copying files, logging in to remote computers, and encoding binary files for transmission of 7-bit ASCII data lines. The ease of connection and low cost have made UUCP one of the most popular information networks in the world.

UUCP network
A term applied to any grouping of computers connected by means of the UUCP programs.

V

variable
In programming languages, shell scripts, command procedures, and the like, a symbol whose value is allowed to change.

variable expansion
The replacement of the variable identifier with its associated strings in a shell command line.

variable modifier
A symbol referring to part of a variable, usually under the assumption that its value is a pathname.

version control file
In a version control system, a file that consists of original text and a set of revisions (deltas) that have been made to it. In RCS, this file is called an "RCS file"; in SCCS, an "s-file."

version control library
A directory that contains files that are organized and maintained under a version control system, such as RCS or SCCS.

version control system
A software tool that aids in the organization and maintenance of file revisions and configurations. In particular, it automates the storing, logging, retrieval, and identification of revisions to source programs, documentation, and data files.

See also *version control library*

VFS (Virtual File System)
A uniform interface that allows common access to files, regardless of the file system on which the files reside.

vi editor
A full-screen text editor. The `vi` editor is a modal editor. In command mode, it accepts commands for cursor movement, text deletion, and so forth. To insert text into the file, the user gives the editor a command that places the editor in input mode, and all keystrokes thereafter are interpreted as input data until the Escape key is pressed.

See also *full-screen editor*

Virtual File System
See *VFS (Virtual File System)*

virtual memory
Because processes and file systems compete for a limited amount of physical memory, the virtual memory subsystem periodically reclaims the oldest pages by writing their contents to swap space or disk (paging). Under heavy loads, entire processes may be suspended to free memory (swapping).

virus
A piece of software designed to attach itself to other computer programs or files in a system and then to replicate itself indefinitely through any available means (disk file, network, and so forth) into other computers. Viruses are usually designed to damage or destroy "infected" programs or systems and are often programmed to become destructive at a specific time, such as the birthday of the virus's programmer.

See also *Trojan Horse*, *worm*

visual editor
See *full-screen editor*

W

WEBES (Web-Based Enterprise Services)
A common architecture implemented by Compaq for many of its service tools. For example Compaq Analyze and Insight Manager utilize the common components of WEBES and add their own functions. Other service tools can be added to the same machine utilizing the same common WEBES components. Each WEBES-based service tool adds functionality to the Director, a process that executes continuously on the machine.

wildcard character
A metacharacter that is used to allow wildcard matching in file names or regular expressions.

See also *metacharacter*, *regular expression*

wired memory
Physical memory that is allocated to the operating system, usually at boot time, and cannot be used for paging. Wired memory is either static or dynamic:

- Static wired memory is allocated at boot time and used for operating system data and text and for system tables. It is also used by the metadata buffer cache, which holds recently accessed UNIX File System (UFS) and CD-ROM File System (CDFS) metadata.

- Dynamically wired memory is allocated at boot time and used for dynamically allocated data structures, such as system hash tables.

word identifier

A piece of a command line delimited by blanks and recognized as a unique entity by the shell. Used to save keystrokes. By using word identifiers, a user can select part of a previous command line for use in the current command line.

wordlist

A C shell variable consisting of more than one word.

working directory

1. The directory from which a file is read or into which a file is written when a program does not include a directory path in the name of the file when operating on it.
2. The user's current directory.

See also *home directory*

worm

A computer program designed to replicate itself and spread through a network into other computers. Worms are not attached to other programs or files. Worms are usually designed to damage or destroy "infected" systems and are often programmed to become destructive at a specific time, such as the birthday of the worm's programmer. Some worms are not designed to cause damage, but they are still harmful because they occupy resources intended for legitimate use.

See also *Trojan Horse*, *virus*

WORM

Refers to storage media that can be written once and read many times, such as a recordable compact disc (CD-R).

X

X/Open Transport Interface

See *XTI (X/Open Transport Interface)*

XPT

A transport layer of software that SCSI peripheral drivers use to originate the execution of CAM (Common Access Method) functions.

XTI (X/Open Transport Interface)

Protocol-independent, transport-layer interface for applications. XTI consists of a series of C language functions based on the Transport Layer

Interface (TLI), which in turn was based on the transport service definition for the OSI model.

X Window System

A network-based windowing interface. The X Window System has been adopted by many major computer manufacturers.

Y

yacc (Yet Another Compiler-Compiler)

A program for generating parsers (programs that can interpret their input in a rational manner). The output from `yacc` is a C language program. The `yacc` program is usually used to generate parsers for interpreting the output of a `lex`-generated front end.

See also *lex*, *parser*

Yellow Pages

See *NIS (Network Information Service)*

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y