SINIX Open Desktop
Open Desktop
Administrator's Guide, Part 2

Edition April 1990 (SINIX Open Desktop V1.0)

Order No. U5746-J-Z95-1-7600
Printed in the Federal Republic of Germany
6330 AG 3900.45 (7920)

Delivery subject to availability;
right of technical modifications reserved.

Published by Bereich
Daten- und Informationstechnik
Postfach 83 09 51, D-8000 München 83

Siemens Aktiengesellschaft

Administrator's Guide Contents

Administering ODT-VIEW

Part 1

Chapter 1: Introduction 1

Chapter 2: X Window System Overview 3
- X Window System Organization 3
- The Window Manager 4
- Input Focus 7
- Selecting Startup Clients 8

Chapter 3: The .Xdefaults File 9
- .Xdefaults Overview 9
- mwm Resource Descriptions and Syntax 11

Chapter 4: The .mwmrc File 33
- .mwmrc Overview 33
- Sample .mwmrc File 34
- Window Manager Functions 36
- Using Functions 44

Chapter 5: Desktop Manager Overview 51
- Changing the Appearance of the Desktop Manager 51
- Changing the Behavior of the Desktop Manager 53
- Typical Applications 55

Chapter 6: Desktop Manager Tutorials 59
- Determining the Appearance of Your Desktop 59
- Building Intelligence into Your File Icons 62
- Loading Files into a Program by Dragging 64
- Building Intelligence into Directories 65

Chapter 7: Desktop Manager Reference 69
Rule Files 69
Mapping Triggers 89
Desktop Command Language 93
Picture Files 97
Defaults Files 101
Message Files and Language Support 111
Command-Line Options 118
X.Deskware Support Utilities 119

Appendix A: Setting Streams Parameters 125
Overview 125
Displaying Parameters 125
Changing Parameters 127
Rebuilding and Rebooting 130

Appendix B: Monochrome Configuration File 131

Appendix C: Customizing Screen Colors 133
Defining Colors in the RGB Database 134

Appendix D: Changing Video Systems 137
Overview 137
Description of the Configuration Scripts 137
Running the Configuration Scripts 138
Examples 139

Administering ODT-OS

Chapter 1: Introduction 1
The System Administrator and Administrative Roles 1
Making Administration Easier with the sysadmsh 2
The Super User Account 3
The Keyboard 4
About This Guide 5

Chapter 2: Using the System Administration Shell 7
Starting sysadmsh 7
How the Screen is Organized 8
Selecting Menu Items 9
Using Forms 11
Using Scan Windows 17
Getting Help 19
The Function Keys 22

Chapter 3: Starting and Stopping the System 23
Starting the System 23
Logging in as the Super User 28
Stopping the System 29
Understanding the Boot Display Information 31

Chapter 4: Using Filesystems 33
What Is a Filesystem? 33
Maintaining Free Space in Filesystems 34
Filesystem Integrity 38

Chapter 5: Maintaining System Security 45
What Is a Trusted System? 46
Running a Trusted System 50
Using the Audit Subsystem 56
Filesystem Protection Features 90
Verifying System Integrity 96
Security-Related Error Messages 101
Adding Dial-in Password Protection 106

Chapter 6: Backing Up Filesystems 107
Strategies for Backups Using sysadmsh 107
Preparations for Scheduled Backups 108
Performing a Scheduled Backup 114
Performing an Unscheduled Backup 117
Verifying a Backup 119
Getting a Backup Listing 120
Restoring Individual Files or Directories from Backups 121
Restoring an Entire Filesystem 124
An Explanation of Backup Levels 125

Chapter 7: Adding Device Drivers with the Link Kit 129
Device Drivers 129

Chapter 8: Using DOS and OS/2 137
OS/2 Coexistence 138
Partitioning the Hard Disk Using fdisk 138
Installing a UNIX Partition on a DOS System 142
Using a UNIX System and DOS with Two Hard Disks 143
Removing an Operating System from the Hard Disk 144
DOS Accessing Utilities 144
Mounting DOS Filesystems on a UNIX System 146

Chapter 9: Administering User Accounts 151
Account Management 152
Default Account Configuration 164
Activity Report Generation 176

Chapter 10: UNIX Directories and Special Device Files 181
UNIX Directories 181
Log Files 187
Special Device Files 189

Chapter 11: Adding Ports and Modems 193
Adding and Configuring Serial Ports 193
Using a Modem on Your System 195

Chapter 12: Using Printers 209
Installing a Printer 211
Summary of User Commands 215
Summary of Administrative Commands 216
Starting and Stopping the LP Print Service 217
Canceling a Print Request 219
Enabling and Disabling Printers 219
Adding a Printer to a Class 220
Setting the System Default Destination 221
Mounting a Form or Print Wheel 222

Removing a Printer or Class 223
Managing the Printing Load 224
Managing Queue Priorities 226
Troubleshooting the Print System 232
Customizing the Print Service 236
Specialized Configuration Options 250
Setting Up RTS/CTS Protocol Serial Printers 261
Using a Printer without the Spooler 264

Chapter 13: Using Floppy Disks and Tape Drives 265
Using Cartridge Tape Drives 265
Using Floppy Disks 274

Chapter 14: Using Bus Cards 279
Installing Bus Cards 279
Adding Additional Memory 281

Chapter 15: Using a Mouse 283
Installing the Hardware 283
Installing a Mouse 284
Using the Mouse 288

Chapter 16: Setting Up Electronic Mail 289
How MMDF Works 289
Configuring MMDF 297
Changing Your Machine or Site Name 309
Routing Example 309
Updating the Database 310
Maintaining Your MMDF System 310
Converting Existing Configuration Files 311

Chapter 17: Adding Hard Disks 315
Before You Start 317
Installing the Hard Disk 321
Adding the New Filesystem 333
Relinking the Kernel 335

Administering ODT-NET

Part 2

Chapter 1: Overview 1
Networking Concepts 2
Common Network Administration Tasks 11

Chapter 2: TCP/IP Network Administration 13
Kernel Configuration 13
Runtime Configuration of STREAMS Drivers 16
Setting Interface Parameters 18
Local Subnetworks 18
Internet Broadcast Addresses 19
Routing 20
Using UNIX System Machines as Gateways 21
Network Servers 21
Network Databases 22
Network Tuning and Troubleshooting 25

Chapter 3: Name Server Operations Guide for BIND 33
The Name Service 33
Types of Servers 34
Setting Up Your Own Domain 36
Remote Servers 39
Initializing the Cache 40
Standard Resource Records 40
Some Sample Files 48
Additional Sample Files 52
Domain Management 54

Chapter 4: Synchronizing Network Clocks 57
How a Time Daemon Works 57
Guidelines 58
Options 59
Daily Operation 60

Chapter 5: Configuring NFS 61
Role of the Operating System in NFS 61
Introducing NFS 62
Setting Up an NFS Client 63
Starting and Stopping NFS 65
Debugging NFS 65
Adding a New User 74
Incompatibilities with Remote Filesystems 74
Handling Clock Skew in User Programs 76

Chapter 6: Managing the LAN Manager Client Network 79
Special Network Files 81
Starting and Stopping the Network 81
NetBIOS 82
Network Parameter Descriptions 83
Configuring for Performance 97

Chapter 7: Building a Remote Network with UUCP 103
What Is UUCP? 103
How to Use This Chapter 104
What You Need 104
UUCP Commands 105
Connecting Remote UUCP Systems with a Modem 111
Configuring UUCP on Your System 118
Administering Your UUCP System 137
Troubleshooting 140
Keeping Traffic and Congestion under Control 142
Complete UUCP Examples 143
UUCP Error Messages 149

Glossary 155

Administering ODT-DOS

Chapter 1 Introduction 1
Who Should Use This Guide 1
Organization of This Guide 1
ODT-DOS Guides 2
Installing ODT-DOS 2
Release Notes 2

Chapter 2 Administering ODT-DOS 3
Using the dosadmin Program 4
Adding And Deleting User Accounts 4
Administering DOS Applications 4
Administering the System Console 6
Administering COM Ports 11
Administering DOS Printers 11
Backing Up the ODT-DOS Filesystem 15
Administering Disk and Diskette Drives 15
Administering the Physical DOS Partition 16
Administering Virtual DOS Partitions and Virtual Floppy Disks 19
Installing Plug-In Cards in Your Computer 25
Making New DOS Images 31
System Files Affected by System Administration 35

Chapter 3 Installing DOS Applications 37
Installing DOS Applications Using dosadmin 37
Installing Copy-Protected DOS Applications 50
Removing DOS Applications 54

Administering ODT-DATA

Chapter 1: Introduction 1
Introduction to Release 6 2
Organization of This Document 2
Associated Publications 4

Chapter 2: Overview of Installation Tools 5
ODT-DATA Installation Utilitybuild 5
ODT-DATA Installation and DBMS Server Start Up 5
The Installation Shut Down Utilityutserver 6

Chapter 3: Configuration Decisions 7
Configuration Requirements 7
General Suggestions for Avoiding Problems 11

Chapter 4: Installing ODT-DATA 13
Manual Initialization 13

Chapter 5: Maintenance Utilities 19
The Server (iidbms) Maintenance Utilitymonitor 19
Shared Memory and Semaphores Report Utilityreport 22
The Locking Facility Reportvckstat 24
The Logging Facility Reportvgstat 26

Chapter 6: Installation Reference Material 31
ODT-DATA Installation and Server Start Up Utility 31
ODT-DATA Installation Shutdown 41

Chapter 7: Troubleshooting with Log Files 45
ODT-DATA Log Files 45

Appendix A: ODT-DATA Startup Files 47
Installation-Wide Startup Files 47
Database-Specific Startup File 47
User-Specific Startup File 48

Appendix B: Authorizing User Access to ODT-DATA and Databases 49
Database Access 49
Defining the Terminal 50
Invoking accessdb 51
Using accessdb 51
Functions in accessdb 52
Summary of Accessdb 59

Appendix C: ODT-DATA Environment Variables 61
Setting Installation Wide Environment Variables 61
Setting User Defined Environment Variables 62
Environment Variable List 62

Appendix D: ODT-DATA System Recovery 71
Using finddbs 71

Appendix E: Running ODT-DATA under the Network File System 75
Configuration Scenarios 75

Glossary 81

Index

Administering ODT-NET

Contents

Chapter 1: Overview 1
Networking Concepts 2
Common Network Administration Tasks 11

Chapter 2: TCP/IP Network Administration 13
Kernel Configuration 13
Runtime Configuration of STREAMS Drivers 16
Setting Interface Parameters 18
Local Subnetworks 18
Internet Broadcast Addresses 19
Routing 20
Using UNIX System Machines as Gateways 21
Network Servers 21
Network Databases 22
Network Tuning and Troubleshooting 25

Chapter 3: Name Server Operations Guide for BIND 33
The Name Service 33
Types of Servers 34
Setting Up Your Own Domain 36
Remote Servers 39
Initializing the Cache 40
Standard Resource Records 40
Some Sample Files 48
Additional Sample Files 52
Domain Management 54

Chapter 4: Synchronizing Network Clocks 57
How a Time Daemon Works 57
Guidelines 58
Options 59
Daily Operation 60

Chapter 5: Configuring NFS 61
Role of the Operating System in NFS 61
Introducing NFS 62
Setting Up an NFS Client 63
Starting and Stopping NFS 65
Debugging NFS 65
Adding a New User 74
Incompatibilities with Remote Filesystems 74
Handling Clock Skew in User Programs 76

Chapter 6: Managing the LAN Manager Client Network 79
Special Network Files 81
Starting and Stopping the Network 81
NetBIOS 82
Network Parameter Descriptions 83
Configuring for Performance 97

Chapter 7: Building a Remote Network with UUCP 103
What Is UUCP? 103
How to Use This Chapter 104
What You Need 104
UUCP Commands 105
Connecting Remote UUCP Systems with a Modem 111
Configuring UUCP on Your System 118
Administering Your UUCP System 137
Troubleshooting 140
Keeping Traffic and Congestion under Control 142
Complete UUCP Examples 143
UUCP Error Messages 149

Glossary 155

Chapter 1
Overview

After you have installed your system following the procedures described in the *Installation Guide*, this guide provides background and reference material for you as the network system administrator. You might need to refer to this information to understand how the network software operates so you can customize your network configuration.

The first part of this chapter presents networking concepts and introduces the four networking products that make up ODT-NET: TCP/IP, SCO™ NFS™, SCO LAN Manager, and UUCP. The latter part of this chapter provides a list of common network administration tasks and directs you to the sections within this guide that help you perform the tasks.

Chapters 2, 3, and 4 relate to TCP/IP, which is the underlying software layer of the network. All network system administrators should become familiar with the general information about TCP/IP presented in Chapter 2. You need to read Chapters 3 and 4 only if you plan to use the optional network services (name server and clock synchronization).

Chapters 5, 6, and 7 contain detailed administrative information about SCO NFS, SCO LAN Manager, and UUCP.

Within the following chapters, you are often referred to the manual pages for further information. You will find all the networking manual pages referred to in this guide together in the "Networking Commands" section of **xman**(X).

Networking Concepts

A distributed network of machines, such as the sample network in Figure 1-1, can provide more aggregate computing power than a mainframe computer, with far less variation in response time over the course of the day. Thus, a network is generally more cost-effective than a central mainframe.

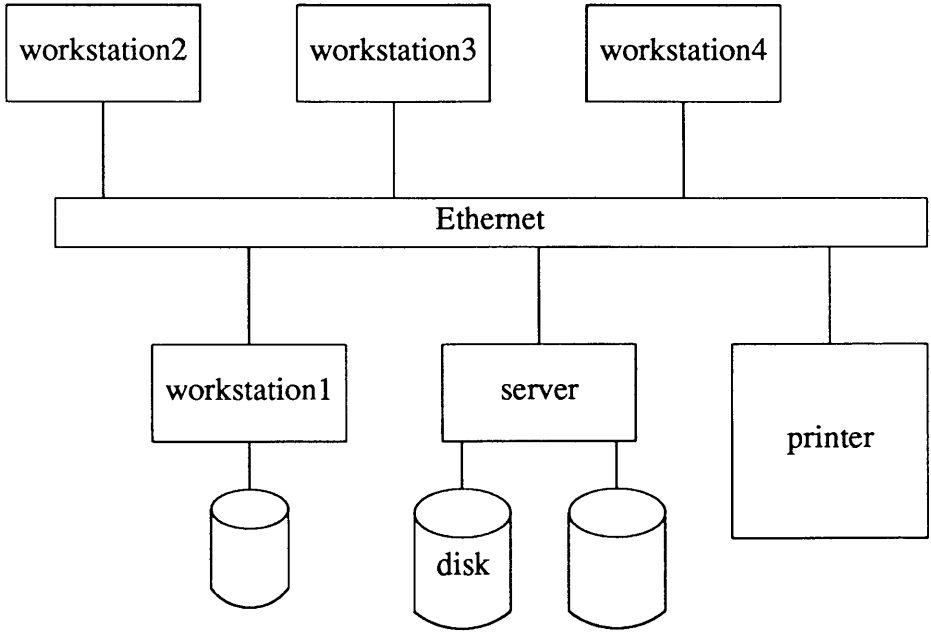


Figure 1-1. Sample Network

The network system administrator oversees the maintenance and operation of the network and is responsible for:

- installing and maintaining network hardware
- installing the network software on each network computer
- assigning names to each computer and device on the network
- assigning names to network users and groups
- adding new users, groups, hardware and software to the network
- performing the commands required to share, remove, and restrict resources
- checking that the required special files are correct

Network Names

Each network computer, user, and group must be identified by a unique name. The network programs use this name to locate computers and to validate network requests.

Network names must be fifteen characters long or less. The first fifteen characters in a network name must be unique to the computer, user, or group using the name. Names cannot be duplicated on the network. For example, you could have computers named "onecomputername" and "twocomputername," but not "computernamesix" and "computernamesixteen." In the last pair of computer names, the first fifteen letters are identical. Each computer name must also be unique regardless of case. For example, the network cannot support a machine named "Bob" and a machine named "bob." For this reason, computer names should use only lowercase letters.

Consistency of Network ID Files

It is imperative that the administration of the computers on the network be consistent. Many functions of the network depend upon a consistent user identification number (user ID) for each individual user on all network machines. If you are connecting existing computers in a network for the first time, be certain that user IDs are consistent from computer to computer. User IDs are defined by the **sysadmsh**(ADM) **Accounts→User→Create** selection and placed in the UNIX® system database files (including */etc/passwd*).

For information on adding user accounts, see the "Administering User Accounts" chapter in *Administering ODT-OS* in the *Administrator's Guide*. For information on maintaining ID consistency, see the **passwd**(C), and *passwd*(F) manual pages.

If you are adding a new computer to the network, or building a network using computers that have never run the UNIX system, make sure the user and group IDs in the new machine's */etc/passwd* and */etc/group* files do not conflict with any other machines on the network.

NOTE: Never edit */etc/passwd* with a text editor. This could damage the password database information and cause your system to refuse further logins. Always use the **sysadmsh**(ADM) **Accounts→User→Create** selection to administer accounts.

User ID numbers must not be duplicated among different users. An individual user can have many different account names, but all account names belonging to an individual user must have the same user ID number.

For example, the user Joe Sparks has accounts on different network computers under the names "joesp", "sparks", and "jsparks". All three of these accounts must have the same user ID number. If the account "joesp" is associated with the user ID number 301, then the accounts under the name "sparks" and "jsparks" must also have user ID number 301.

NOTE: If user ID numbers are not compatible on all network machines, the results of some network commands can be misleading. For example, if a user has different ID numbers on "machine1" and "machine2," and this user tries to access a directory on "machine2" as a user on "machine1," then the local user ID number on "machine1" determines the file ownership information. This means that files on "machine2" may not be accessible because the individual is not recognized across the network.

The entries for accounts belonging to Joe Sparks look like this:

in //machine1/etc/passwd:

```
joesp::301:110:Joe Sparks:/usr/joesp:/bin/csh
```

in //machine2/etc/passwd:

```
sparks::301:110:Joe Sparks:/usr/sparks:/bin/csh
```

in //machine3/etc/passwd:

```
jsparks::301:110:Joe Sparks:/u/jsparks:/bin/sh
```

TCP/IP Background

TCP/IP is a set of protocols used to interconnect computer networks and to route traffic among many different computers. "TCP" means Transmission Control Protocol, and "IP" means Internet Protocol. Protocols are standards that describe allowable formats, error handling, message passing, and communication standards. Computer systems that conform to communications protocols such as TCP/IP are thus able to speak a common language. This enables them to transmit messages accurately to the correct destination, despite differences in the hardware and software of the various machines.

Many large networks conform to these protocols, including the DARPA Internet (Defense Advanced Research Projects Agency Internet). A variety of universities, government agencies, and computer firms are connected to an internetwork that follows the TCP/IP protocols. Thousands of machines are connected to this internet. Any machine on the internet can communicate with any other. (The term internetworking refers to the action of joining two or more networks together. The result can be described as a network of networks, which is called an "internet.") Machines on the internet are referred to as "hosts" or "nodes."

TCP/IP provides the basis for many useful services, including electronic mail, file transfer, and remote login. Electronic mail is designed to transfer short text files. The file transfer application programs transfer very large files containing programs or data. They also provide security checks controlling file transfer. Remote login allows users on one computer to log in at a remote machine and carry on an interactive session.

The Internet Protocol (IP)

The Internet Protocol, IP, defines a connectionless packet delivery. This packet delivery connects one or more packet-handling networks into an internet. The term "connectionless" means that the sending and receiving machines are not connected by a direct circuit. Instead, individual packets of data (datagrams) are routed through different machines on the internet to the destination network and receiving machine. Thus, a message is broken up into several datagrams that are sent separately. Note that connectionless packet delivery by itself is not reliable. Individual datagrams may or may not arrive, and they probably will not arrive in the order in which they were sent. TCP adds reliability.

A datagram consists of header information and a data area. The header information is used to route and process the datagram. Datagrams may be fragmented into smaller pieces, depending on the physical requirements of the networks they cross. (When a gateway sends a datagram to a network that cannot accommodate the datagram as a single packet, the datagram must be fragmented into pieces that are small enough for transmission.) The datagram fragment headers contain the information necessary to reassemble the fragments into the complete datagram. Fragments do not necessarily arrive in order; the software module implementing the IP protocol on the destination machine must reassemble the fragments into the original datagram. If any fragments are lost, the entire datagram is discarded.

The Transmission Control Protocol (TCP)

The Transmission Control Protocol,TCP, works with IP to provide reliable delivery. It provides a means to ensure that the various datagrams making up a message are reassembled in the correct order at their final destination and that any missing datagrams are sent again until they are correctly received.

The primary purpose of TCP is to provide a reliable, secure, virtual-circuit connection service between pairs of communicating processes on top of unreliable subnetworking of packets, where loss, damage, duplication, delay, or misordering of packets can occur. Also, security provisions such as limiting user access to certain machines can be implemented through TCP.

TCP is concerned only with total end-to-end reliability. It makes few assumptions about the possibility of obtaining reliable datagram service. If a datagram is sent across an internet to a remote host, the intervening networks do not guarantee delivery. Likewise, the sender of the datagram has no way of knowing the routing path used to send the datagram. Source-to-destination reliability is provided by TCP in the face of unreliable media; this makes TCP well-suited to a wide variety of multi-machine communication applications.

Reliability is achieved through checksums (error detection codes), sequence numbers in the TCP header, positive acknowledgment of data received, and retransmission of unacknowledged data.

Protocol Layering

Communications software protocols are divided into different layers, where the lowest layer is the hardware that physically transports the data, and the highest layer is the applications program on the host machine. Each layer is very complex in its own right, and no single protocol could encompass all the tasks of the various layers. As discussed earlier, the Internet Protocol handles the routing of datagrams, while the Transmission Control Protocol, which is the layer above IP, provides reliable transmission of messages that were divided into datagrams. The applications programs in turn rely on TCP to send information to the destination host.

TCP/IP has four software layers built on an underlying hardware layer. Its model is shown in Figure 1-2.

Layer	Functionality
4	Application
3	Transport
2	Internet
1	Network Interface

Figure 1-2. TCP/IP Model

The layers operate as follows:

Network Interface Layer	Accepts and transmits data over the network
Internet Layer	Takes care of communication among machines
Transport Layer	Provides communication between application programs
Application Layer	Accesses the Internet, and sends and receives data

To the applications programs, TCP/IP appears to provide a full-duplex virtual circuit between the machines. In actuality, all information is divided into datagrams, which may then be further fragmented during transmission. The software modules implementing IP then reassemble the individual datagrams, while the modules implementing TCP make sure that the various datagrams are reassembled in the order in which they were originally sent.

There are several higher-level specialized protocols for specific applications such as terminal traffic (**telnet**(TC)) and file transfer (**ftp**(TC)), and protocols for other network functions such as gateway-status monitoring. In this manual, however, these are not usually referred to as protocols, but rather as programs or services.

Message Routing

The following sections explain gateways and network addresses. These two concepts are the key to understanding how datagrams are routed through an internet.

Gateways

The various networks that compose an internet are connected through gateway machines. A gateway is a machine that is connected to two or more networks. It can route datagrams from one network to another. Gateways route the datagrams based on the destination network, rather than the individual machine (host) on that network. This simplifies the routing algorithms. The gateway decides which network should be the next destination of a given datagram. If the destination host for the datagram is on that network, the datagram can be sent directly to that host. Otherwise, it continues to pass from gateway to gateway until it reaches the destination network.

Network Addresses

Each host machine on a TCP/IP internet has a 32-bit network address. The address includes two separate parts: the network ID and the host machine ID. Machines that serve as gateways thus have more than one address, because they are on more than one network. Internet addresses are assigned by the Network Information Center (NIC) located at SRI International in Menlo Park, California. The NIC assigns only network IDs; the individual network administrators then assign the host machine IDs for their network.

There are three classes of network addresses, corresponding to small, medium, and large networks. The larger the network, the larger the number of hosts on that network; likewise, smaller networks have fewer hosts. Thus, when the 32-bit network address is divided between the network ID and the host machine ID, larger networks need a larger number of bits to specify all the hosts on the network uniquely. Also, there are only a small number of really large networks, and so fewer bits are needed to identify these networks uniquely. The

network addresses are thus divided into three classes, identified as A, B, or C. The following table lists these classes and their formats.

Class	Network Size Configuration
Class A	Allocates a 7-bit network ID and a 24-bit host ID
Class B	Allocates a 14-bit network ID and a 16-bit host ID
Class C	Allocates a 21-bit network ID and an 8-bit host ID

All network addresses are 32 bits. The first bit of a Class A address is **0** (zero), identifying the address as Class A. Class B addresses begin with the digits **10**, and Class C addresses begin with **11**.

This system of network address classes provides a unique address for the entire statistical distribution of types of networks that might be expected among the various networks using this address system. There are a smaller number of large networks, having many hosts (Class A), a larger number of small networks, consisting of a lesser number of hosts (Class C), and a medium number of networks made up of a medium number of hosts (Class B).

Network addresses are often written as four decimal integers separated by periods (.), where each decimal number represents one octet of the 32-bit network address. For example, a machine might have the address 128.12.3.5.

How NFS Fits into the Network

Even in a network environment, sharing programs and data can sometimes be difficult. Either files have to be copied to each machine where they are needed, or users have to log into the remote machine with the required files. Network logins are time-consuming, and multiple copies of a file become confusing as incompatible changes are made to different copies.

To solve this problem, a distributed filesystem was designed to permit client systems to access shared files on a remote system. Client machines, also called clients, request resources provided by other machines, which are called servers. A server machine makes particular filesystems available, and client machines can mount these as local filesystems. Users can then access remote files as if they were on the local machine.

SCO Network File System (NFS) is a software product that allows you to mount directories across the network and then treat remote files as if they were local. NFS was developed as a standard for the exchange of data between different machines and operating systems.

NFS fits into the application layer of the TCP/IP model. NFS was designed to fit into the network services architecture and not to extend the operating system onto the network. Thus NFS is an interface to allow a variety of machines and operating systems to play the roles of client and server, but it is not a distributed operating system.

The goal with NFS is to make all disks available as needed. Individual workstations have access to all information residing anywhere on the network. Printers and supercomputers may also be available on the network.

LAN Manager Connects to Microsoft LAN Manager

LAN Manager provides a distributed filesystem with a Microsoft® LAN Manager network. In this local-area network, OS/2®, DOS, and UNIX system machines use TCP/IP to share files.

LAN Manager is compatible with both IBM® PC-Network, Microsoft MS-NET, and Microsoft LAN Manager software, allowing several systems to share files across a local-area network. LAN Manager provides an easy-to-use, flexible, and powerful means of merging multiple OS/2, DOS, and UNIX system machines into a single network.

A LAN Manager network is made up of one or more server computers that control network resources, and one or more client (or consumer) computers that use the resources of the servers. Each computer is connected to the network by a cable. Requests are sent by the consumers and acted upon by a server, then returned to the consumer.

There are certain protocols necessary for the computers to verify requests and perform actions. Controlling these protocols is the function of LAN Manager. LAN Manager uses the "session" or "transport layer" interface provided by the network transport hardware. This interface makes LAN Manager independent of individual network hardware configurations.

When a user on a consumer machine uses a LAN Manager command to access a server computer, the request is passed through various software and hardware layers. These include the LAN Manager distributed filesystem protocol, NetBIOS software and hardware, a transport subsystem, and the cable connecting the machines.

Your Open Desktop™ system can act only as a consumer (or client) to access files from a DOS or OS/2 server; it cannot act as a server to the DOS or OS/2 machine. Through LAN Manager, your Open Desktop system can also consume from a XENIX-NET server. For information about providing access to a XENIX-NET server, see the documentation provided with the XENIX-NET server.

Wide-Area Networks through UUCP

To communicate with computers outside of the local TCP/IP network, the **uucp**, **cu**, and **ct** programs provide access to wide-area networks through telephone lines.

UUCP is a series of programs that provide file-transfer and remote execution capabilities. **cu** and **ct** provide interactive terminal sessions with a computer at a remote site.

cu connects your computer to a remote computer so you can be logged in on both at the same time. You can transfer files or execute commands on either computer without dropping the initial link.

ct connects your computer to a remote terminal so the user of the remote terminal can log in. The user of a remote terminal can call the computer and request that the computer call it back. The computer then drops the initial link so that the remote terminal's modem is available when it is called back.

Common Network Administration Tasks

Certain common tasks are performed by many network system administrators. The purpose of this section is to quickly describe those tasks, then direct you to where you can find the detailed information you need to complete the task. The tasks covered are:

- adding to the hosts database
- setting up routing tables
- establishing user equivalence
- setting up anonymous **ftp**
- mounting a remote filesystem at boot time
- using the name server

Adding to the Hosts Database

The */etc/hosts* file is a list of hosts on the network. Network library routines and server programs use this file to translate between host names and DARPA Internet addresses when the name server is not being used. (Chapter 5 describes how to use the name server.)

To add a machine to your network, you can add an entry to the */etc/hosts* file. Refer to the *hosts*(SFF) manual page for a description of the file format.

Setting Up Routing Tables

Routing tables provide the information needed to properly route packets to their destinations. See "Routing" in Chapter 2 for descriptions of two possible approaches for maintaining routing information. "Network Tuning and Troubleshooting" contains a section on obtaining information about the system routing tables.

Establishing User Equivalence

You can control who has access to a machine through the network by establishing user equivalence within the */etc/hosts.equiv* file. The **rlogin**, **rcp**, and **rcmd** commands use this file to verify access privileges. "Network Databases" in Chapter 2 contains a section that explains how to use this file. Refer also to the *hosts.equiv*(SFF) manual page for a description of the file format.

Setting Up Anonymous ftp

You can set up a public **ftp** account on your system for remote users to transfer files. You can restrict access to certain protected directories within the **ftp** home directory. "Network Databases" in Chapter 2 contains a section about the */etc/ftpusers* file, which also describes how to set up the public **ftp** account.

Mounting a Remote Filesystem at Boot Time

You can set up your distributed filesystem so that a remote filesystem is mounted when you system boots. To do this, you can add an entry to the */etc/default/filesys* file. You must also create the directory through which your system will access the remote filesystem and ensure that your system is listed in the */etc/exports* file on the server. "Introducing NFS" and "Setting Up an NFS Client" in Chapter 5 explain in detail how this is done.

Using the Name Server

Instead of using the */etc/hosts* file for host table lookup, you can use the Berkeley Internet Name Domain (BIND) service for storing and retrieving host names and addresses. To set up this optional network service, you should read Chapter 3.

Chapter 2

TCP/IP Network Administration

This chapter covers topics related to setting up and administering your ODT-NET TCP/IP network. When you installed your system, many of these tasks were performed automatically to configure a basic networked system. If you want to customize your installation or expand your network, you should read this chapter.

If your network is not performing well, the section "Network Tuning and Troubleshooting" at the end of this chapter might provide helpful suggestions.

Kernel Configuration

The following table lists the drivers that must be included in the kernel, along with their associated device nodes.

Name	Device Node	Description
arp	/dev/inet/arp	Address Resolution Protocol
arpproc	(none)	
ip	/dev/inet/ip	Internet Protocol
icmp	/dev/inet/icmp	Internet Control Message Protocol
tcp	/dev/inet/tcp	Transmission Control Protocol
udp	/dev/inet/udp	User Datagram Protocol
llcloop	/dev/llcloop	Loopback interface
socket	/dev/socksys	Socket compatibility package
cp	/dev/socksys1	Copy protection driver
vty	/dev/ptyp*nn*	Virtual TTY driver†
ttyp	/dev/ttyp*nn*	

† The Virtual TTY driver is used by **rlogin**(TC) and **telnet**(TC). There must be one ptyp device and one ttyp device for each virtual TTY configured. Following ptyp or ttyp in the device node name is a two-digit hexadecimal number corresponding to the minor number of the device. For example, **vty** minor 0 is referenced by device node */dev/ptyp00*, and **ttyp** minor 0 is referenced by device node */dev/ttyp00*.

In addition to the drivers listed above, you may also include one or more drivers for your network interface hardware:

Name	Device Node	Description
e3A*nn*	/dev/e3A*nn*	3COM 3C501 ethernet board
e3B*nn*	/dev/e3B*nn*	3COM 3C503 ethernet board
wdn*n*	/dev/wdn*n*	Western Digital WD8003E ethernet board
sl*n*	/dev/slip	Serial Line IP interface

The character *n* in the device nodes indicates any one of the digits 0 through 3. That is, up to four boards of each type are supported. If there were two 3COM 3C503 Ethernet boards, their device nodes would be */dev/e3A0* and */dev/e3A1*.

The interrupt vectors you choose for the various Ethernet boards should be consistent with your hardware requirements.

All drivers must have references in the following files:

- An entry in */etc/conf/cf.d/mdevice*
- A file corresponding to that driver in the */etc/conf/sdevice.d* directory
- An entry in */etc/conf/cf.d/sdevice*

These drivers are normally added to the kernel configuration during installation of TCP/IP. The following display shows the information from a partial *mdevice* file:

```
ip       ocis    iSc     ip      0       23      0       256     -1
rip      ocis    iSc     rip     0       24      0       256     -1
socket   ocrwis  ic      sock    0       25      0       256     -1
ttyp     ocrwi   ict     ttyp    0       26      0       16      -1
vty      ocrwi   ic      vty     0       27      0       16      -1
arpproc  oci     iS      app     0       0       0       256     -1
e3A0     I       iScH    e3c     0       28      1       1       -1
icmp     ocis    iSc     icmp    0       29      0       256     -1
llcloop  ocis    iSc     lo_     0       30      0       256     -1
slip     s       iSc     sl      0       31      0       256     -1
tcp      ocis    iSc     tcp     0       33      0       256     -1
udp      ocis    iSc     udp     0       35      0       256     -1
```

Some of the information in this file may vary depending on the system configuration. In these cases, the numbers that are used depend on the specific system configuration and are probably different from the values shown in this example.

Column six contains the major block device number, which varies depending upon the drivers that were installed in the system and the order in which they were installed. The actual value for any given driver does not actually matter as long as each driver has a different number and the number in this file matches the major number of the device name in the */dev* directory that is supposed to refer to it. The **arpproc** module is a special case, as it has no corresponding pathname in */dev*; for this driver, the block major device number is 0.

The following is a partial *sdevice* file (comments have been removed for clarity):

```
sio     Y       1       7       1       4       3f8     3ff     0       0
sio     Y       1       7       1       3       2f8     2ff     0       0
slip    Y       256     0       0       0       0       0       0       0
socket  Y       256     0       0       0       0       0       0       0
sp      Y       0       0       0       0       0       0       0       0
spt     Y       0       0       0       0       0       0       0       0
ss      Y       0       0       0       0       0       0       0       0
str     Y       0       0       0       0       0       0       0       0
svdsp   Y       1       0       0       0       0       0       0       0
svkbd   Y       1       0       0       0       0       0       0       0
sxt     N       1       0       0       0       0       0       0       0
sy      Y       1       0       0       0       0       0       0       0
tcp     Y       256     0       0       0       0       0       0       0
timod   Y       1       0       0       0       0       0       0       0
tirdwr  Y       1       0       0       0       0       0       0       0
```

The *sdevice* file is actually assembled from component files in the directory */etc/conf/sdevice.d*. Each component file contains the line describing that driver. The "Y" or "N" in the second column indicates whether the the driver is to be linked into the kernel. Column six is the interrupt vector, which varies depending upon which cards are in the system and the vectors for which they are setup.

The format of these files is defined in **sdevice**(F) and **mdevice**(F).

Runtime Configuration of STREAMS Drivers

STREAMS configuration (linking the various STREAMS drivers and modules together) is handled by the **slink**(ADMN) program, which is normally executed at boot time by **tcp**(ADMN). The **slink** program reads the file */etc/strcf*, which contains a list of STREAMS operations to perform. Most of */etc/strcf* is the same on every system. However, under unusual circumstances, it may be necessary to edit the section of */etc/strcf* that configures the network interfaces. Examples for various types of network drivers are provided. In some cases, it is necessary to write new driver setup procedures. See **slink**(ADMN) and **strcf**(SFF) for further information.

SLIP drivers are handled automatically by the **slattach**(ADMN) command, which is invoked in the */etc/tcp* script. This portion of the script is set up during installation of the SLIP driver.

The following sections present examples of **slink** configuration commands for several different driver types.

Cloning Drivers with One Major Number per Interface

Drivers of this type, such as the 3COM 3C503 *e3B0* driver or Western Digital WD8003E *wd* driver, use cloning but do not support a method of selecting a particular network interface (such as **unit select**). Rather, this is done by allocating a separate major device number to each network interface. The **slink** function **cenet** configures an interface of this type. The command line to configure such an interface has the form:

```
cenet ip /dev/e3B0 e3B0 0
```

To add a second interface, add the following line:

```
cenet ip /dev/e3B0 e3B0 1
```

Note that the device node actually used is formed by concatenating the given device node name prefix (*/dev/e3B0*) and the given unit number (*0* or *1*). The interface name is formed in a similar manner using the supplied interface name prefix (*e3B0*) and the unit number. Thus, the first example configures an interface named *e3B0*, which accesses the device referred to by */dev/e3B0*.

Cloning Drivers Using unit select or DL_ATTACH

These drivers have only one device node and one major number, which are used for all interfaces. (The SLIP drivers are of this type, but they are a special case in that individual SLIP interfaces do not need explicit configuration in */etc/strcf*. The STREAMS configuration of SLIP drivers is handled by the **slattach**(ADMN) command, which is invoked from */etc/tcp* during system startup. The appropriate **slattach** command is automatically placed in the */etc/tcp* file during installation of TCP/IP Runtime.) The desired interface is selected using either the **unit select** or the **DL_ATTACH** primitive. (Normally, a given driver recognizes only one of these primitives.) A primitive is a type of command used to invoke a primitive operation. A primitive operation can be described as part of an interface between two programs or pieces of software. In this case, a primitive operation is a service provided by one of the protocol layers.

The **slink** functions **uenet** and **denet** configure this type of driver; **uenet** uses **unit select**, while **denet** uses **DL_ATTACH**. The command line to configure an interface of this type has the form:

```
uenet ip /dev/abc en 0
```

For a driver that uses **DL_ATTACH**, use **denet** in place of **uenet**. To configure a second interface, add the following line:

```
uenet ip /dev/abc en 1
```

The **denet** and **uenet** functions form the interface name in the same manner as does **cenet** (see previous section), but the device node name is unchanged (*/dev/abc* is open in both of these examples).

Non-Cloning Drivers

Drivers of this type have a separate device node for each minor device, with some fixed number of minor devices allocated to each network interface. The **slink** functions **senetc** and **senet** are used for this driver type. (The **senetc** function allows the specification of a convergence module.) The following command line configures such an interface:

```
senetc ip eli /dev/emd0 /dev/emd1 en0
```

If a convergence module is not required, use **senet** in place of **senetc** and omit "eli."

The last argument (*en0* in this example) gives the name by which the newly created interface is known for the purpose of performing interface-configuration operations via **ifconfig**(ADMN). For further information, refer to the section entitled "Setting Interface Parameters" later in this chapter.

Assuming that there are four minor devices assigned to each network interface, a second interface would be configured as follows:

```
senetc ip eli /dev/emd4 /dev/emd5 en1
```

Setting Interface Parameters

All network interface drivers, including the loopback interface, require that their host addresses be defined at boot time. This is done with **ifconfig**(ADMN) commands included in the */etc/tcp* shell script. These commands are normally set up automatically during installation. This configuration applies only to simple, basic configurations. For example, if you want to use the network feature of **ifconfig**, you need to edit */etc/tcp* manually and modify the **ifconfig** commands there.

ifconfig can also be used to set options for an interface at boot time. Options are set independently for each interface and apply to all packets sent using that interface. These options include disabling the use of the Address Resolution Protocol. This may be useful if a network is shared with hosts running software that does not yet provide this function. Alternatively, translations for such hosts can be set in advance or published by a UNIX System host by use of the **arp**(ADMN) command.

Local Subnetworks

In TCP/IP, the DARPA Internet support includes the concept of the subnetwork. This is a mechanism that enables several local networks to appear as a single Internet network to off-site hosts. Subnetworks are useful because they allow a site to hide the local topology, requiring only a single route in external gateways. This also means that local network numbers may be locally administered.

To set up local subnetworks, you first need to know how much of the available address space is to be partitioned. The term "address" is used here to mean the Internet host part of the 32-bit address. Sites with a class A network number have a 24-bit address space with which to work, sites with a class B network number have a 16-bit address space; and sites with a class C network number have an 8-bit address space. To define local subnets you must steal some bits from the local host address space for use in extending the network portion of the internet address.

This reinterpretation of internet addresses is done only for local networks. It is not visible to off-site hosts. For example, if your site has a class B network number, hosts on this network have an Internet address that contains the network number, 16 bits, and the host number,

another 16 bits. To define 254 local subnets, each possessing at most 255 hosts, 8 bits may be taken from the local part to be used for the subnetwork ID. (The use of subnets 0 and all-*1*'s, 255 in this example, is discouraged to avoid confusion about broadcast addresses.) New network numbers are then constructed by concatenating the original 16-bit network number with the extra 8 bits containing the local subnetwork number.

The existence of local subnetworks is communicated to the system when a network interface is configured with the **netmask** option to the **ifconfig**(ADMN) program. A network mask defines the portion of the internet address that is to be considered the network part for that network. This mask normally contains the bits corresponding to the standard network part as well as the portion of the local part that was assigned to subnets. If no mask is specified when the address is set, a mask is set according to the class of the network. For example, at Berkeley (class B network 128.32), 8 bits of the local part are reserved for defining subnetworks. Consequently, the */etc/tcp* file contains lines of the form:

```
/etc/ifconfig e3B0 netmask 0xffffff00 128.32.1.7
```

This specifies that for interface *e3B0*, the upper 24 bits of the internet address should be used in calculating network numbers (netmask 0xffffff00). The internet address of the interface is 128.32.1.7 (host 7 on network 128.32.1). Hosts *m* on subnetwork *n* of this network would then have addresses of the form 128.32.*n.m*. For example, host 99 on network 129 would have an address 128.32.129.99. For hosts with multiple interfaces, the network mask should be set for each interface, although in practice only the mask of the first interface on each network is actually used.

Internet Broadcast Addresses

The broadcast address for internet networks is defined according to RFC-919 as the address with a host part of all *1*'s. The address used by 4.2BSD was the address with a host part of *0*. The UNIX System uses the standard broadcast address (all *1*'s) by default, but allows the broadcast address to be set (with **ifconfig**) for each interface. This allows networks consisting of both 4.2BSD and UNIX System hosts to coexist while the upgrade process proceeds. In the presence of subnets, the broadcast address uses the subnet field as for normal host addresses, with the remaining host part set to *1*'s (or *0*'s, on a network that has not yet been converted). The UNIX System hosts recognize and accept packets sent to the logical-network broadcast address as well as those sent to the subnet broadcast address, and, when using an all-*1*'s broadcast, also recognize and receive packets sent to host *0* as a broadcast.

Routing

If your environment allows access to networks not directly attached to your host, you need to set up routing information to allow packets to be properly routed. Two schemes are supported by the system. The first employs the routing table management daemon **routed**(ADMN) to maintain the system routing tables. The routing daemon uses a variant of the Xerox Routing Information Protocol to maintain up-to-date routing tables in a cluster of local-area networks. By using the */etc/gateways* file, the routing daemon can also initialize static routes to distant networks. (See the next section for further discussion.) When the routing daemon is started (usually from */etc/tcp*), it reads */etc/gateways* if it exists and installs those routes defined there. It then broadcasts on each local network to which the host is attached to find other instances of the routing daemon. If any responses are received, the routing daemons cooperate in maintaining a globally consistent view of routing in the local environment. This view can be extended to include remote sites also running the routing daemon by setting up suitable entries in */etc/gateways*. See **route**(ADMN) for a more thorough discussion.

The second approach is to define a default or wildcard route to a smart gateway and depend on the gateway to provide ICMP routing redirect information to create dynamically a routing data base. This is done by adding an entry to */etc/tcp* as in the following example:

```
/etc/route add default smart-gateway 1
```

See **route**(ADMN) for more information. The system uses the default route as a last resort in routing packets to their destinations. Assuming the gateway to which packets are directed can to generate the proper routing redirect messages, the system then adds routing table entries based on the information supplied. This approach has certain advantages over the routing daemon, but it is unsuitable in an environment where there are only bridges. (For example, pseudo-gateways do not generate routing-redirect messages.) Further, if the smart gateway goes down, there is no alternative, save manual alteration of the routing table entry, to maintain service.

The system always listens to, and processes, routing redirect information, and so it is possible to combine both of the above facilities. For example, the routing table management process might be used to maintain up-to-date information about routes to geographically local networks, while employing the wildcard routing techniques for distant networks. The **netstat**(TC) program displays routing table contents as well as various routing-oriented statistics. The following example displays the contents of the routing tables:

```
netstat -r
```

Alternatively, the following shows the number of routing table entries dynamically created as a result of routing redirect messages and so forth:

```
netstat -r -s
```

Using UNIX System Machines as Gateways

Any UNIX System machine that is connected to more than one network functions as a gateway. At a gateway machine, packets received on one network that are destined for a host on another network are automatically forwarded. If a packet cannot be forwarded to the desired destination, an ICMP error message is sent to the originator of the packet. When a packet is forwarded back through the interface on which it arrived, an ICMP redirect message is sent to the source host if it is on the same network. This improves the interaction of UNIX System gateways with hosts that configure their routes via default gateways and redirects.

Local-area routing within a group of interconnected Ethernets and other such networks can be handled by **routed**(ADMN). Gateways between the ARPANET or MILNET and one or more local networks require an additional routing protocol, the Exterior Gateway Protocol (EGP), to inform the core gateways of their presence and to acquire routing information from the core. (EGP is not currently supported in this product.)

Network Servers

In the UNIX System, most of the server programs are started by a super server, called the "internet daemon." The internet daemon, */etc/inetd*, acts as a master server for programs specified in its configuration file, */etc/inetd.conf*, listening for service requests for these servers, and starting up the appropriate program whenever a request is received. The configuration file includes lines containing a service name (as found in */etc/services*), the type of socket the server expects (for example, stream or dgram), the protocol used with the socket (as found in */etc/protocols*), whether to wait for each server to complete before starting up another, the user name under which the server should run, the server program's name, and at most five arguments to pass to the server program. Some trivial services are implemented internally in **inetd**(SFF), and their servers are listed as internal. For example, an entry for the file-transfer protocol server would appear as:

```
ftp    stream    tcp    nowait    root    /etc/ftpd    ftpd
```

Consult **inetd**(ADMN) for more details on the format of the configuration file and the operation of the Internet daemon.

Network Databases

Several data files are used by the network library routines and server programs. Most of these files are host independent and updated only rarely. The following table lists the data files used.

File	Manual Reference	Use
/etc/hosts	hosts (SFF)	host names
/etc/networks	networks (SFF)	network names
/etc/services	services (SFF)	list of known services
/etc/protocols	protocols (SFF)	protocol names
/etc/hosts.equiv	rshd (ADMN)	list of "trusted" hosts
/etc/ftpusers	ftpd (ADMN)	list of "unwelcome" **ftp** users
/etc/inetd.conf	inetd (ADMN)	list of servers started by **inetd**

The files distributed are set up for ARPANET or other internet hosts. Local networks and hosts should be added to describe the local configuration. Network numbers must be chosen for each Ethernet. For sites not connected to the Internet, these can be chosen more or less arbitrarily; otherwise, the normal channels should be used for allocation of network numbers.

The /etc/hosts.equiv File

There are several files that are used to establish user equivalence. One is the */etc/hosts.equiv* file, which covers the system as a whole, except for the root account. The other is the *.rhosts* file in the individual user account's home directory. This file covers only the individual user account. (For root, this is */.rhosts*.) These two files work together with a third file, */etc/passwd*, to determine the extent of user equivalence.

There are two ways to establish user equivalence:

- An entry in *.rhosts* and in */etc/passwd*
- An entry in */etc/hosts.equiv* and in */etc/passwd*

In both cases, */etc/passwd* must contain an entry for the user name from the remote machine. However, the two methods have differing scopes. If the file *.rhosts* is used in a particular account, then user equivalence is established for that account only. However, if there is an entry in */etc/hosts.equiv* for a host name and an account on that host, then that account has user equivalence for any account (except root). If the entry in */etc/hosts.equiv* has only the remote host name, then any user on that host has user equivalence for all local accounts (except root). Such a host is considered a "trusted host."

NOTE: Entries in */etc/hosts.equiv* can create large holes in system security. Be sparing in their use. In most circumstances, it is unwise to create entries that allow all users on remote machines to access all accounts on your local machine.

For example, suppose you have an account under the user name "Test1" on machine "Admin." You want to establish user equivalence on the remote machine "Systemb." The administrator for the machine Systemb must add an entry to the */etc/passwd* file for an account name Test1. They must also include the following entry in the file */etc/hosts.equiv* on Systemb:

```
Admin   Test1
```

This gives user equivalence for all accounts except root to user Test1 on the machine Systemb. Suppose that Test1 really only needed access to the account Testb on Systemb. Then it would be better to remove the above entry from */etc/hosts.equiv* on Systemb and use the following entry in the file *.rhosts* in the home directory for Testb:

```
Admin   Test1
```

Note that entries for *.rhosts* must include both the system name and the account name. The file */etc/hosts.equiv* does allow entries for the system name only, as discussed earlier.

If there are entries in both *.rhosts* and */etc/hosts.equiv* for the same machine or machine/account combination, then the entry from */etc/hosts.equiv* determines the extent of user equivalence.

The /etc/ftpusers File

The **ftp** server included in the system provides support for an anonymous **ftp** account. Because of the inherent security problems with such a facility, you should read this section carefully if you want to provide such a service.

An anonymous account is enabled by creating a user called **ftp**. When a client uses the anonymous account, a **chroot**(ADM) system call is performed by the server to restrict the client from moving outside that part of the filesystem where the **ftp** home directory is located. Because a **chroot** call is used, certain programs and files used by the server process

must be placed in the **ftp** home directory. Further, you must be sure that all directories and executable images are unwritable. The following directory setup is recommended:

```
# cd ~ftp
# chmod 555 .; chown ftp .; chgrp ftp .
# mkdir bin etc pub lib dev
# chown root bin etc lib dev
# chmod 555 bin etc lib dev
# chown ftp pub
# chmod 777 pub
# cd bin
# cp /bin/sh /bin/ls .
# chmod 111 sh ls
# cd ../etc
# cp /etc/passwd /etc/group .
# chmod 444 passwd group
# cd ../lib
# cp /shlib/libc_s .
# cd ..
# find /dev/socksys -print | cpio -dumpv .
```

When local users want to place files in the anonymous area, they must place them in a subdirectory. In the setup here, the directory *~ftp/pub* is used.

Another issue to consider is the */etc/passwd* file placed here. It can be copied by users who use the anonymous account. They can then try to break the passwords of users on your machine for further access. A good choice of users to include in this copy might be root, daemon, uucp, and the **ftp** user. All passwords here should probably be *.

Aside from the problems of directory modes and such, the **ftp** server provides a loophole for interlopers if certain user accounts are allowed. The file */etc/ftpusers* is checked on each connection. If the requested user name is located in the file, the request for service is denied. It is suggested that this file contain at least the following names:

```
uucp
root
```

Accounts with nonstandard shells should be listed in this file. Accounts without passwords need not be listed in this file; the **ftp** server does not service these users.

Network Tuning and Troubleshooting

It is likely that from time to time you will encounter problems using your network. The first thing to do is check your network connections. On networks such as the Ethernet a loose cable tap or poorly placed power cable can result in severely deteriorated service. The **ping**(ADMN) command is particularly useful for confirming the existence of network connections. If there is no hardware problem, check next for routing problems and addressing problems.

The **netstat**(TC) program can also be helpful in tracking down hardware malfunctions. In particular, look at the **-i** and **-s** options in the manual page. The **netstat**(TC) program also shows detailed information about network behavior. Examples of **netstat** displays appear later in this chapter.

If you think a communication protocol problem exists, consult the protocol specifications and attempt to isolate the problem in a packet trace. The **SO_DEBUG** option can be supplied before establishing a connection on a socket, in which case the system traces all traffic and internal actions (such as timers expiring) in a circular trace buffer. This buffer can then be printed out with the **trpt**(ADMN) program. Most of the servers distributed with the system accept a **-d** option forcing all sockets to be created with debugging turned on. Consult the appropriate manual pages for more information.

STREAMS Tuning

The **crash**(ADM) command can be used to display STREAMS usage of buffers of various sizes. Typical symptoms of inadequate STREAMS buffer space include the following: lost connections for no reason; processes that communicate over the network hang; and programs that communicate over the network suddenly malfunction. Use the UNIX Link Kit **configure** command to increase STREAMS buffer resources.

Active Connections Display

The active connections display is the default display of the **netstat**(TC) command. It displays a line of information for each active connection on the local machine under the headings described below.

netstat -a

Active Internet connections (including servers) are as follows:

```
scobox$ netstat -a
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address       Foreign Address  (state)
ip         0      0  *.*                 *.*
tcp        0      0  scobox.telnet       scoter.2460      ESTABLISHED
tcp        0      0  *.smtp              *.*              LISTEN
tcp        0      0  *.1024              *.*              LISTEN
tcp        0      0  *.sunrpc            *.*              LISTEN
tcp        0      0  *.chargen           *.*              LISTEN
tcp        0      0  *.daytime           *.*              LISTEN
tcp        0      0  *.time              *.*              LISTEN
tcp        0      0  *.domain            *.*              LISTEN
tcp        0      0  *.finger            *.*              LISTEN
tcp        0      0  *.exec              *.*              LISTEN
tcp        0      0  *.ftp               *.*              LISTEN
tcp        0      0  *.telnet            *.*              LISTEN
tcp        0      0  *.login             *.*              LISTEN
tcp        0      0  *.shell             *.*              LISTEN
tcp        0      0  scobox.listen       *.*              LISTEN
tcp        0      0  scobox.nterm        *.*              LISTEN
tcp        0      0  *.*                 *.*              CLOSED
udp        0      0  *.1035              *.*
udp        0      0  *.1034              *.*
udp        0      0  *.1033              *.*
udp        0      0  *.1032              *.*
udp        0      0  *.2049              *.*
udp        0      0  *.1028              *.*
udp        0      0  *.sunrpc            *.*
udp        0      0  scobox.domain       *.*
udp        0      0  localhost.domain    *.*
scobox$
```

Descriptions of the Display Headings

- The protocol used in the connection.
- Receive queue. The number of received characters (bytes) of data waiting to be processed.
- Send queue. The number of characters (bytes) of data waiting to be transmitted.
- The port number of the local connection, displayed symbolically. The port numbers are taken from the */etc/services* file.
- The port number of the remote connection, displayed symbolically. The port numbers are taken from the */etc/services* file.
- The current state of the connection. Each protocol has its own set of states. For the protocol-dependent states that can be displayed, see the appropriate protocol specification.

Interfaces

This display describes activities on all the local machine's interfaces to the net, in the form of a table of cumulative statistics. This display is available through **netstat** with the **-i** option.

netstat -i

```
scobox$ netstat -i
Name Mtu  Network     Address      Ipkts   Ierrs Opkts   Oerrs Collis
en0  1500 sco-eng-ne  scobox       No Statistics Available
e3B0 1500 128.174.14  128.174.14.1 0       0     0       0     0
lo0  2048 loopback    localhost    189     0     189     0     0
scobox$
```

Descriptions of the Display Headings

Each interface is described by a line with the following headings:

Name	The name of the network interface. For example, *en0* is the name of the first Ethernet interface board.
Mtu	Maximum transmission unit (in bytes). This is the largest size permitted for any single packet sent through this interface.
Network	The name of the network address of the interface as given in */etc/networks*.
Address	The name of the machine address of the interface in */etc/hosts*.
Ipkts	Input packets. The number of packets received on the interface.
Ierrs	Input errors. The number of errors detected in packets of data received on this interface.
Opkts	Output packets. The number of packets transmitted on the interface.
Oerrs	Output errors. The number of errors detected and corrected in packets of data transmitted on this interface.
Collis	Collisions that occurred on the network.

Routing Tables

The Routing Table display provides information about the usage of each route you have configured. A route consists of a destination host or network and a network interface used to exchange packets. Direct routes are created for each interface attached to the local host.

netstat -r

```
scobox$ netstat -r
Routing tables
Destination     Gateway          Flags    Refcnt Use     Interface
localhost       localhost        UH       4      0       lo0
sco-eng-net     scobox           U        4      537     en0
128.174.14      128.174.14.1     U        0      0       e3B0
128.174         scoffle          UG       0      0       en0
scobox$
```

Descriptions of the Display Headings

The information displayed for each route is as follows.

Destination — The network or machine to which this route allows you to connect.

Gateway — The name of the gateway you configured for this route. If you are directly connected, this is a local address. Otherwise, it is the name of the machine through which packets must be routed.

Flags — The state of the route. Valid states are:

U	up
G	a route to a gateway
N	a route to a network
H	a route to a host

Refcnt — The current number of active connections using the route. Connection-oriented protocols normally hold on to a single route for the duration of the connection, while connectionless protocols obtain a route and then discard it as needed.

Use — The current number of packets sent using this route.

Interface — The name of the physical network interface used to begin the route.

Statistics Display

The Protocol Statistics display provides protocol-specific errors. The errors in the display are grouped under headings for each higher-level protocol in your system. The headings are protocol-specific.

- Internet Protocol (ip)
- Internet Control Message Protocol (icmp)
- Transmission Control Protocol (tcp)
- User Datagram Protocol (udp)

netstat -s

```
ip:
        3209 total packets received
        0 bad header checksums
        0 with size smaller than minimum
        0 with data size < data length
        0 with header length < data size
        0 with data length < header length
        0 fragments received
        0 fragments dropped (dup or out of space)
        0 fragments dropped after timeout
        0 packets forwarded
        0 packets not forwardable
        0 redirects sent
icmp:
        1 call to icmp_error
        0 errors not generated because old message was icmp
        Output histogram:
                destination unreachable: 1
```

(Continued on next page.)

(Continued)

```
        0 messages with bad code fields
        0 messages < minimum length
        0 bad checksums
        0 messages with bad length
        Input histogram:
                destination unreachable: 640
        0 message responses generated
tcp:
        348 packets sent
                202 data packets (3661 bytes)
                0 data packets (0 bytes) retransmitted
                101 ack-only packets (60 delayed)
                0 URG only packets
                0 window probe packets
                0 window update packets
                45 control packets
        411 packets received
                233 acks (for 3654 bytes)
                19 duplicate acks
                0 acks for unsent data
                200 packets (1677 bytes) received in-sequence
                0 completely duplicate packets (0 bytes)
                0 packets with some dup. data (0 bytes duped)
                9 out-of-order packets (0 bytes)
                0 packets (0 bytes) of data after window
                0 window probes
                0 window update packets
                0 packets received after close
                0 discarded for bad checksums
                0 discarded for bad header offset fields
                0 discarded because packet too short
        25 connection requests
        12 connection accepts
        21 connections established (including accepts)
        72 connections closed (including 0 drops)
        16 embryonic connections dropped
        233 segments updated rtt (of 259 attempts)
        0 retransmit timeouts
                0 connections dropped by rexmit timeout
```

(Continued on next page.)

(Continued)

```
        0 persist timeouts
        0 keepalive timeouts
                0 keepalive probes sent
                0 connections dropped by keepalive
        0 connections lingered
                0 linger timers expired
                0 linger timers cancelled
                0 linger timers aborted by signal
udp:
        0 incomplete headers
        0 bad data length fields
        0 bad checksums
```

Chapter 3

Name Server Operations Guide for BIND

A name server is a network service that enables clients to name resources or objects and share this information with other objects in the network. The Berkeley Internet Name Domain (BIND) Server implements the DARPA Internet name server for the UNIX operating system. In effect, this is a distributed database system for objects in a computer network. BIND is fully integrated into network programs for use in storing and retrieving host names and addresses. The system administrator can configure the system to use BIND as a replacement for the original host table lookup of information in the network hosts file */etc/hosts*. The default configuration does not use BIND. BIND is initially disabled. If you want to use it, you must first set up the necessary configuration files.

The Name Service

The basic function of the name server is to provide information about network objects by answering queries. The advantage of using a name server over the host table lookup for host-name resolution is to avoid the need for a single centralized clearinghouse for all names. The authority for this information can be delegated to the different organizations on the network responsible for it.

The host table lookup routines require that the master file for the entire network be maintained at a central location by a few people. This works well for small networks where there are only a few machines and the different organizations responsible for them cooperate. However, this does not work well for large networks where machines cross organizational boundaries.

With the name server, the network can be broken into a hierarchy of domains. The name space is organized as a tree, according to organizational or administrative boundaries. Each node, called a domain, is given a label, and the name of the domain is the concatenation of all the labels of the domains from the root to the current domain, listed from right to left, separated by dots. A label need only be unique within its domain. The whole space is partitioned into several areas called zones, each starting at a domain and extending down to the leaf domains or to domains where other zones start. Zones usually represent

administrative boundaries. An example of a host address for a host at the University of California, Berkeley, would look as follows:

```
monet.Berkeley.EDU
```

The top-level domain for educational organizations is EDU; Berkeley is a subdomain of EDU and monet is the name of the host. Additional top-level domains include:

COM	Commercial Organizations
GOV	Government Organizations
MIL	Military Departments
ORG	Miscellaneous Organizations

Types of Servers

There are several types of servers. These are:

- master servers
- caching-only servers
- remote servers
- slave servers

These types of servers are described in more detail in the following four sections.

Master Servers

A master server for a domain is the authority for that domain. This server maintains all the data corresponding to its domain. Each domain should have at least two master servers: a primary master, and some secondary masters to provide backup service if the primary is unavailable or overloaded. A server may be a master for multiple domains, being primary for some domains and secondary for others.

Primary

A primary master server is a server that loads its data from a file on disk. This server may also delegate authority to other servers in its domain.

Secondary

A secondary master server is a server that is delegated authority and receives its data for a domain from a primary master server. At boot time, the secondary server requests all the data for the given zone from the primary master server. This server then periodically checks with the primary server to see if it needs to update its data.

Caching-Only Servers

All servers are caching servers. This means that the server caches the information that it receives for use until the data expires. A caching only server is a server that is not authoritative for any domain. This server services queries and asks other servers that have the authority for the information needed. All servers keep data in their caches until the data expires, based on a time-to-live field attached to the data when it is received from another server.

Remote Servers

A remote server is an option given to people who would like to use a name server on their workstation or on a machine that has a limited amount of memory and CPU cycles. With this option, you can run all of the networking programs that use the name server without running the name server on the local machine. All of the queries are serviced by a name server that is running on another machine on the network.

Slave Server

A slave server is a server that always forwards queries it cannot satisfy locally to a fixed list of forwarding servers instead of interacting with the master name servers for the root and other domains. The queries to the forwarding servers are recursive queries. There may be one or more forwarding servers, and they are tried in turn until the list is exhausted. A slave and forwarder configuration is typically used when you do not wish all the servers at a given site to be interacting with the rest of the Internet servers. A typical scenario would involve a number of workstations and a departmental timesharing machine with Internet access. The workstations might be administratively prohibited from having Internet access. To give the workstations the appearance of access to the Internet domain system, the workstations could

be slave servers to the timesharing machine, which would forward the queries and interact with other name servers to resolve the query before returning the answer. An added benefit of using the forwarding feature is that the central machine develops a much more complete cache of information that all the workstations can take advantage of. The use of slave mode and forwarding is discussed further under the description of the named bootfile commands.

Setting Up Your Own Domain

When setting up a domain that is going to be on a public network, the site administrator should contact the organization in charge of the network and request the appropriate domain registration form. An organization that belongs to multiple networks (such as CSNET, DARPA Internet, and BITNET) should register with only one network.

The contacts are as follows:

DARPA Internet

Sites that are already on the DARPA Internet and need information on setting up a domain should contact HOSTMASTER@SRI-NIC.ARPA. You may also want to be placed on the BIND mailing list, which is a mail group for people on the DARPA Internet running BIND. This group discusses future design decisions, operational problems, and other related topics. To request placement on this mailing list, send mail to the following address:

```
bind-request @ucbarpa.Berkeley.EDU.
```

CSNET

A CSNET member organization that has not registered its domain name should contact the CSNET Coordination and Information Center (CIC) for an application and information about setting up a domain.

An organization that already has a registered domain name should keep the CIC informed about how it would like its mail routed. In general, the CSNET relay prefers to send mail via CSNET if possible (as opposed to BITNET or the Internet). For an organization on multiple networks, this may not always be the preferred behavior. The CIC can be reached via electronic mail at cic @ sh.cs.net, or by phone at (617) 497-2777.

BITNET

If you are on the BITNET and need to set up a domain, contact INFO@BITNIC.

Boot File

The name server uses several files to load its database. The major file used is the boot file. This is the file that is first read when **named** starts up. This tells the server what type of server it is, which zones it has authority over, and where to get its initial data. The default location for this file is */etc/named.boot*. However, this can be changed by setting the BOOTFILE variable when you compile **named** or by specifying the location on the command line when **named** starts up.

Domain

The boot file contains a line of code that designates the default domain. The line for the server looks like this:

```
domain          Berkeley.Edu
```

The name server uses this information when it receives a query for a name without a "." that is unknown. When it receives one of these queries, it appends the name in the second field to the query name. This is an obsolete facility, which will be removed from future releases.

Directory

The directory line specifies the directory in which the name server should run, allowing the other filenames in the boot file to use relative pathnames.

```
directory               /usr/local/lib/named
```

If you have more than a couple of named files to be maintained, you may wish to place the named files in a directory such as */usr/local/domain* and adjust the directory command properly. The main purposes of this command are to make sure **named** is in the proper directory when trying to include files by relative pathnames with $INCLUDE and to allow **named** to run in a location that is reasonable to dump core if it feels the urge.

Primary Master

The line in the boot file that designates the server as a primary server for a zone looks like the following:

```
primary             Berkeley.Edu    ucbhosts
```

The first field specifies that the server is a primary one for the zone stated in the second field. The third field is the name of the file from which the data is read.

Secondary Master

The line for a secondary server is similar to that for the primary, except that it lists addresses of other servers (usually primary servers) from which the zone data is obtained.

```
secondary           Berkeley.Edu    128.32.0.10     128.32.0.4
```

The first field specifies that the server is a secondary master server for the zone stated in the second field. The two network addresses specify the name servers that are primary for the zone. The secondary server gets its data across the network from the listed servers. Each server is tried in the order listed until it successfully receives the data from a listed server. If a filename is present after the list of primary servers, data for the zone is dumped into that file as a backup. When the server is first started, the data are loaded from the backup file if possible, and a primary server is then consulted to check that the zone is still up-to-date.

Caching-Only Server

You do not need a special line to designate that a server is a caching server. A caching-only server is indicated by the absence of authority lines, such as secondary or primary in the boot file.

All servers should have the following line in the boot file to prime the name server's cache:

```
cache               .               root.cache
```

The period (.) specifies the current domain. All cache files listed are read in at named boot time and any values still valid are reinstated in the cache and the root name server information in the cache files are always used. For information on the cache file, see the later section, "Initializing the Cache."

Forwarders

Any server can make use of forwarders. A forwarder is another server capable of processing recursive queries to try to resolve queries on behalf of other systems. The **forwarders** command specifies forwarders by internet address as follows:

```
forwarders          128.32.0.10 128.32.0.4
```

There are two main reasons for wanting to do so. First, the other systems may not have full network access and may be prevented from sending any IP packets into the rest of the network and, therefore, must rely on a forwarder that does have access to the full net. The second reason is that the forwarder sees a union of all queries as they pass through the forwarder's server and, therefore, the forwarder builds up a very rich cache of data compared to the cache in a typical workstation name server. In effect, the forwarder becomes a meta-cache that all hosts can benefit from, thereby reducing the total number of queries from that site to the rest of the net.

Slave Mode

Slave mode is used if the use of forwarders is the only possible way to resolve queries because of lack of full net access or if you wish to prevent the name server from using other than the listed forwarders. Slave mode is activated by placing the simple command

slave

in the bootfile. If **slave** is used, then you must specify forwarders. When in slave mode, the server forwards each query to each of the forwarders until an answer is found or the list of forwarders is exhausted.

Remote Servers

To set up a host that uses a remote server instead of a local server to answer queries, create the file */etc/resolv.conf*. This file designates the name servers on the network that should be sent queries. It is not advisable to create this file if you have a local server running. If this file exists, it is read almost every time **gethostbyname**(SLIB) or **gethostbyaddr** is called.

Initializing the Cache

The name server needs to know the identities of the authoritative name servers for the root domain of the network. To do this, you have to prime the name server's cache with the address of these higher authorities. This is done in a file called *root.cache*. The location of this file is specified in the boot file */etc/named.boot*.

There are three standard files used to specify the data for a domain. These files are:

named.local
hosts
host.rev.

The *named.local* file specifies the address for the local loopback interface, better known as *localhost*, with the network address 127.0.0.1. The location of this file is specified in the boot file.

The *hosts* file contains all the data about the machines in this zone. The location of this file is specified in the boot file.

The *hosts.rev* file specifies the IN-ADDR.ARPA domain. This is a special domain for allowing address-to-name mapping. Because Internet host addresses do not fall within domain boundaries, this special domain was formed to allow inverse mapping. The IN-ADDR.ARPA domain has four labels preceding it. These labels correspond to the four octets of an Internet address. All four octets must be specified even if an octet is zero. The Internet address 128.32.0.4 is located in the domain 4.0.32.128.IN-ADDR.ARPA. This reversal of the address is awkward to read but allows for the natural grouping of hosts in a network.

Standard Resource Records

The records in the name server data files are called resource records. The following is a general description of a resource record:

```
{name}    {ttl}    addr-class    Record Type    Record Specific data
```

Resource records have a standard format, as shown above. The first field is always the name of the domain record and it must always start in column 1. For some resource records, the name can be left blank. In such cases, the name of the previous resource record is used. The second field is an optional time-to-live field. This specifies how long this data is stored in the database. When this field is left blank, the default time-to-live is specified in the Start of Authority resource record discussed later in this chapter. The third field is the address class. There are currently two classes: IN for internet addresses and ANY for all address classes.

The fourth field states the type of the resource record. The fields after that are dependent on the type of the resource record. Case is preserved in names and data fields when loaded into the name server. All comparisons and lookups in the name server database are case-insensitive.

The following characters have special meanings:

.	A free-standing dot in the name field refers to the current domain.
@	A free-standing @ in the name field denotes the current origin.
..	Two free-standing dots represent the null domain name of the root when used in the name field.
\X	Where X is any character other than a digit (0-9), \X quotes that character so that its special meaning does not apply. For example, "\." can be used to place a dot character in a label.
\DDD	Where each D is a digit, \DDD is the octet corresponding to the decimal number described by DDD. The resulting octet is assumed to be text and is not checked for special meaning.
( )	Parentheses are used to group data that crosses a line. In effect, line terminations are not recognized within parentheses.
;	A semicolon starts a comment; the remainder of the line is ignored.
*	An asterisk signifies a wildcard.

Most resource records have the current origin appended to names if they are not terminated by a ".". This is useful for appending the current domain name to the data, such as machine names, but can cause problems where you do not want this to happen. The following is a good rule of thumb: if the name is not in the domain for which you are creating the data file, end the name with a ".".

Separating Data into Multiple Files

An include line begins with $INCLUDE (starting in column 1) and is followed by a file name. This feature is particularly useful for separating different types of data into multiple files. Here is an example:

```
$INCLUDE /usr/named/data/mailboxes
```

The line would be interpreted as a request to load the file */usr/named/data/mailboxes*. The $INCLUDE command does not cause data to be loaded into a different zone or tree. This is simply a way to allow data for a given zone to be organized in separate files. For example, mailbox data might be kept separately from host data using this mechanism.

Changing an Origin in a Data File

Use the $ORIGIN command to change the origin in a data file. The line starts in column 1 and is followed by a domain origin. This is useful for putting more than one domain in a data file. For example, */etc/named.hosts* might contain lines of the form:

```
$ORIGIN CC.Berkeley.EDU
[assorted domain data...]
$ORIGIN EE.Berkeley.EDU
[assorted domain data...]
```

The Start of Authority Resource Record (SOA)

The Start of Authority record designates the start of a zone. An SOA record includes the following fields:

- Name
- Origin
- Person in charge
- Serial number
- Refresh
- Retry
- Expire
- Minimum

"Name" is the name of the zone. "Origin" is the name of the host on which this data file resides. "Person in charge" is the mailing address for the person responsible for the name server. "Serial number" is the version number of this data file; this number should be incremented whenever a change is made to the data. (Note that the name server cannot handle numbers over 9999 after the decimal point.) "Refresh" indicates how often, in seconds, a secondary name server is to check with the primary name server to see if an update is needed. "Retry" indicates how long, in seconds, a secondary server is to retry after a failure to check for a refresh. "Expire" is the upper time limit, in seconds, that a secondary name server is to use the data before it expires for lack of getting a refresh. Minimum is the default number of seconds to be used for the time-to-live field on resource records. There should only be one SOA record per zone. Here is an example of an SOA record:

```
name  {ttl}  addr-class  SOA     Origin              Person in charge
@            IN          SOA     ucbvax.Berkeley.Edu.  kjd.ucbvax.Berkeley.Edu. (
                         1.1     ; Serial
                         3600    ; Refresh
                         300     ; Retry
                         3600000 ; Expire
                         3600 )  ; Minimum
```


The Name Server Resource Record (NS)

The name server record (NS) lists a name server responsible for a given domain. The first name field lists the domain that is serviced by the listed name server. There should be one NS record for each primary master server for the domain. Here is an example of a name server record:

```
{name}    {ttl}    addr-class    NS    Name servers name
                   IN            NS    ucbarpa.Berkeley.Edu.
```

The address class is IN (Internet addresses), and the record type is name server (NS). The record uses the default ttl (time-to-live) value. Here, the record-specific data is the identity of the name server.

The Address Resource Record (A)

The address record (A) lists the address for a given machine. The name field is the machine name and the address is the network address. There should be one A record for each address

of the machine. Here is an example of an address record for a machine named *ucbarpa* with two network addresses:

```
{name}     {ttl}    addr-class    A    address
ucbarpa             IN            A    128.32.0.4
                    IN            A    10.0.0.78
```

The Host Information Resource Record (HINFO)

The host information resource record (HINFO) is for host-specific data. It lists the hardware and operating system that are running at the listed host. It should be noted that only a single space separates the hardware information and the operating-system information. If you want to include a space in the machine name, you must quote the name. Host information is not specific to any address class, so ANY may be used for the address class. There should be one HINFO record for each host. Here is an example:

```
{name}    {ttl}    addr-class    HINFO    Hardware      OS
                   ANY           HINFO    VAX-11/780    UNIX
```

Note that the current release ignores any records that appear after an HINFO record. Thus, you can use only one HINFO record within the file, and it should be the last record in the file.

The Well-Known Services Resource Record (WKS)

The well-known services record (WKS) describes the well-known services supported by a particular protocol at a specified address. The list of services and port numbers comes from the list of services specified in */etc/services*. There should be only one WKS record per protocol per address. Here is an example of a WKS record:

```
{name}  {ttl} addr-class  WKS   address     protocol  list of services
              IN          WKS   128.32.0.10  UDP       who route timed domain
              IN          WKS   128.32.0.10  TCP       (echo telnet
                                                       discard sunrpc sftp
                                                       uucp-path systat daytime
                                                       netstat qotd nntp
                                                       link chargen ftp
                                                       auth time whois mtp
                                                       pop rje finger smtp
                                                       supdup hostnames
                                                       domain
                                                       name server)
```

The Canonical Name Resource Record (CNAME)

The canonical name resource record (CNAME) specifies an alias for a canonical name. An alias should be the only record associated with the alias name; all other resource records should be associated with the canonical name and not with the alias. Any resource records that include a domain name as their value (for example, NS or MX) should list the canonical name, not the alias. Here is an example of a CNAME record:

```
aliases     {ttl}   addr-class   CNAME   Canonical name
ucbmonet            IN           CNAME   monet
```

The Domain Name Pointer Resource Record (PTR)

A domain name pointer record (PTR) allows special names to point to some other location in the domain. The following example of a PTR record is used in setting up reverse pointers for the special IN-ADDR.ARPA domain. This line is from the example:

```
hosts.rev file.
```

In this record, the name field is the network number of the host in reverse order. You only need to specify enough octets to make the name unique. For example, if all hosts are on network 127.174.14, then only the last octet needs to be specified. If hosts are on networks 128.174.14 and 127.174.23, then the last two octets need to be specified. PTR names should be unique to the zone. Here is an example of a PTR record:

```
name    {ttl}   addr-class   PTR   real name
7.0             IN           PTR   monet.Berkeley.Edu.
```

The Mailbox Resource Record (MB)

The mailbox resource record has a record type of MB. It lists the machine where a user wants to receive mail. The name field is the user's login; the machine field denotes the machine to which mail is to be delivered. Mail box names should be unique to the zone. Here is an example of an MB record:

```
name      {ttl}   addr-class    MB    Machine
miriam            IN            MB    vineyd.DEC.COM.
```

The Mail Rename Resource Record (MR)

The mail rename record (MR) can be used to list aliases for a user. The name field lists the alias for the name listed in the fourth field, which should have a corresponding MB record. Here is an example of a mail rename record:

```
name            {ttl}   addr-class    MR    corresponding MB
Postmistress            IN            MR    miriam
```

The Mailbox Information Resource Record (MINFO)

The mail information record MINFO creates a mail group for a mailing list. This resource record is usually associated with a mail group, but it can be used with a mail box record. The "name" specifies the name of the mailbox. The "requests" field is where mail such as requests to be added to a mail group should be sent. The "maintainer" is a mailbox that should receive error messages. This is particularly appropriate for mailing lists when errors in members' names should be reported to a person other than the sender. Here is an example of this record:

```
name    {ttl}   addr-class    MINFO    requests        maintainer
BIND            IN            MINFO    BIND-REQUEST    kjd.Berkeley.Edu.
```

The Mail Group Member Resource Record (MG)

The mail group record (MG) lists members of a mail group.

```
{mail group name}    {ttl}    addr-class    MG    member name
                              IN            MG    Bloom
```

An example for setting up a mailing list is as follows:

```
Bind        IN    MINFO    Bind-Request              kjd.Berkeley.Edu.
            IN    MG       Ralph.Berkeley.Edu.
            IN    MG       Zhou.Berkeley.Edu.
            IN    MG       Painter.Berkeley.Edu.
            IN    MG       Riggle.Berkeley.Edu.
            IN    MG       Terry.pa.Xerox.Com.
```

The Mail Exchanger Resource Record (MX)

```
name           {ttl}  addr-class  MX  preference value  mailer exchanger
Munnari.OZ.AU.  IN                MX  0                 Seismo.CSS.GOV.
*.IL.           IN                MX  0                 RELAY.CS.NET.
```

Mail exchanger records (MX) are used to identify a machine that knows how to deliver mail to a machine that is not directly connected to the network. In the first example above, Seismo.CSS.GOV. is a mail gateway that knows how to deliver mail to Munnari.OZ.AU. but other machines on the network cannot deliver mail directly to Munnari. These two machines may have a private connection or use a different transport medium. The preference value is the order that a mailer should follow when there is more then one way to deliver mail to a single machine. See RFC974 for more detailed information.

Wildcard names containing the character "*" may be used for mail routing with MX records. There are likely to be servers on the network that simply state that any mail to a domain is to be routed through a relay. In the second example above, all mail to hosts in the domain IL is routed through RELAY.CS.NET. This is done by creating a wildcard resource record, which states that *.IL has an MX of RELAY.CS.NET.

Some Sample Files

The following sections contain sample files for the name server. This covers example boot files for the different types of server and example domain database files.

Caching-Only Server

```
;
; Boot file for Caching Only Name Server
;

; type    domain              source file or host
;
domain    Berkeley.Edu
cache     .                   /etc/named.ca
primary   0.0.127.in-addr.arpa   /etc/named.local
```

Primary Master Server

```
;
; Boot file for Primary Master Name Server
;

; type      domain                source file or host
;
directory   /usr/local/lib/named
primary     Berkeley.Edu          ucbhosts
primary     32.128.in-addr.arpa   ucbhosts.rev
primary     0.0.127.in-addr.arpa  named.local
cache       .                     root.cache
```

Secondary Master Server

```
;
; Boot file for Secondary Name Server
;
; type      domain                source file or host
;
directory   /usr/local/lib/named
secondary   Berkeley.Edu          128.32.0.4 128.32.0.10 128.32.136.22 ucbhost.bak
secondary   32.128.in-addr.arpa   128.32.0.4 128.32.0.10 128.32.136.22 ucbhosts.rev.bak
primary     0.0.127.in-addr.arpa  named.local
cache       .                     root.cache
```

The /etc/resolv.conf File

```
domain Berkeley.Edu
name server 128.32.0.4
name server 128.32.0.10
```

root.cache

```
;
; Initial cache data for root domain servers.
;
.                   99999999   IN   NS   SRI-NIC.ARPA.
                    99999999   IN   NS   NS.NASA.GOV.
                    99999999   IN   NS   TERP.UMD.EDU.
                    99999999   IN   NS   A.ISI.EDU.
                    99999999   IN   NS   BRL-AOS.ARPA.
                    99999999   IN   NS   GUNTER-ADAM.ARPA.
                    99999999   IN   NS   C.NYSER.NET.
;  Prep the cache (hotwire the addresses).
SRI-NIC.ARPA.       99999999   IN   A    10.0.0.51
SRI-NIC.ARPA.       99999999   IN   A    26.0.0.73
NS.NASA.GOV.        99999999   IN   A    128.102.16.10
BRL-AOS.ARPA.       99999999   IN   A    128.20.1.2
A.ISI.EDU.          99999999   IN   A    26.3.0.103
BRL-AOS.ARPA.       99999999   IN   A    192.5.25.82
GUNTER-ADAM.ARPA.   99999999   IN   A    26.1.0.13
C.NYSER.NET.        99999999   IN   A    128.213.5.17
TERP.UMD.EDU.       99999999   IN   A    10.1.0.17
```

named.local

```
@     IN    SOA    ucbvax.Berkeley.Edu. kjd.ucbvax.Berkeley.Edu. (
                   1        ; Serial
                   10800      ; Refresh
                   1800       ; Retry
                   3600000 ; Expire
                   86400 )    ; Minimum
      IN    NS     ucbvax.Berkeley.Edu.
1     IN    PTR    localhost.
```

hosts

```
;
;     @(#)ucb-hosts     1.1     (berkeley)     86/02/05
;

@                  IN    SOA      ucbvax.Berkeley.Edu. kjd.monet.Berkeley.Edu. (
                                  1.1        ; Serial
                                  3600       ; Refresh
                                  300        ; Retry
                                  3600000    ; Expire
                                  3600 )     ; Minimum
                   IN    NS       ucbarpa.Berkeley.Edu.
                   IN    NS       ucbvax.Berkeley.Edu.
localhost          IN    A        127.1
ucbarpa            IN    A        128.32.4
                   IN    A        10.0.0.78
                   IN    HINFO    VAX-11/780 UNIX
arpa               IN    CNAME    ucbarpa
ernie              IN    A        128.32.6
                   IN    HINFO    VAX-11/780 UNIX
ucbernie           IN    CNAME    ernie
monet              IN    A        128.32.7
                   IN    A        128.32.130.6
                   IN    HINFO    VAX-11/750 UNIX
ucbmonet           IN    CNAME    monet
```

```
ucbvax          IN   A       10.2.0.78
                IN   A       128.32.10
                IN   HINFO   VAX-11/750 UNIX
                IN   WKS     128.32.0.10 UDP syslog route timed domain
                IN   WKS     128.32.0.10 TCP ( echo telnet
                             discard sunrpc sftp
                             uucp-path systat daytime
                             netstat qotd nntp
                             link chargen ftp
                             auth time whois mtp
                             pop rje finger smtp
                             supdup hostnames
                             domain
                             name server )
vax             IN   CNAME   ucbvax
toybox          IN   A       128.32.131.119
                IN   HINFO   Pro350 RT11
toybox          IN   MX      0  monet.Berkeley.Edu
miriam          IN   MB      vineyd.DEC.COM.
postmistress    IN   MR      Miriam
Bind            IN   MINFO   Bind-Request kjd.Berkeley.Edu.
                IN   MG      Ralph.Berkeley.Edu.
                IN   MG      Zhou.Berkeley.Edu.
                IN   MG      Painter.Berkeley.Edu.
                IN   MG      Riggle.Berkeley.Edu.
                IN   MG      Terry.pa.Xerox.Com.
```

hosts.rev

```
;
;    @(#)ucb-hosts.rev    1.1    (Berkeley)    86/02/05
;

@       IN   SOA   ucbvax.Berkeley.Edu. kjd.monet.Berkeley.Edu. (
                   1.1      ; Serial
                   10800    ; Refresh
                   1800     ; Retry
                   3600000  ; Expire
                   86400 )  ; Minimum
        IN   NS    ucbarpa.Berkeley.Edu.
        IN   NS    ucbvax.Berkeley.Edu.
4.0     IN   PTR   ucbarpa.Berkeley.Edu.
6.0     IN   PTR   ernie.Berkeley.Edu.
7.0     IN   PTR   monet.Berkeley.Edu.
10.0    IN   PTR   ucbvax.Berkeley.Edu.
6.130   IN   PTR   monet.Berkeley.Edu.
```

Additional Sample Files

The following sections contain an additional set of sample files for the name server.

named.boot

```
;
; Name Server boot file for Domain sco.COM
;
; Type        Domain        Source file or Host
;
domain        sco.COM
primary       sco.COM       /etc/named.data/sco-hosts
cache         .             /etc/named.data/root.cache
secondary     sco.COM       /etc/named.data/sco-host.s.rev
primary       sco.COM       /etc/named.data/named.local
```

root.cache

```
;
; Initial cached data for root domain servers.
;
;.                    99999999 IN NS      USC-ISIB.ARPA.
;                     99999999 IN NS      BRL-AOS.ARPA.
;                     99999999 IN NS      SRI-NIC.ARPA.
;
; Insert your own name servers here
;
;                     99999999 IN NS      scovert.sco.COM
;
; Prep the cache (hotwire the addresses)
;
tandy.sco.COM.              99999999 IN A192.9.200.2
;viscous.sco.COM.    99999999 IN A128.0.21.6
;
; Root servers go here
;
tandy.sco.COM.              99999999 IN A192.9.200.2
;SRI-NIC.ARPA.              99999999 IN A10.0.0.51
;USC-ISIB.ARPA.             99999999 IN A10.3.0.52
;BRL-AOS.ARPA.              99999999 IN A128.20.1.2
;BRL-AOS.ARPA.              99999999 IN A192.5.22.82
```

named.local

```
;
; Don't forget to increment the serial number in
; named.soa
;

$INCLUDE /etc/named/sco.soa
192.9.200.2   IN PTR localhost.
```

sco-host.s.rev

```
;
; Don't forget to increment the serial number in
; named.soa
;

$INCLUDE /etc/named/sco.soa

192.9.200.1   IN     PTR    merlin
192.9.200.2   IN     PTR    tandy
192.9.200.3   IN     PTR    tvi
```

sco.soa

```
;
; Don't forget to increment the serial number when you
; change this.  SCCS or RCS might be a good idea here.
;

@             IN SOA  tandy.sco.COM.    root.tandy.sco.COM.  (
                      1.0      ; Serial
                      3600     ; Refresh
                      300      ; Retry
                      3600000  ; Expire
                      3600 )   ; Minimum
              IN NS   tandy.sco.COM.
```

Domain Management

This section contains information for starting, controlling, and debugging **named**(ADMN), the Internet domain name server.

Starting the Name Server

The host name should be set to the full domain style name (that is, monet.Berkeley.EDU.) using **hostname**(TC). The name server is started automatically if the configuration file */etc/named.boot* is present. Do not attempt to run **named** from **inetd**(ADMN). This continuously restarts the name server and defeats the purpose of having a cache.

/etc/named.pid

When **named** is successfully started, it writes its process ID into the file */etc/named.pid*. This is useful to programs that want to send signals to **named**. The name of this file can be changed by defining PIDFILE to the new name when compiling **named**.

/etc/hosts

The **gethostbyname** library call can detect whether **named** is running. If it is determined that **named** is not running, it looks in */etc/hosts* to resolve an address. This option was added to allow **ifconfig**(ADMN) to configure the machine's local interfaces and to enable a system manager to access the network while the system is in single-user mode. It is advisable to put the local machine's interface addresses and a couple of machine names and addresses in */etc/hosts*, so the system manager can copy files from another machine with **rcp** when the system is in single-user mode. The format of */etc/hosts* has not changed. See **hosts**(SFF) for more information. Because the process of reading */etc/hosts* is slow, it is not advisable to use this option when the system is in multiuser mode.

Reload

There are several signals that can be sent to the **named** process to have it do tasks without restarting the process. The SIGHUP signal causes **named** to read *named.boot* and reload the database. All previously cached data is lost. This is useful when you have made a change to a data file and you want **named** 's internal database to reflect the change.

Debugging

When **named** is running incorrectly, look first in */usr/adm/syslog* and check for any messages logged by **syslog**. Next, send it a signal to see what is happening.

SIGINT dumps the current database and cache to */usr/tmp/named_dump.db* This should give you an indication as to whether the database was loaded correctly. The name of the dump file can be changed by defining DUMPFILE to the new name when compiling **named**.

NOTE: The following two signals only work when **named** *is built with DEBUG defined.*

SIGUSR1 - Turns on debugging. Each following USR1 increments the debug level. The output goes to */usr/tmp/named.run*. The name of this debug file can be changed by defining DEBUGFILE to the new name before compiling **named**.

SIGUSR2 - Turns off debugging completely.

For more detailed debugging, define DEBUG when compiling the resolver routines into */usr/lib/libsocket.a*.

Chapter 4

Synchronizing Network Clocks

The clock synchronization service is composed of a collection of time daemons (**timed**(ADMN)) running on the machines in a local-area network. The algorithms implemented by the service are based on a master-slave scheme. The time daemons communicate with each other using the Time Synchronization Protocol (TSP), which is built on the DARPA UDP protocol.

A time daemon has a two-fold function. First, it supports the synchronization of the clocks of the various hosts in a local-area network. Second, it starts (or takes part in) the election that occurs among slave time daemons when, for any reason, the master disappears. The synchronization mechanism and the election procedure employed by the program **timed** are described in the manual page **timed**(ADMN). This chapter is mainly concerned with the administrative and technical issues of running **timed** at a particular site. The next section is a brief overview of how the time daemon works.

How a Time Daemon Works

A master time daemon measures the time differences between the clock of the machine on which it is running and those of all other machines on its network. The master computes the network time as the average of the times provided by nonfaulty clocks. (A clock is considered to be faulty when its value is more than a small specified interval apart from the majority of the clocks of the other machines.) The master time daemon then sends to each slave time daemon the correction that should be performed on the clock of its machine. This process is repeated periodically.

Because the correction is expressed as a time difference rather than an absolute time, transmission delays do not interfere with the accuracy of the synchronization. When a machine comes up and joins the network, it starts a slave time daemon that asks the master for the correct time and resets the machine's clock before any user activity can begin. The time daemons are thus able to maintain a single network time in spite of the drift of clocks away from each other. The present implementation is capable of keeping processor clocks synchronized to within 20 milliseconds, but some hardware is not adjustable at less than 1 second intervals.

To ensure that the service provided is continuous and reliable, it is necessary to implement an election algorithm to elect a new master should the machine running the current master crash, the master terminate (for example, because of a runtime error), or the network be partitioned. Under this algorithm, slaves are able to realize when the master has stopped functioning and to elect a new master from among themselves. It is important to note that the failure of the master results only in a gradual divergence of clock values; thus, the election need not occur immediately.

The machines that are gateways between distinct local-area networks require particular care. A time daemon on such machines may act as a "submaster." This artifact depends on the current inability of transmission protocols to broadcast a message on a network other than the one to which the broadcasting machine is connected. The submaster appears as a slave on one network and as a master on one or more of the other networks to which it is connected.

A submaster classifies each network as one of three types. A slave network is a network on which the submaster acts as a slave. There can only be one slave network. A master network is a network on which the submaster acts as a master. An ignored network is any other network that already has a valid master. The submaster tries periodically to become master on an ignored network, but gives up immediately if a master already exists.

Guidelines

While the synchronization algorithm is quite general, the election algorithm, which requires a broadcast mechanism, puts constraints on the kind of network on which time daemons can run. The time daemon works only on networks with broadcast capability augmented with point-to-point links. Machines that are only connected to point-to-point, non-broadcast networks cannot use the time daemon.

If submasters are excluded, there is normally only one master time daemon in a local-area internetwork. During an election, only one of the slave time daemons becomes the new master. Not all machines are suitable as masters; some do not have sufficiently accurate timing mechanisms or cannot afford the extra overhead. Therefore, a subset of machines must be designated as potential master time daemons. A master time daemon requires CPU resources proportional to the number of slaves (in general, more than a slave time daemon), and so it may be advisable to limit master time daemons to machines with more powerful processors or lighter loads. Also, machines with inaccurate clocks should not be used as masters. This is a purely administrative decision; an organization may well allow all of its machines to run master time daemons.

At the administrative level, a time daemon on a machine with multiple network interfaces may be told to ignore all but one network or to ignore one network. This is done with the **timed -n** *network* and **-i** *network* options, respectively, at startup time. Typically, the time daemon would be instructed to ignore all but the networks belonging to the local administrative control.

There are some limitations to the current implementation of the time daemon. It is expected that these limitations will be removed in future releases. The constant NHOSTS in */usr/src/etc/timed/globals.h* limits the maximum number of machines that can be directly controlled by one master time daemon. The maximum is (NHOSTS - 1). Currently, the maximum is 99. The constant must be changed and the program recompiled if a site wishes to run **timed** on a larger network.

In addition, there is a pathological situation to be avoided at all costs. This situation can occur when time daemons run on multiply-connected local-area networks. In this case, time daemons running on gateway machines are submasters, and they act on some of those networks as master time daemons. Consider machines A and B that are both gateways between networks X and Y. If time daemons were started on both A and B without constraints, it would be possible for submaster time daemon A to be a slave on network X and the master on network Y, while submaster time daemon B would be a slave on network Y and the master on network X. This loop of master time daemons does not function properly or guarantee a unique time on both networks, and it causes the submasters to use large amounts of system resources in the form of network bandwidth and CPU time. In fact, this kind of loop can also be generated with more than two master time daemons, when several local-area networks are interconnected.

Options

The options for the **timed** command are:

-n *network*	Considers the named network.
-i *network*	Ignores the named network.
-t	Places tracing information in */usr/adm/timed.log*.
-M	Allows this time daemon to become a master. A time daemon run without this option is forced into the state of slave during an election.

Daily Operation

The **timedc(ADMN)** command is used to control the operation of the time daemon. It can be used to do the following:

- measure the differences between machines' clocks
- find the location where the master **timed** is running
- cause election timers on several machines to expire at the same time
- enable or disable tracing of messages received by **timed**

See the manual pages on **timed**(ADMN) and **timedc**(ADMN) for more detailed information.

The **rdate(ADMN)** command can be used to set the network date.

Chapter 5
Configuring NFS

SCO ODT-NET Network File System (NFS) is a software product that allows you to mount directories across the network and then treat remote files as if they were local. NFS was developed by Sun Microsystems as a standard for the exchange of data between different machines and operating systems. With the assistance of Lachman Associates, SCO implemented NFS on the UNIX operating system.

As a system administrator, you are responsible for configuring NFS, solving network problems, adding new machines, and adding new users. To assist you in these tasks, the UNIX system provides a convenient set of maintenance commands. Along with the UNIX system utilities, some additional utilities are provided for NFS administration.

Role of the Operating System in NFS

Unlike many recently marketed and distributed operating systems, the UNIX system was originally designed without knowledge that networks existed. This "networking ignorance" presents several impediments to linking the UNIX system with currently available high-performance networks:

- The UNIX system was never designed to yield to a higher authority (like a network-authentication server) for critical information or services. As a result, some operating system semantics are hard to maintain over the net. For example, it may not always be appropriate to trust user ID 0 (*root*).

- The execution semantics in the UNIX system is difficult. For example, these operating systems allow a user to remove an open file, yet the file does not disappear until closed by everyone. In a network environment, a client UNIX machine may not own an open file. Therefore, a server may remove a client's open file.

- When a machine running the UNIX system halts operation, it takes all its applications down with it. When a network node halts for any reason (whether client or server), it should not bring down all of its bound neighbors. The treatment of node failure on a network raises difficulties in any system and is especially difficult in the UNIX environment. A system of *stateless* protocols were implemented to circumvent the problem of a halting server dragging down its bound clients. In this context, *stateless* means that a client is independently responsible for completing work, and that a server need not remember anything from one call to the next. In other words, the server keeps no state. With no state left on the server, there is no state to recover when the server halts and comes back up. From the client's point of view, a halted server appears no different from a very slow server.

In implementing the UNIX system over the network, System V NFS remains compatible whenever possible. However, the following decisions have introduced incompatibilities:

- A preference for making a networked UNIX system a collection of network services rather than having it evolve into a distributed operating system
- A preference for making system halt recovery easier, from both the implementation and the administration points of view

Introducing NFS

In NFS, a server exports directories to be shared and clients mount the directories to access the files in them. The following subsections describe the general interaction of files and daemons in these activities. For details of what happens when a client mounts a remote filesystem, see the section "Remote Mount Failed" later in this chapter.

How Files Interact

To export a directory, the server must list it along with access rights in its */etc/exports* file. In addition, the server must have a name and address for each client to receive services in its */etc/hosts* file.

To mount a directory, the client must have its */etc/default/filesys* file contain all the remote filesystems and locations through which it can access directories. Any **mount**(NADM) command automatically adds an entry to the client's */etc/mnttab*(F) file, and any **umount** command automatically deletes its entry in this file, so this file shows the currently mounted directories.

How Daemons Interact

Two remote programs implement the NFS service — **mountd**(NADM) and **nfsd**(NADM). A client's **mount** request invokes **mountd**, which checks the server's */etc/exports* file for access permission of the client and returns a pointer to a filesystem. After **mount** completes, access to that mount point directory and below goes through the **nfsd** daemon on the server system. Client kernel file access requests (delayed-write and read-ahead) are handled by the **biod** (see **nfsd**(NADM)) daemons on the client.

Setting Up an NFS Client

Setting up an NFS client involves checking the */etc/default/filesys* file and mounting remote filesystems. All examples in this section use the client named *earth*.

Checking the /etc/default/filesys File

A client uses its */etc/default/filesys* file to keep track of specific remote filesystems and directory locations, called *mount points*, through which the client can access directories. You need to make sure that all filesystem information, including access rights, appears in this file. If any information is not present, edit the file to add it. You also need to use **mkdir** to create any mount points that do not already exist in the filesystem.

Here is a sample file:

```
bdev=jupiter:/usr/man mountdir=/usr/man \
fsck=no rcfsck=no rcmount=yes \
mount=no fstyp=NFS nfsopts="soft,rsize=1024,wsize=1024"
```

This entry defines the following:

- The remote resource to mount is jupiter:/usr/man (bdev= entry)
- This resource should be mounted on /usr/man (mountdir= entry)
- It should not be checked by **fsck**(ADM) (fsck= entry)
- It should not be checked by **fsck** when being mounted during normal startup procedures (rcfsck= entry)

- It should be mounted automatically when this system is brought into multiuser mode (rcmount= entry)
- Users should not be able to mount this filesystem on demand (mount=no entry)
- This is an NFS filesystem (fstyp= entry), and that there are some NFS specific option to be given to mount(NADM).

See **filesys**(F) for a description of the syntax and contents of */etc/default/filesys*.

Mounting the Remote Filesystem

The following text replaces the section of the same name.

Any exported filesystem can be remote mounted onto a machine, as long as its server can be reached over the network and the machine is included in the */etc/exports* list for that filesystem. On the machine where the filesystem is to be mounted, the superuser executes the **mount** command. For example, to mount the manual pages from remote machine *jupiter* on the directory */usr/jupiter.man*, enter:

```
mount -r -f NFS,soft jupiter:/usr/man /usr/jupiter.man
```

This example illustrates the use of the **soft** option to specify a soft mount, which means that the client will not retry the mount operation if the server does not respond to the first request. The default is a hard mount, which means that the client continues to try to mount the requested directory even if the server does not respond. A soft mount is recommended for read-only directories, such as online manual pages. The underlying reasoning is that with a soft mount, halting the server does not halt the client.

To make sure the filesystem is mounted where it is expected to be, use the **mount** command without any arguments. This displays the currently mounted filesystems.

Starting and Stopping NFS

NFS must be started by a shell script invoked by **init**(M) when the system is brought up to multi-user mode (run level 2). This is normally configured automatically during installation by linking the script */etc/nfs* to */etc/rc2.d/S89nfs* You need to start NFS on at least one server and at least one of each server's clients to have a functional network.

To stop NFS on a node, log in as *root* and enter the following command:

```
# /etc/nfs stop
```

When you stop NFS on a node, its resources cannot be accessed from any other node. NFS is shut down automatically when the system is taken to run levels 0, 1, or 6.

Debugging NFS

Most problems involving System V NFS network services lie in the one of the following four areas, which are listed in order of probability:

1. The network-access control policies do not allow the operation, or architectural constraints prevent it.
2. The client software or environment is not functioning correctly.
3. The server software or environment is not functioning correctly.
4. The network is not functioning correctly.

To debug NFS, you should be familiar with how NFS works and the names and functions of the various daemons and database files:

biod (see **nfsd**(NADM))	NFS daemon
mount(NADM)	mounts a filesystem
mountd(NADM)	NFS mount requested server
nfsd(NADM)	NFS daemon
rpcinfo(NADM)	reports RPC information
showmount(NADM)	shows all remote mounts
filesys(F)	format of filesystem mount commands table
mnttab(NF)	format of mounted filesystem table
exports(NF)	NFS filesystem being exported

For example, consider a sample mount request as made from an NFS client machine:

```
# mount -f NFS krypton:/usr/src /krypton.src
```

The example asks the server machine *krypton* to return a file handle (**fhandle**) for the directory */usr/src*. This **fhandle** is then passed to the kernel in the **mount**(NS) system call. The kernel looks up the directory */krypton.src* and, if everything is correct, ties the **fhandle** to the directory in a mount record. From now on, all filesystem requests to that directory and below go through the **fhandle** to the server *krypton*.

This describes the way the system should work. We now look at the things that can go wrong: first, some general pointers; then, a list of the possible errors and what might have caused them.

General Hints

When there are network or server problems, programs that access hard-mounted remote files fail in different ways from those that access soft-mounted remote files. Hard-mounted remote filesystems cause programs to retry until the server responds again. Soft-mounted remote filesystems return errors after trying for a while.

Once a hard mount succeeds, programs that access hard-mounted files will hang as long as the server fails to respond. In this case, NFS displays the following message on the console:

```
server not responding
```

On a soft-mounted filesystem, a program receives an error message when it tries to access a file whose server is dead.

If a client is having NFS trouble, first check that the server is up and running. From a client, type the following command to see if the server is up at all:

```
# rpcinfo -p server_name
```

It should display a list of program, version, protocol, and port numbers similar to the following:

```
program vers proto   port
 100000    2   tcp    111  portmapper
 100000    2   udp    111  portmapper
 100017    1   tcp   1024  rexd
 100005    1   udp   1027  mountd
 100003    2   udp   2049  nfs
 100024    1   udp   1039  status
 100024    1   tcp   1025  status
 100021    1   tcp   1026  nlockmgr
 100021    1   udp   1051  nlockmgr
 100020    1   udp   1054  llockmgr
 100020    1   tcp   1028  llockmgr
 100021    2   tcp   1032  nlockmgr
```

You can also use **rpcinfo** to check if the **mountd** server is running:

 # rpcinfo -u *server_name* **mountd**

This should return:

```
program 100005 version 1 ready and waiting
```

If these steps fail, try to log in on the server's console to see if it is running.

If the server is alive but a client machine cannot reach it, check the Ethernet connections between the machines.

If the server and the network are alive, use **ps** to check the client daemons. A **nfsclnt** and several **biod** daemons should be running. For example, the command:

 # ps -ef

should print lines for **nfsclnt** and **biod**.

The following sections deal with the most common types of failure. The section "Remote Mount Failed" covers the steps to take if a remote mount fails. The sections "Programs Are Hung" and "Everything Works Slowly" discuss servers that do not respond when filesystems are mounted.

Remote Mount Failed

This section deals with problems related to mounting remote filesystems. If **mount** fails for any reason, check the sections below for specific details about what to do. They are arranged according to where they occur in the mounting sequence and labeled with the error message likely to be displayed.

The **mount** command can get its parameters either from the command line or from the file */etc/default/filesys*. (See **mount**(NADM).) The example below assumes command-line arguments, but the same debugging techniques would apply if */etc/default/filesys* were the source of the options.

The interaction of the various parts of the mount request should be considered. In the example mount request given above:

```
mount -f NFS krypton:/usr/src /krypton.src
```

mount goes through the following steps to mount a remote filesystem.

1. The **mount** command opens */etc/mnttab* and checks that this mount was not already done.
2. The **mount** command parses the first argument into host *krypton* and remote directory */usr/src*.
3. The **mount** command then resolves the host *krypton* into an Internet protocol address.
4. The **mount** command calls *krypton*'s portmapper to get the port number of **mountd**.
5. The **mount** command calls *krypton*'s **mountd** and passes it to the */usr/src* pathname.
6. *krypton*'s **mountd** reads its */etc/exports* and looks for the exported filesystem that contains */usr/src*.
7. *krypton*'s **mountd** does a **nfs_getfh**(NS) system call on */usr/src* to get the **fhandle**.
8. *krypton*'s **mountd** returns the **fhandle** to the client.
9. On the client machine, **mount** does a **mount**(NS) system call with the **fhandle** and */krypton.src*.

10. The **mount** command checks whether or not the caller is a superuser and */krypton.src* is a directory.

11. The **mount** command does a **statfs**(S) call to *krypton*'s NFS server (**nfsd**).

12. The **mount** command opens */etc/mnttab* and adds an entry.

Any one of these steps can fail, and some of them can fail in more than one way. The following sections give detailed descriptions of the failures associated with specific error messages.

`mount: can't open /etc/mnttab`

The table of mounted filesystems is kept in the file */etc/mnttab*. This file must exist before **mount** can succeed. The file */etc/mnttab* is created when the system is booted, and it is maintained automatically after that by the **mount** and **umount** commands.

`mount: /dev/nfsd is already mounted,` *sys_name* `is busy, or allowable number of mount points exceeded`

This message reveals an attempt to mount a filesystem that is already mounted. All NFS mount requests that fail with this message display the name */dev/nfsd* (a byproduct of the implementation) regardless of the actual mount request.

`mount:` *name* `or` *name*`, no such file or directory`

The **-f NFS** or **krypton:** part of the sample command was probably omitted. The **mount** command assumes that a local mount is being done unless the **-f** flag is used on the command line, or the requested directory as listed in */etc/default/filesys* specifies filesystem type **NFS**.

More simply, this message also appears when the specified local mount point is not an existing directory.

`mount: can't open /etc/default/filesys`

The **mount** command tried to look up the information needed to complete a mount request in */etc/default/filesys*, but there was no such file. As the system administrator, you need to create this file as part of initial system setup.

sys_name `not in hosts database`

The system name specified on the **mount** command suffixed by the ":" is not listed in the file */etc/hosts*. Check the spelling of the host name and the placement of the colon in the **mount** command.

`mount: directory argument` *name* `must be a full pathname`

The second argument to **mount** is the path of the directory to be covered. This must be an absolute path starting at slash (/).

`mount: ... server not responding(1): RPC_PMAP_FAILURE - RPC_TIMED_OUT`

Either the server to which the mount is being attempted is down, or its portmapper is dead or hung. Attempt to log in to that machine; if that attempt succeeds, then the problem may be in the portmapper. Run the following command from your system as the superuser to test the portmapper on the server system:

 # rpcinfo -p *hostname*

The result should be a list of registered programs. If this is not the case, kill the remote portmapper and restart it. Restarting the portmapper is a complicated process because all registered services are lost, and their associated daemons must be restarted, too. To find the process IDs of **portmap** and other service daemons, enter this command as the superuser:

 # ps -ef

Kill the daemons with:

 # kill -9 *portmap_pid daemon_id1 daemon_id2*

Start new ones with commands such as:

```
# /etc/portmap
# /etc/mountd
# /etc/nfsd
```

Another alternative to all this is simply to reboot the server when it is convenient. Because of the stateless nature of the NFS server implementation, there should be no adverse effect on the clients of the system, other than the time that they suspend awaiting the return of the server.

If the server is up but it is not possible to **rlogin** to it, check the client's Ethernet connection by trying to **rlogin** to another machine, or to ping another machine. Also, check the server's Ethernet connection.

`mount: ... server not responding: RPC_PROG_NOT_REGISTERED`

This means that **mount** got through to the portmapper, but the NFS mount daemon **mountd** was not registered. The server should be checked to ensure that */etc/mountd* exists and is running.

`mount: /dev/nfsd or` *name*`, no such file or directory`

Either the remote directory does not exist on the server or the local directory does not exist. Again, note that */dev/nfsd* is always printed to represent the remote directory.

`mount: access denied for` *sys_name*`:`*name*

The client machine on which the mount attempt is being made is not in the server's export list for the filesystem to be mounted. A list of the server's exported filesystems can be obtained by running:

```
# showmount -e hostname
```

If the desired filesystem is not in the list, or the machine name is not in the user list for the filesystem, then check the */etc/exports* file on the server for the correct filesystem entry. A filesystem name that appears in the */etc/exports* file but not in the output from **showmount** indicates a failure in **mountd**. Perhaps it could not read that line in the file, or it could not find the filesystem, or else the filesystem name was not a locally mounted filesystem. See *exports*(NF) for more information.

This message can also be an indication that authentication failed on the server. It may be displayed because the machine that is attempting the mount is not in the server's export list, because the server is not aware of the machine, or because the server does not believe the identity of the machine. Check the server's */etc/exports* file.

`mount:` *name*`:  no such file or directory`

The remote path on the server is not a directory.

`mount: not superuser`

The **mount** command can be used only by the superuser, because it affects the filesystem for the entire machine.

Programs Are Hung

If programs hang doing file-related work, the NFS server may be dead. The following message may be displayed on the machine's console:

`NFS server` *sys_name* `not responding, still trying`

The message includes the name of the NFS server that is down.

This is probably a problem either with one of the NFS servers or with the Ethernet.

If a client machine completely stops responding, check the servers from which the filesystems were mounted. If one of them is down, client machines may wait indefinitely. When the server comes back up, programs continue automatically and are unaffected.

If a soft-mounted server dies, other work should not be affected. Programs that time-out trying to access soft-mounted remote files will fail, but it should still be possible to get work done on other filesystems.

If other clients of the server seem to be functioning correctly, the Ethernet connection and the connection of the server should be checked.

Everything Works Slowly

If access to remote files seems unusually slow, the server should be checked by entering (on the server):

```
# ps -ef
```

If the server is functioning and other users are getting good response, block I/O daemons on the client should be checked with the command **ps -ef** (on the client) and looking for **biod**. If the daemons are not running or are hung, they should be killed. First, find the process IDs by entering:

```
# ps -ef | grep biod
```

Then, kill them with:

```
# kill -9 pid1 pid2 pid3 pid4
```

Restart the daemons with:

```
# /etc/biod 4
```

To determine whether the daemons are hung, use **ps** as above, then copy a large file. Another **ps** will show whether the **biod**s are accumulating CPU time; if not, they are probably hung.

If **biod** appears to be functioning correctly, check the Ethernet connection. Use the **nfsstat -c** and **nfsstat -s** commands to discover whether a client is doing a lot of retransmitting. A retransmission rate of 5% is considered high. Excessive retransmission usually indicates a bad Ethernet board, a bad Ethernet tap, a mismatch between board and tap, or a mismatch between the client machine's Ethernet board and the server's board.

Serial Number Validation Errors

When you see a message of the following nature:

```
Copy Protection Violation!
Duplicate Serial Number(s) detected on 123.4.567.007
Product(s) att00001 shutting down.
```

a validation error has occurred. This means that there are two copies of the same program running at the same time. You must remove one of the copies.

Adding a New User

Adding a new user involves adding an entry to the proper password files and, for local logins, creating a home directory on the new user's machine. For general information on how to do this, see *Administering ODT-OS* in the *Administrator's Guide*.

Within a network environment, to add a new user, you should add a password file entry to every machine on the local network. Each user and group must have a unique identification number across the entire network, rather than just on the home machine. A server machine must have an entry for each user on each client machine who will be using NFS services.

The user ID should be a number unique to the user. A system knows the user by the ID number associated with the login name; therefore, a login name must have the same user ID number in all password files of machines that are networked in a local domain. Failure to keep IDs unique will prevent users from moving files between directories on different machines, because the system will respond as if the directories were owned by two different users. In addition, file ownership may become confused when an NFS server exports a directory to an NFS client whose password file contains users with UIDs matching those of different users on the NFS server.

Incompatibilities with Remote Filesystems

A few things work in different ways, or not at all, on remote NFS filesystems. The next section discusses the incompatibilities and offers suggestions for working around them.

No SU over the Network

Under NFS, a server exports filesystems it owns, so that clients may remotely mount them. When a user on a client machine becomes the superuser, it is denied superuser permission on remote-mounted filesystems. Consider the following actions performed by the user *jsbach* on the server machine:

```
$ cd
$ touch test1 test2
$ chmod 777 test1
$ chmod 700 test2
$ ls -l test*
-rwxrwxrwx  1 jsbach       0 Mar 21 16:12 test1
-rwx------  1 jsbach       0 Mar 21 16:12 test2
```

The superuser on the client machine tries to access the remote filesystem on the server machine, intending to use the superuser privileges:

```
$ su
Password:
# cd /usr2/jsbach
# touch test1
# touch test2
touch: test2: Permission denied
# ls -l test*
-rwxrwxrwx  1 jsbach       0 Mar 21 16:16 test1
-rwx------  1 jsbach       0 Mar 21 16:12 test2
```

When checking permissions for accessing a remote-mounted filesystem, the server considers the superuser from a client machine to be in the "other" category.

The problem usually shows up during the execution of a setuid *root* program. Programs that run as *root* cannot access files or directories unless the permission for "other" allows it. When operating from a client machine, the uid for root is mapped to -2 (or 65534).

Another aspect of this problem is that ownership of remote-mounted files sometimes cannot be changed, specifically if they are on a server that does not permit users to execute **chown**. Because *root* is treated as the "other" user for remote accesses, only *root* on the server can change the ownership of remote files. For example, consider a user trying to **chown** a new program *a.out* that must be setuid to *root*. It does not work, as shown below:

```
$ chmod 4777 a.out
$ su
Password:
# chown root a.out
a.out: Not owner
```

To change the ownership, you must either log in to the server as *root* and make the change, or move the file to a filesystem owned by the user's machine (for example, */usr/tmp*, which is usually owned by the local machine) and make the change there.

File Operations Not Supported

File locking of directories is not supported on remote filesystems.

In addition, Append mode and atomic writes are not guaranteed to work on remote files accessed by more than one client simultaneously.

Cannot Access Remote Devices

With NFS, it is not possible to access a remote-mounted device or any other character, block-special file, or named pipes.

Cannot Access Different Filesystem Types

Under NFS, it is not possible to access a filesystem of a type different from that of your native system. For example, you are running NFS under UNIX. It is not possible to access a DOS filesystem from your UNIX client machine.

Cannot Access Indirect Filesystems

Under NFS, it is not possible to access a filesystem indirectly. For example, you may not use a server machine to access a filesystem on a third server machine. All NFS access must be direct from server to client.

Handling Clock Skew in User Programs

Because the NFS architecture differs in some minor ways from earlier versions of the XENIX and UNIX systems, users should be aware of those places where their own programs could run up against these incompatibilities.

The clocks on the NFS server and client may be out of sync because each machine keeps its own time. This might cause problems. Consider the following example.

Many programs assume that an existing file could not be created in the future. For example, the command **ls -l** has two basic forms of output, depending upon how old the file is:

```
$ date
Sat Apr 12 15:27:48 GMT 1986
$ touch file2
$ ls -l file*
-rw-r--r--  1 jsbach          0 Dec 27  1984 file
-rw-r--r--  1 jsbach          0 Apr 12 15:27 file2
```

The first type of output from **ls** prints the year, month, and day of last file modification if the file is more than six months old. The second form prints the month, day, hour, and minute of last file modification if the file is less than six months old.

The **ls** command calculates the age of a file simply by subtracting the modification time of the file from the current time. If the result is greater than six months, the file is "old".

Assume that the time on the server is Apr 12 15:30:31, which is three minutes ahead of the local machine's time:

```
$ date
Apr 12 15:27:31 GMT 1986
$ touch file3
$ ls -l file*
-rw-r--r--  1 jsbach          0 Dec 27  1983 file
-rw-r--r--  1 jsbach          0 Apr 12 15:26 file2
-rw-r--r--  1 jsbach          0 Apr 12  1986 file3
```

The difference between the current time and the library's modify time makes the **ls** command think that the new file was created long ago.

In general, users should remember that applications that depend upon local time and/or the filesystem timestamps will have to deal with clock skew problems if remote files are used.

Chapter 6

Managing the LAN Manager Client Network

Installation of the ODT-NET LAN Manager Client network consists of establishing unique user and group IDs throughout the network and configuring the computer as a consumer. Chapter 1 discusses the importance of establishing consistency of network ID files. The rest of this section describes how to use the **mkself**(XC) command to configure a computer and how to add a new computer to the network. When you install the LAN Manager Client software, the **custom** program performs these operations automatically. The information is provided here as a tool for system maintenance; it is not necessary to duplicate the actions performed by the **custom** program.

The remaining sections of the chapter discuss:

- the special network files used by LAN Manager Client
- starting and stopping the LAN Manager Client network
- NetBIOS
- network parameters
- configuring for performance

This chapter focuses on administering a LAN Manager Client network to connect to a DOS or OS/2 machine on a LAN Manager network. For information about providing access to a XENIX-NET server, see the documentation provided with the XENIX-NET server.

Defining Network Computers: The mkself Utility

The **mkself** command builds the necessary network files on a network computer. The **custom** program performs this operation for all network computers.

From the root account, enter the command:

```
# mkself name
```

where *name* is the network name of the computer. (See Chapter 1 of this guide for computer naming restrictions.) The **mkself** utility edits the */etc/systemid* file to include the machine name.

Adding Network Computers

Once the network is installed, it may be necessary to alter the network's setup to meet changing computing needs and to remedy any problems that arise due to changes in your system.

When adding a new computer to the network, the procedure is essentially the same as for initial installation. Most tasks are performed by the **custom** utility during the installation. If you choose to have the network start automatically, **custom** places the **net start rdr** command in the file */etc/rc.d/6/xnet.6*. This command automatically starts your computer as a network consumer each time you enter multiuser mode. Remember that you must be logged in as the super user to add computers.

1. Choose a name and password for the new machine. The name must not duplicate any name already on the network or any top-level file or directory name. See "Starting and Stopping the Network" later in this chapter for the startup commands.

2. **custom** configures each computer by using the **mkself** command to add the machine name to the file */etc/systemid*.

All consumers must be added as users to the server machine they access.

Special Network Files

The network requires special files to operate. These files allow network commands and requests to locate users, groups, computers, and resources. This section describes each of the special networking files. This section does not explain any files already present in the UNIX operating system that are used by the network. Also, the actual LAN Manager Client program files are not listed as part of this section. The network parameters mentioned here are described in detail under "Network Parameter Descriptions" later in this chapter.

Consumer Configuration File (/usr/lib/xnet/constable)

The *constable* file is used by UNIX system consumers to specify five parameters that affect the performance of data transfer between the consumer and a server. The parameters can be individually specified for different servers. This file does not need to be changed for the normal operation of LAN Manager, although modifications to it can improve network performance.

Network Parameter File (/usr/lib/xnet/xnetrc)

Network parameters can be altered via the file */usr/lib/xnet/xnetrc*. After modifications have been made to this file, the network must be restarted for the changes to take effect.

Starting and Stopping the Network

This section explains how to start the LAN Manager Client network both automatically and by entering commands. It also explains how to stop the network. Remember that you must be logged in as the super user to give these commands.

Starting the LAN Manager Client Network

To start your computer as a consumer, use one of the following commands:

net start rdr	Starts your computer as a consumer.
xfc on	Starts your computer as a consumer.

If, during the installation procedure, you chose to activate LAN Manager Client automatically upon going into multiuser mode, the installation script copies the file *xnet.6* from */usr/lib/xnet/rc.d* to the */etc/rc.d/6* directory. This automatically starts your computer as a

network consumer each time you start your computer in multiuser mode. Once this file is copied into */etc/rc.d/6/xnet.6*, manual entering of the **xfc** or **net start** command is not needed.

If, during the installation procedure, you chose not to have LAN Manager Client start automatically, the file *xnet.6* is not copied to the */etc/rc.d/6* directory. This file is stored in */usr/lib/xnet/rc.d*. If you decide at this time to have LAN Manager Client start automatically, copy this file to */etc/rc.d/6*.

Stopping the LAN Manager Network

If a serious problem occurs with the network software or hardware, it may be necessary to halt the network processes. Normally, users simply log off until the problem is fixed. However, if users can continue their work without using the network, the following commands stop network functions but leave a computer available for local work:

net stop rdr	Stops your computer as a consumer. Existing connections to servers are unaffected. No new network connections are allowed.
xfc off	Shuts down the consumer. Existing connections to servers are unaffected. No new network connections are allowed.
xfc abort	Immediately stops the consumer. All connections to servers are closed. No new network connections are allowed.

NetBIOS

LAN Manager Client operates as a NetBIOS consumer. The maximum and default number of NetBIOS sessions available can vary with the networking hardware being used.

LAN Manager Client filesharing creates and maintains a list of sessions during the course of operation. These sessions can be in the active state, which means that one server process exists on the server side of the session for each client process on the consumer side of the session. If no server-consumer relationship currently exists, this means that either there are no open remote files or that there are no processes with remote current directories. If this is the case, the session is said to be dormant.

LAN Manager Client uses its own limited virtual circuit table. It does not try to use more than the number of virtual circuits specified in the NVCSUSED parameter. If LAN Manager Client needs to communicate with a machine with which it does not currently have a session, it creates a new session. If this causes LAN Manager Client to exceed the size of its virtual circuit table, dormant circuits are recycled. If no circuits are dormant, LAN Manager Client returns an error message.

If you will be using the Open Desktop Development System to build int5c applications, there are some special considerations you should be aware of. These applications use NetBIOS sessions. The NetBIOS sessions used by these applications are in addition to those used by the LAN Manager Client filesharing. Thus, if you wish to use both the LAN Manager Client filesharing and the applications that share the NetBIOS with LAN Manager Client, you must limit the number of NetBIOS sessions that the filesharing attempts to use. This is done by specifying the value of the NVCSUSED parameter in the */usr/lib/xnet/xnetrc* file, in conjunction with the MAXVCS parameter in the */etc/conf/cf.d/mtune* file.

The value of the NVCSUSED parameter must be set to equal the total number of NetBIOS sessions available, less the number of simultaneous sessions to be used by other applications and utilities. Use the **xnstatus** command to determine the number of NetBIOS sessions available to be configured. See the **xnstatus**(XC) manual page for more information about the **xnstatus** command. The number of NetBIOS sessions available can be adjusted to be between one and the maximum number available using **xnreset**.

Network Parameter Descriptions

The network filesharing parameters located in the file */etc/conf/cf.d/mtune* determine the amount of memory allocated to data structures in the kernel. These parameters have an effect only at kernel link time. In contrast, the parameters located in */usr/lib/xnet/xnetrc* determine how LAN Manager Client distributes the network resources among the network machines. The *xnetrc* parameters affect the runtime operation of LAN Manager Client and do not affect the size of the kernel. For example, the NPTE parameter in *mtune* determines the size of the (system-wide) network process table that LAN Manager Client uses. The NPTE parameter keeps track of the processes using network resources. The MSPVC limits the number of processes that can use a particular virtual circuit, even though more spaces may be available in the network process table.

Changing Network Parameters

To change a network parameter located in */etc/conf/cf.d/mtune*, add the parameter to the file */etc/conf/cf.d/stune* and assign it the desired value. When you have finished modifying *stune*, type:

link_unix

to relink the kernel. Network parameter values specified in */etc/conf/cf.d/stune* override those specified in */etc/conf/cf.d/mtune*.

NOTE: It is not necessary to change any of the parameters under normal conditions. Changing any parameter may adversely affect LAN Manager Client's performance.

The following table shows the parameters named in */etc/conf/cf.d/mtune* that you can include and modify in */etc/conf/cf.d/stune*. The default values listed are for machines using the Intel® 80386™ processor.

Descriptions of the parameters follow the tables.

Parameter	Default Value
NNCB_DEV	32
NNCBS	32
NNCB_NAMES	16
NFSTREAM	32
NFSBUFS	60
NBINDX	80
MAXVCS	20
NPTE	64
NBIDS	32
NTIDS	32
NEXS	150
NWB	32
NASEVENT	40
NCONSTABLE	32
NALIAS	16
NRECYCLE	3
NCALLRETRY	10
XITONCLOSE	0
CNCBS	0
CORMAPNCB	0

Streams Parameter	Default Value
NQUEUE	256
NSTREVENT	64
N4K	0
N2K	10
N1K	20
N512	10
N256	10
N128	10
N64	30
N16	60
N4	128

NOTE: These defaults may be different on your system.

CNCBS

Number of co-resident NCB structures allocated.

Default=0

This parameter is set to 0 when only one NetBIOS is installed.

CORMAPNCB

This is a structure used for mapping NCBS to sessions.

Default=0

This parameter is set to 0 when only one NetBIOS is installed.

MAXVCS

Maximum number of virtual circuit table entries.

Default=20

This structure is used by the network filesharing software to keep track of the state of the virtual circuits. The value of this parameter should not exceed the number of virtual circuits supported by the lower layers of the network.

NALIAS

Number of alias table entries.

Default=16

The value for NALIAS assigned in the file */etc/conf/cf.d/mtune* specifies the number of 12-word structures (nc alias) reserved in the kernel data segment. Each time a UNIX system consumer uses the **net use** command, an entry in this table is used. This table keeps track of the mapping of the alias name used to the appropriate virtual circuit. If all the table space is used, the **net use** command returns an error message to the user.

NASEVENT

Number of async event cells allocated in the kernel.

Default=40

One event cell is used for each outstanding read-ahead or write-behind request. If this parameter is set too low, this message is displayed:

```
LAN Manager Client: low on async cells (NASEVENT)
```

To fix this problem, either raise NASEVENT or decrease the read-ahead and write-behind window sizes. For more information on the window sizes, see "Configuring for Performance."

NBIDS

Number of bind table entries.

Default=30

The value for NBIDS assigned in the file */etc/conf/cf.d/mtune* specifies the number of 11-word structures (bidtab) reserved in the kernel data segment. The error message "Server: out of bind entries (NBIDS)" indicates that NBIDS needs to be increased on your network.

NBINDX

Number of BINDX structures.

Default=80

Used for tracking read-ahead requests. This message indicates that NBINDX should be raised or the read-ahead windows should be reduced:

```
LAN Manager Client: low on bindxs (NBINDX)
```

For more information on read-ahead windows, see "Configuring for Performance" later in this chapter.

NCALLRETRY

Number of attempts to establish a session.

Default=10

This is the number of times LAN Manager Client attempts to establish a session if the initial attempt fails. This retry only occurs when certain conditions on the server exist, and it is likely that a retry succeeds.

NCONSTABLE

Number of consumer table structure entries.

Default=32

This parameter corresponds to the number of distinct servers specified in the file */usr/lib/xnet/constable*. If you add more servers, increase this value accordingly.

NEXS

Number of exclude table entries.

Default=150

The value for NEXS assigned in the file */etc/conf/cf.d/mtune* specifies the number of four-word structures (extab) reserved in the kernel data segment. The error message "Server: out of exclude fid entries (NEXS)" indicates NEXS may need to be increased on your network.

NFSBUFS

Maximum number of network buffers.

Default=60

This parameter determines the total number of network buffers available. Reducing this number increases the amount of memory available for user programs.

NFSTREAM

Number of streams to the adapter.

Default=32

One stream is used for each filesharing virtual circuit.

NNCBS

Maximum number of NCBs.

Default=32

NNCB_DEV

Number of sessions the network driver can handle.

Default=32

NNCB_NAMES

Maximum number of names that the network device allows.

Default=16

NPTE

Number of process environment table entries.

Default=64

The value for NPTE assigned in the file */etc/conf/cf.d/mtune* specifies the number of 11-word structures (nptab) reserved in the kernel data segment. These structures are used by the network software to keep track of processes using network resources. An entry in this table is used when a local process first references a remote file. The entry is released when the process exits.

The error message "out of network process table space (npte)" indicates that the value of NPTE is too low for your network activity.

NQUEUE

Number of stream queues.

Default=256

The number of stream queues must be at least four times the value of NFSTREAM, plus all queues used by other software in the kernel.

If you change the value of the NQUEUE parameter, errors can occur.

NRECYCLE

Maximum number of dormant circuits LAN Manager Client recycles.

Default=4

This is the maximum number of dormant circuits that LAN Manager Client recycles when its virtual circuit table is full, and it needs to consume from a new server.

NSTREVENT

Number of stream event cells.

Default=64

NTIDS

Number of tree connect table entries.

Default=32

The value for NTIDS assigned in the file */etc/conf/cf.d/mtune* specifies the number of five-word structures (tidtab) reserved in the kernel data segment. Tree connects bind a consumer alias with a server. The error message "Server: out of tree connect entries (NTIDS)" indicates that the NTIDS value may need to be increased.

NWB

Number of write-behind control structures.

Default=32

One NWB is allocated for each file being written to. This message indicates that more NWBs should be allocated:

```
LAN Manager Client: out of write behind table entries (NWB)
```

XITONCLOSE

Allows virtual circuits to become dormant when all files close.

Default=0

This is a Boolean variable. If XITONCLOSE is set to non-zero, filesharing virtual circuits become dormant when all files are closed. If it is zero, circuits remain active until all processes that use remote resources on the circuit have exited. The advantage to leaving the circuit active is that subsequent opens have less overhead associated with them on the server machine. Applications that frequently open and close many files benefit from this. The advantage to allowing circuits to become dormant is that the server then has the option of recycling the circuit to serve another machine if it becomes necessary.

Number of Stream Data Blocks

N4

Number of stream data blocks of size 4.

Default=60

N16

Number of stream data blocks of size 16.

Default=60

N64

Number of stream data blocks of size 64.

Default=30

N128

Number of stream data blocks of size 128.

Default=10

N256

Number of stream data blocks of size 256.

Default=10

N512

Number of stream data blocks of size 512.

Default=20

N1K

Number of stream data blocks of size 1024.

Default=20

N2K

Number of stream data blocks of size 2048.

Default=12

N4K

Number of stream data blocks of size 4096.

Default=0

Changing Network Filesharing Parameters Located in xnetrc

The network filesharing parameters located in the */usr/lib/xnet/xnetrc* file determine various functional limits in the LAN Manager Client software. The network server reads the *xnetrc* file when filesharing is started with the **net start** command. The filesharing program checks for entries that override default filesharing parameters.

The network system administrator can change some network parameters by modifying the */usr/lib/xnet/xnetrc* file. It is unnecessary to change any of the parameters under normal conditions. Changing any parameters can adversely affect LAN Manager Client performance.

The following table shows the parameters and default values as named in the */usr/lib/xnet/xnetrc* file. Parameter descriptions follow the table.

Parameter	Default Value
NVCSUSED	20
MBPVC	15
MCONVCS	20
MFPVC	500
MSERVCS	20
MTPVC	20
NBIOSIZ	0 (auto-configured)
NETDEV	0
NBUFALLOC	0 (auto-configured)
NORDONLY	0

NVCSUSED

Maximum number of virtual circuits that are used by the network filesharing code.

Default=20

This structure is used by the network filesharing software to keep track of the state of the consumer virtual circuits. The value of this parameter should not exceed the number of virtual circuits supported by the lower layers of the network. It should also not exceed the number specified in MAXVCS in the *mtune* file. Applications written using the int5c library can use virtual circuits other than the ones specified in NVCSUSED.

MBPVC

Maximum number of binds per server virtual circuit.

Default=15

Non-core protocol virtual circuits allow consumers to "bind" to a directory or file. XENIX computers use this process when a user changes to a directory on a remote computer. This parameter limits the number of bind table entries that a single virtual circuit can use.

MCONVCS

Maximum number of virtual circuits for use by the consumer.

Default=20

The value of MCONVCS cannot exceed the value of NVCSUSED.

MFPVC

Maximum number of fids per server virtual circuit.

Default=500

This parameter limits the number of open files allowed at one time by the consumer using a virtual circuit.

MSERVCS

Maximum number of virtual circuits for use by the server.

Default=20

The value of MSERVCS cannot exceed the value of NVCSUSED.

MTPVC

Maximum number of tree connects per virtual circuits.

Default=20

This parameter limits the number of tree connect table entries (tids) that can be used by a single virtual circuit. Tree connects are used to "login" to a server when the server is validating users.

The maximum value equals the value of NTIDS in the *mtune* file.

NBIOSIZ

Size of network buffers.

Default=0

This parameter defines the size of the actual buffer space used to send and receive messages. This parameter is auto-configured if a value of 0 is specified.

For detailed information on configuring the buffer size, see "Configuring for Performance."

NETDEV

Major device number of the network device used for filesharing.

Default=0

NETDEV specifies which networking device filesharing requests are routed to. Normally, the device driver is provided by your network supplier.

NBUFALLOC

Number of network buffers.

Default=0

This parameter specifies the total number of network buffers allocated when filesharing is started. This parameter is auto-configured if a value of 0 is specified.

For detailed information on configuring buffers, see "Configuring for Performance."

NORDONLY

No read-only core protocol file simulation.

Default=0

The core protocol supports file opens that are exclusive to a consumer, but can be shared among processes within that consumer. However, it is often desirable to share such files among other network computers, as long as only read accesses are permitted. If NORDONLY is set for zero, such files will be available across the network for reading. If it is non-zero, no multiple-computer accesses will be allowed on open files.

Configuring for Performance

This section explains how to improve LAN Manager Client's performance by altering the configuration of your system. None of what is described here is required for the normal operation of LAN Manager Client.

Adjusting Consumer Read and Write Windows

By editing the file */usr/lib/xnet/constable* on your consumer machines, you can individually adjust the read and write windows (described below) for each server that the consumer is in contact with. You can greatly enhance network performance by carefully tuning these parameters.

The *constable* file has this format:

```
#               Special                  General                     General
#            Read Support              Read-Ahead                  Write-Behind
#
#--------+---------------+---------------------------+---------------+
# server         reado        reada    read     read          write
# name           on/off       timeout  window   ahead         window
#
*                1            500      3        3             4
```

Note that every line in the table except the last one is commented out. This line has an asterisk (*) in the "server name" column, and it lists the default values for each parameter. To specify different values for a specific server, add a line to the bottom of the file that has the server's name at the beginning of the line, and the values you want to use for that server in the other columns. Any machine that does not have a line of its own uses the defaults specified on the line beginning with the asterisk.

The sections below describe the parameters and suggest how to change them to improve performance.

reado on/off

This parameter is a toggle. A value of 1 turns on an improved protocol set, and a value of 0 turns it off. It should always be set to 1.

reada timeout

This sets the length of time that buffered data remains valid, in units of 1/50th of a second. For example, if this parameter is set to 500, data that was read from a server and placed in a buffer remains valid for 10 seconds. If the data is not read from the buffer within this period, the buffer is flushed and the data is re-read from the server.

Setting this parameter to a high number improves network performance, but it can cause data integrity problems if the data being read changes frequently. As a general guideline, use a high value for machines that contain relatively stable data (such as archive servers) and a lower value for machines with more volatile data.

read window

With this parameter, you can specify the number of read requests that may remain outstanding before a reply is received. A value of zero or one causes the read requests to be sent one at a time; after a request is sent, no other requests are sent until a reply has been received. A value of 2 allows two requests to be sent before a reply is received. Figure 6-1 illustrates this difference:

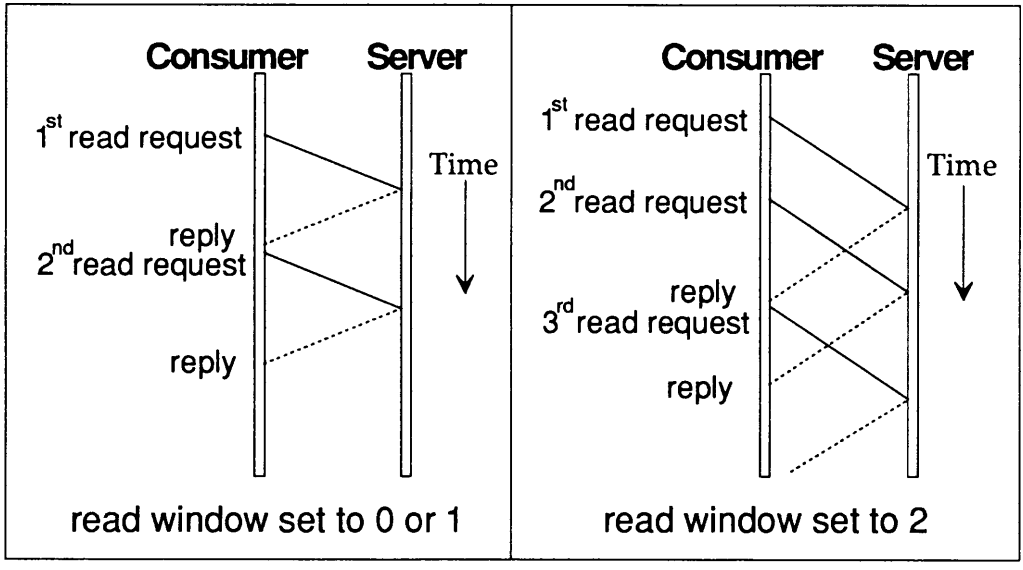


Figure 6-1. The read window size affects performance.

Setting this parameter to a high value dramatically improves network performance, but it is a drain on system resources. You may wish to test several different values to find what works best on your system.

read ahead

The read-ahead value is the number of extra read requests that LAN Manager Client sends after the initial request was satisfied. By setting this parameter to a value greater than 0, you allow LAN Manager Client to anticipate a user process by requesting data that the process has not yet asked for.

A high read-ahead value improves network performance, but it increases the drain on system resources. As with the read window, the best way to find the ideal value is through experimentation.

write window

The write window is very similar to the read window. This parameter specifies the number of write requests that may be outstanding before the server acknowledges that the data was received intact. If this is unclear, refer again to Figure 6-1. Change the word "read" in the figure to "write".

Adjusting Network Buffer Sizes

The network buffer size is the maximum amount of data that can be sent from or received by a machine in a single packet. You can improve network performance by properly configuring buffer sizes.

There are two important things to keep in mind when adjusting buffer sizes. First, large buffers use more memory. If you specify too large a buffer size, other processes may suffer for the lack of available memory. Second, when two machines with different buffer sizes are involved in a transaction, the smaller of the two buffer sizes is used. This is called the "effective buffer size."

The following sections provide some tips on configuring buffer sizes for maximum performance.

Checking the Buffer Size

To find the size of the network buffer that is being used for LAN Manager Client communications between two computers, enter the **net stat** command on one of the two computers. The *bufsz* column of the **net stat** output lists the effective buffer size for every machine to which the computer is connected.

Buffer Size Is Automatically Configured by Default

These two parameters in the file */usr/lib/xnet/xnetrc* specify the size and number of the network buffers:

Name	Configures
NBIOSIZ	buffer size
NBUFALLOC	number of buffers

If these parameters are set to zero, LAN Manager Client checks the amount of available memory and sets NBIOSIZ and NBUFALLOC to what it thinks are good values. To override the autoconfigured default, set the parameter to any value other than zero.

For more information on the *xnetrc* file, see "Network Parameter Descriptions" earlier in this chapter.

Calculating the Maximum Buffer Size

The maximum allowable size of the network buffer (NBIOSIZ) is determined by the size of the largest streams buffer configured on your machine. Use this formula to calculate the maximum size of the network buffer:

Max Network Buffer size = (Max Streams Buffer Size * 2) – 48 bytes

To find out what the maximum streams buffer is on your machine, check these parameters:

nblk8172	N8K	0
nblk4096	N4K	0
nblk2048	N2K	10
nblk1024	N1K	20
nblk512	N512	10
nblk256	N256	10
nblk128	N128	10
nblk64	N64	30
nblk16	N16	60
nblk4	N4	128

These parameters are found in the file */etc/conf/cf.d/mtune*.

These parameters set the number of streams buffers of various sizes, from 4 bytes to 8 Kbytes. In the previous example, there are no 8- or 4-Kbyte buffers configured, there are ten 2-Kbyte buffers, thirty 64-byte buffers, and so on. Because no 8- or 4-Kbyte buffers are configured, the maximum streams buffer size configured in the example is 2 Kbytes, or 2048 bytes. Thus, the maximum network buffer size is 4048 bytes. Rounding down to the nearest 1-Kbyte increment, this gives us a maximum network buffer size of 3 Kbytes. By configuring some 4-Kbyte streams buffers, you can raise this value to 7 Kbytes.

For more information on the streams buffer parameters, see "Network Parameter Descriptions" earlier in this chapter.

Chapter 7

Building a Remote Network with UUCP

This chapter explains how to use the ODT-NET UUCP package to build a remote network system for your computer using a normal telephone line and a modem.

NOTE: UUCP is not a terminal emulation program. If you want to use your modem to dial into another computer and log on, you should refer to *Administering ODT-OS* in the *Administrator's Guide* and follow the instructions for adding dial-in and dial-out modems. If you plan to do extensive file transfers between UNIX and XENIX systems, you should set up a UUCP connection.

What Is UUCP?

The UUCP package permits UNIX and XENIX systems to communicate as part of a remote network. The name UUCP is an acronym for "UNIX-to-UNIX Copy." The UUCP package consists of a group of programs that provide the following capabilities:

- Remote file transfer (**uucp**)
- Remote command execution (**uux**)
- Mail to and from remote sites (via **mail**)

Used primarily over phone lines, UUCP can be used to connect with specific remote machines on a demand or scheduled basis, and by either dialing out or allowing other machines to call in.

UUCP uses a batch method to manage communications traffic, storing (or "spooling") requests for later execution when actual contact is made between systems. When UUCP commands are executed, work files and any data files needed are created in */usr/spool/uucp*. The program **uucico** scans this directory for the instructions contained in any work files and executes them. Although it is possible to execute commands immediately, most systems call other systems according to a daily schedule (usually during the evenings to reduce connection costs).

How to Use This Chapter

This chapter describes how to build a UUCP system and covers both hardware installation and software configuration. There are also sections on routine maintenance and troubleshooting.

The following is an outline of what must be done to set up your UUCP network:

1. Connect and configure a modem or direct wire.
2. Configure the UUCP software using **uuinstall**.
3. Create login accounts for any sites that will be calling your system.
4. Test your connections with each remote site.

The most important task of configuring UUCP is the editing of several control files that act as the database for UUCP. The next few sections describe the function of these files, and "Configuring UUCP on Your System" explains the information that these files contain. The **uuinstall** utility edits these files for you and explains each entry. **uuinstall** also includes an extensive help facility. Read "Configuring UUCP on Your System" carefully before running **uuinstall** to understand the UUCP database.

What You Need

To set up your UUCP communication system, you need:

- At least one RS-232 serial line (or serial port) on your computer to use for UUCP.
- The UUCP and MAIL packages extracted from your UNIX System distribution using **custom**(ADM).
- A modem. Supported modems include models by Hayes, Penril, Ventel, Vadic, Rixon, AT&T, and Telebit. You can supply **Dialers** entries or dialer programs for other modems. (For best results, use dialer programs.) Instructions for the Hayes Smartmodem 1200 or 2400 and compatibles are given later in this chapter.
- A standard telephone jack for access to the telephone system.
- A cable to connect the serial port to the modem.

UUCP Commands

UUCP programs are divided into two categories: user programs and administrative programs. The paragraphs that follow describe the programs in each category.

User Programs

The user programs for basic networking are in */usr/bin*. No special permission is needed to use these programs. These commands are all described in *Using ODT-NET* in the User's Guide.

Administrative Programs

Most of the administrative programs, control files, and scripts are in */usr/lib/uucp*. Two exceptions are **uuinstall** and **uulog**, which are in */etc* and */usr/bin*, respectively.

uulog Displays the contents of a specified computer's log files. Log files are created for each remote computer your computer communicates with. The log files contain records of each use of **uucp**, **uuto**, and **uux**.

uuclean Cleans up the spool directory. It is normally executed from a shell script called **uudemon.clean**, which can be set up to be run by **cron**.

uutry Tests call-processing capabilities and does a moderate amount of debugging. It invokes the **uucico** daemon to establish the communications link.

uucheck Checks for the presence of basic networking directories, programs, and support files. It also checks the **Permissions**, **Systems**, and **Devices** files for syntax errors.

uuinstall Configuration script for UUCP control files and ports.

UUCP Directories

There are three directories associated with UUCP:

/usr/spool/uucp	The working directory for UUCP. Work files, log files, and all UUCP communications traffic are stored here.
/usr/spool/uucppublic	This is the publically readable or writable target directory used for most file transfers.
/usr/lib/uucp	Most of the UUCP programs are stored here, as well as the supporting database/control files. The main user programs, including **uux** and **uucp**, are found in */usr/bin*.

/usr/lib/uucp also contains configuration files for UUCP (distinguished by their capitalized names). The most important to understand are:

Systems	Contains information needed to establish a link to a remote computer, including the name of the connecting device associated with the remote computer, when the computer can be reached, telephone number, login sequence, and password.
Permissions	Defines the access level granted to computers when they attempt to transfer files or remotely execute commands on your computer.
Devices	Contains information concerning the port name, speed, and type of the automatic call units (modems), direct links, and network devices.

UUCP Background Programs

uucp traffic is managed by three *daemons*, or supervisory programs, that run in the background, handling file transfers and command executions. (The daemons can also be executed manually as commands.)

uucico	Selects the device used for the link, establishes the link to the remote computer, performs the required login sequence and permission checks, transfers data and executes files, logs

results, and (if requested) notifies the user by **mail** of transfer completions. When the local **uucico** daemon calls a remote computer, it "talks" to the **uucico** daemon on the remote computer during the session.

uuxqt Executes remote program execution. It searches the spool directory for execute files (X.*file*) that were sent from a remote computer. When an X.*file* file is found, **uuxqt** opens it to get the list of data files that are required for the execution. It then checks to see if the required data files are available and accessible. **uuxqt** also verifies that it has permission to execute the requested command.

uusched Schedules the queued work in the spool directory. Before starting the **uucico** daemon, **uusched** randomizes the order in which remote computers are called.

How UUCP Works

When you enter a UUCP command, the program creates a work file and usually a data file for the requested transfer. The work file contains information required for transferring the file(s). The data file is a copy of the specified source file. After these files are created in the spool directory, the **uucico** daemon is started.

The **uucico** daemon attempts to establish a connection to the remote computer. First it gathers the information required for establishing a link to the remote computer from the **Systems** file. This is how **uucico** knows what type of device to use in establishing the link. Next, **uucico** searches the **Devices** file looking for the devices that match the requirements listed in the **Systems** file. After **uucico** finds an available device, it attempts to establish the link and log in on the remote computer.

When **uucico** logs in on the remote computer, it starts the **uucico** daemon on the remote computer. The two **uucico** daemons then negotiate the line protocol to be used in the file transfer(s). The local **uucico** daemon then transfers the file(s) that you are sending to the remote computer. The remote **uucico** places the file in the specified pathname(s) on the remote computer. After your local computer completes the transfer(s), the remote computer may send files that are queued for your local computer. The remote computer can be denied permission to transfer these files with an entry in the **Permissions** file. (This is also affected by directory permissions.) If this is done, the remote computer must establish a link to your local computer to perform the transfers. A remote computer can also request files.

If the remote computer or the device selected to make the connection to the remote computer is unavailable, the request remains queued in the spool directory. If set up to run by **cron**, each hour (default), **uudemon.hour** starts the **uusched** daemon. When the **uusched** daemon starts, it searches the spool directory for the remaining work files, generates the random order in which these requests are to be processed, and then starts the transfer process (**uucico**) described in the previous paragraphs.

A Sample UUCP Transaction

The following traces the execution of a **uucp** command:

1. A user on a system called "kilgore" wishes to send a copy of the file *minutes.01.10* to a remote system called "obie". To accomplish this, the user enters the following command:

   uucp minutes.01.10 obie\!usr/spool/uucppublic

   Note that the exclamation point need only be escaped (preceded by a "\") if the **csh** is used; the Bourne shell (**sh**) does not require this.

2. A work file is created in the */usr/spool/uucp/obie* directory, C.obie*xxxx*, where *xxxx* is the job number.

3. The **uusched** daemon schedules the request for execution by **uucico**.

4. When the execution time is reached, **uucico** first checks the **Systems** file and confirms that "obie" is a recognized system and that a call is permitted at this time.

5. Using the information in the **Systems** file, **uucico** next locates the modem device and tty port associated with it as stored in the **Devices** file.

6. Using the phone number in the **Systems** file and the modem type from the **Devices** file, **uucico** uses the appropriate modem commands from the **Dialers** file (or runs a dialer program from the */usr/lib/uucp* directory) to connect to the remote system.

UUCP Control Files (sites: *kilgore* and *obie*)

Systems:

```
obie Any ACU 2400 14081234567 \
   --ogin:-BREAK-ogin: nuucp ssword: mavra
```

Devices:

```
ACU tty1A - 2400 dialHA24
```

Permissions:

```
LOGNAME= ukilgore MACHINE= kilgore \
  READ=/usr/spool/uucppublic:/usr/kilgore \
  WRITE=/usr/spool/uucppublic:/usr/kilgore \
  REQUEST=no  SENDFILES=call \
  COMMANDS=rmail:rnews:uucp
```

7. **uucico** creates a lock file (LCK..tty*xx*) to lock the serial line, and a lock file (LCK..obie) to lock the called system in the directory */usr/spool/uucp*.

8. **uucico** uses the login sequence and password defined in the **Systems** file to log-in to "obie", whose own **uucico** confirms that "kilgore" is recognized before beginning the actual transaction.

9. The calling system, "kilgore," (sometimes known as the *guest*) is said to be the "master" of the transaction; the called system, "obie," (also known as the *host*) is said be the "slave." The slave **uucico** checks the local **Permissions** file to confirm that the master is authorized to transfer the file.

10. The master ("kilgore") transmits the file in packets that are checked for errors and retransmitted if garbled. During reception, the file is stored in a temporary file (TM.*xxxx*) in the */usr/spool/uucp* directory. When the transfer is complete, the file is moved to the proper destination, in this case */usr/spool/uucppublic/minutes.01.10*.

11. Each machine records its side of the transaction in log files. For example, "obie" would have the exchange recorded in a file called */usr/spool/uucp/.Log/uucp/kilgore*.

12. Unless the slave system "obie" has requests of its own, a hangup request is sent, the connection is terminated, and the lock files removed.

For remote command execution (via **uux**), an execute X.*file* is created in the */usr/spool/uucp* directory. The **uuxqt** daemon scans this directory for work, checks the **Permissions** file to confirm permission to execute the command, then executes it. The device name should have the form

/dev/tty*nn*

where *nn* is the number of the corresponding line. For example, */dev/tty1a* usually corresponds to COM1. You need the name of the actual line for later steps.

The serial port should be owned by **uucp**. To make sure the line is owned by **uucp** enter this command:

chown uucp /dev/tty*nn*

where *nn* is the number of the corresponding line.

Connect a Serial Cable

You connect two computers together using an RS-232 cable. The actual pin configurations sometimes vary between machines.

Typically, the cable should connect pins 2, 3, and 7 on one computer to the same pins on the second computer. Sometimes the cable must be *nulled*, which means that pin 2 on one machine is connected to pin 3 on the other, and vice versa. Since the connections can vary, check the hardware manuals for each computer to determine the proper pin connections.

Testing a Connection

For this section, *tty2a* is used as the example serial port for both machines.

To test the wire connection between two machines:

1. Disable the serial lines on each machine. On each computer, enter the command:

   disable /dev/tty2a

   Be sure to disable the modem control line as well:

   disable /dev/tty2A

2. Attach one end of the serial wire to one of the machines. Attach the other end to the standard data port of a terminal.

3. Enter this command at the computer:

   (stty 9600; date) < /dev/tty2a > /dev/tty2a

   tty2a is our example serial line, and the date command provides sample output.

   You should see the output of the **date** command appear on the terminal screen. Repeat this procedure on the other machine.

   If this doesn't work, check the following:

   - The wire is plugged in properly at each end.
   - The continuity of the wire.
   - The terminal is configured correctly (baud rate, parity, etc.).
   - The serial line is disabled.
   - You are using the correct pin numbers.

NOTE: An unterminated serial cable can cause serious system problems. Do not leave serial cable dangling.

Connecting Remote UUCP Systems with a Modem

With a modem, you can communicate with computers over standard phone lines. These are the steps to install a modem:

- Choose a serial port.
- Set the dialing configuration.
- Connect the modem and set the switches or registers.
- Test the connection.

The following sections explain each step in detail. Make certain you are aware of special services on your phone line; "call waiting" can disrupt UUCP communications.

Choose a Serial Port

Choose the RS-232 serial port you want to use with the system and connect to the modem. If there are no lines available, you must install a new serial line or make one available by removing any device connected to it. If you remove a terminal, make sure no one is logged in.

Find the name of the device special file associated with the port by referring to *Administering ODT-OS*. The device name should have the form

/dev/tty*nn*

where *nn* is the number of the corresponding port. For example, */dev/tty1A* corresponds to COM1. You need the name of the actual port for later steps.

NOTE: */dev/tty1a* and */dev/tty1A* are the same physical port; **tty1a** is used for terminals and direct connections; *tty1A* is used for modem connections.

The serial port should be owned by *uucp*. To make sure the line is owned by *uucp* enter this command:

chown uucp /dev/tty*nn*

where *nn* is the number of the corresponding line.

Set the Dialing Configuration

The modem can be used to both send and receive calls. You must set the appropriate switches on the modem. The instructions that follow are specific to Hayes-compatible modems, but other modems are supported. You should refer to the modem manual for connection instructions and see the section "Adding Dial-Out Entries to the Devices File" under "Configuring UUCP on Your System" for a complete list of supported modems and dialer programs. (If you are setting up a Hayes Smartmodem 2400 or compatible, see the next section for configuration instructions.) Follow these steps to configure a Hayes Smartmodem 1200 or compatible modem:

1. Remove the front cover of the modem and locate the 8-position configuration switch. (See the modem reference manual for instructions on how to locate the switch on your particular model.)

2. Set the switches as they appear here:

	1	2	3	4	5	6	7	8
up	●	●		●	●	●	●	
down			●					●

Table 7.1 explains each of the settings.

3. Replace the front cover.

Table 7.1.

Hayes-Compatible Switch Settings

Switch	Position	Function
1	up*	Modem responds to DTR from computer
	down	Modem forces DTR high, so no signal is required from computer
2	up*	Result codes in English
	down	Numeric result codes
3	up	No result codes
	down*	Result codes are sent in response to each modem command
4	up*	Commands are echoed
	down	Commands are not echoed
5	up*	Modem will answer phone
	down	Modem will not answer phone
6	up*	CD asserted when carrier is actually present
	down	CD and DSR forced high
7	up*	Modem attached to single-line phone
	down	Modem attached to multi-line phone
8	up	Modem does not recognize dialing commands
	down*	Modem recognizes dialing commands

If you have a different modem, consult your reference manual for the proper switch settings to both send and receive calls.

Connect the Modem

Once your modem's dialing configuration is set, you are ready to connect the modem to your computer. For proper modem operation, the RS-232 cable must provide the pin connections in Table 7.2.

Note that the computer's serial connector must have a DTE (Data Terminal Equipment) configuration. The modem is assumed to have a DCE (Data Communications Equipment) configuration. If both pieces of equipment have DTE or DCE, you need a null modem connection.

Table 7.2.

Pin Connections

Name	Computer (DTE)	Modem (DCE)
Protective Ground	1	1
Transmit Data (TD)	2	2
Receive Data (RD)	3	3
Request to Send (RTS)	4	4
Clear to Send (RTS)	5	5
Data Set Ready (DSR)	6	6
Signal Ground (SG)	7	7
Carrier Detect (CD)	8	8
Data Terminal Ready (DTR)	20	20

These connections are explained in the reference manual for your modem.

Review the installation instructions given in your modem's manual, then follow these steps:

1. Connect the RS-232 serial cable to the serial-line connector on the modem, then to the serial-line connector on your computer. Make sure the cable is fully connected. (A 2-3-7 pin terminal cable is not sufficient. We suggest a ribbon cable to connect all appropriate wires.)

2. Plug the telephone line cable into the "line" or "wall" connector on the modem, then into the wall jack.

3. Plug in the power cord of the modem.

Configuring a Hayes 2400 or Compatible Modem

Although most aspects of modem installation are similar, a Hayes 2400 Smartmodem or compatible modem requires online configuration if it is to be used as a dial-in line. Note that the Hayes 2400 does not answer the phone with a 2400 baud carrier if it was not setup with 2400 baud commands.

1. Make sure that the **Devices** file contains an entry for the line:

   ```
   ACU  ttynn  -  300-2400  /usr/lib/uucp/dialHA24
   ```

2. Disable **getty** temporarily with:

   disable tty*nn*

3. You must then configure the modem by issuing set up commands via **cu**(C). Enter:

   cu -s2400 -l/dev/tty*nn* dir

   where *nn* is the "tty" number of the serial line. Press **Return**.

4. Next, enter the following commands to configure the modem. They are saved in the modem's non-volatile memory. If you do not want to save the settings, do not enter the last command (AT&W). Commands are in the left column and short descriptions of what they do are in the right column. Follow each command with a **Return**:

AT&F	Fetch factory configuration.
ATT	Tone dialing.
ATL0	Low speaker volume.
AT&D2	Set DTR "2": go on hook when DTR drops.
AT&C1	Set dcd "1": dcd tracks remote carrier.
ATS0=1	Answer phone after 1 ring (AA light should come on).
ATS2=128	Disable modem escape sequence.
ATE0	No echo (modem will no longer echo what is sent to it).
ATQ1	Quiet mode (modem will not respond with "OK" after this command or any that follow).
AT&W	Saves settings in non-volatile memory.

5. Exit from **cu** by entering a "tilde" and a "period", followed by **Return**:

   ~.

   (Sometimes it is necessary to press **Return** once before entering the tilde-period.)

6. Re-enable the port only if you wish the modem to receive calls:

   enable tty*nn*

The modem is now configured and ready for testing.

Variable Rate Modems

Some modems can determine the connection baud rate from the carrier sent by a remote system. These modems inform the local system of the connection baud rate before issuing the carrier detect signal. The Hayes 2400 dialer supplied with UUCP detects different connection baud rates and informs UUCP and **cu** when it exits with a successful connection.

The speed fields in **Devices** and **Systems** can specify a range of baud rates for a connection. If a dialer supports baud rates from 300 to 2400 baud, enter the baud rate range in the speed field of **Devices** as follows:

300-2400

If a dialer or modem does not allow variable baud rates, place a single baud in the speed field. If a remote system supports several different speeds, place the range of baud rates in the speed field of **Systems**. If the remote system connects at a single baud rate, place that number in **Systems**. UUCP passes the intersection of the **Systems** and **Devices** baud rate ranges to the dialer when connecting. If the dialer connects outside of the baud range, it returns a bad baud rate error. Otherwise, it returns the baud rate of the connection.

Test the Modem

As the last step of the modem installation, you should test the modem to make sure that it can send and receive calls. Once you have verified that the modem is working, you can begin to use the communications system.

To test the modem, follow these steps:

1. If you are using a Hayes 1200 or compatible, make sure the volume switch on the modem is at an appropriate level. You must be able to hear the modem to carry out this test successfully. Refer to your modem reference manual for the location of this switch.

2. Ensure that the **Systems** file has an entry for the system you intend to call, and that the **Devices** file has a matching entry for tty*nn*.

3. Start the **uutry** program by entering:

   /usr/lib/uucp/uutry -x6 *sitename*

4. Listen carefully to the modem. You should hear each digit as the number is dialed, then hear a high-pitched signal when the other modem connects, followed by silence.

5. The dialer automatically disconnects any call that it cannot complete. Do not interrupt using **Del** or otherwise stop **uutry**. Let the dialer hang up.

6. If the signal is not present, make certain:

   - you have connected the modem to the telephone jack
   - the jack is connected to the phone system
   - you gave the correct phone number in the **Systems** file

7. If you do not hear the modem dial, make certain:

   - the volume switch is up
   - the modem is connected to the correct serial line and that the cable connection is tight
   - you gave the correct tty line in the **Devices** file
   - modem's power is on
   - there are no LCK.. files in */usr/spool/uucp*.

8. **uucico** only allows one call to a given system every ten minutes. You can wait before retrying, or remove the file associated with the site you are calling in the directory */usr/spool/uucp/Status*.

Configuring UUCP on Your System

To configure your UUCP system, you must edit a series of files that contain information about, and control the actions of the UUCP programs. The UUCP control files are in the */usr/lib/uucp* directory. You can modify these files with a standard text editor, or use the **uuinstall**(ADM) program as described here. The descriptions in the latter part of this section provide details on the structure of these files so you can edit them manually.

An Important Consideration: Call or Be Called?

There are three ways to configure a UUCP site:

- a *dial-in* only site.
- a *dial-out* only site.
- a dial-in/out site.

As a dial-in site, other computers call up and log in to your system. They can transfer files and execute certain commands.

As a dial-out site, your computer calls up other computers and logs in. Your computer initiates file transfers to and from the remote machine, as well as local and remote command execution.

Setting Up the Control Files with uuinstall

The rest of this section is concerned with the configuration or control files that act as the UUCP database. The **uuinstall**(ADM) utility provides a simple way to configure these files. Read the rest of the chapter to familiarize yourself with the descriptions of each file and the entries required. The **uuinstall** utility includes a complete series of help files (accessed by pressing **?** while in the menus) so it is not necessary to keep referring to the documentation. When you have some understanding of how each of the control files is used, follow this procedure:

1. Invoke **uuinstall** by logging in as **root** and entering the following command:

   /etc/uuinstall

Δ **sysadmsh** users select: System→Configure→Network→UUCP

The main **uuinstall** menu is displayed:

```
          UUCP Administration Utility
          ===========================

1. Display or update site name
2. Display or update list of remote sites       (Systems)
3. Display or update direct- or dial-out lines  (Devices)
4. Display or update direct- or dial-in lines
5. Check consistency of UUCP files
6. Test connection with remote site
7. Convert old UUCP files to new format

Choose an option (1-7), or enter "q" to quit :
```

Use the **uuinstall** options as follows:

- If you did not set your site name at installation time, or you wish to change your site name, do so using the first option.
- Choose the devices to be used for dialing-in or out and enter them in the **Devices** file using the "Display or update dial-in or dial-out devices" option.
- Identify sites your system will have contact with by creating entries in the **Systems** file with the "Display or update list of remote sites" option.
- Add the tty lines to be used to the */etc/inittab* file using the "Display or update line connections" option.

NOTE: If you wish any changes made to */etc/inittab* to be permanent, you must also make the same change to */etc/conf/cf.d/init.base*. This is because each time the kernel is relinked (as when a driver is added or tunable parameter changed), */etc/inittab* is reconstructed from the entries found in */etc/conf/cf.d/init.base*.

2. If other systems will be calling yours, create login accounts as described in "Creating Login Accounts for Sites Dialing-In" later in this section.
3. If other systems will be calling yours, define a security scheme that includes what commands and directories can be used in the **Permissions** file.

When you install the UUCP system, or make any modifications, you should be logged in as super user (root). Virtually all of the UUCP files are writable only by the super user, and many of them are also readable and executable only by **root** and **uucp**. Make sure when you are done that all of the UUCP files are owned by **uucp** and not **root**. UUCP does not work correctly if it cannot read or execute its files. To check the permissions of the UUCP files, use the following commands:

```
cd /
fixperm -n -v -dUUCP /etc/perms/*
```

This command displays any UUCP files with incorrect permissions.

NOTE: The files **Systems** and **Permissions** contain unencrypted passwords, and they should, therefore, be readable only by **uucp** (and **root**). Note also that the program */usr/bin/ct* must be owned by **root** and not by **uucp** to work correctly.

Changing your Site Name

Use the **uuinstall** utility to change the name of your UUCP site. If you wish to change your sitename manually, or wish to maintain different names on different networks, refer to the *Administering ODT-OS* in the *Administrator's Guide*.

Selecting and Defining a UUCP Port

As discussed earlier, you must select a serial port, disable it if it is to be used for dial-out only, or enable it for dial-in, and edit the serial line entry in the */etc/inittab* file.

NOTE: If you wish any changes made to */etc/inittab* to be permanent, you must also make the same change to */etc/conf/cf.d/init.base*. This is because each time the kernel is relinked (as when a driver is added or tunable parameter changed) */etc/inittab* is reconstructed from the entries found in */etc/conf/cf.d/init.base*.

1. Select the serial line. Use a line with modem control (for example, */dev/tty1A*) for a dial-in or dial-out line, or a line without modem control (for example, */dev/tty2a*) for a direct connection. For more information, see "Choose a Serial Port" earlier in this section.

2. Disable the serial line. If you are using a modem, be sure it is installed and tested. If the serial line is to be a dial-in line, substitute **enable** for disable and enter the command:

   disable /dev/tty*nn*

   where *nn* is the number of your serial line. If the line is already disabled/enabled, the command displays an error message that you can safely ignore.

3. Edit the */etc/inittab* file. This file contains a list of possible login terminals. Enter the following command to display the current entries for the different serial lines:

   cat /etc/inittab

   tty entries have the following form:

   t*n*:2:respawn:/etc/getty tty*n* m

   where *n* is the **tty** number. If you need to change an entry, you can do so with a text editor. For more information on the */etc/inittab* file and the various control codes, see the **getty**(M) and **inittab**(F) manual pages.

   For example, an entry for a dial-out line (connected to a modem) might

   look like this:

```
t2A:2:respawn:/etc/getty tty2A m
```

   An example entry for a direct line between two computers might be:

```
t2a:2:respawn:/etc/getty tty2a m
```

   If the line is to be shared between dial-in and dial-out, ensure that it has an appropriate entry in */usr/lib/uucp/Devices* and in */etc/inittab*.

Creating Login Accounts for Dial-in Sites

A dial-in site must provide a login entry for the sites that call it. These entries are placed in the */etc/passwd* file.

A UUCP login entry has the same form as an ordinary user login entry but it has a special login directory and login program instead of the normal user directory and shell. (Refer to *Administering ODT-OS* in the Administrator's Guide for more information on creating login accounts)

NOTE: "uucp" should not be used as the name of a UUCP user or login account; it is the name of the uucp owner/administrator.

To create a UUCP login entry, follow these steps:

1. Choose a new user name and a user ID (identification number) for the UUCP login. The name can be any combination of letters and digits that is no more than eight characters long. The user ID must be an integer in the range 50 to 65535.

   Make sure the name and ID are unique. A UUCP login entry must not have the same name or ID as any other login entry.

2. To create the new account, invoke the **sysadmsh** and make the following selection:

   Accounts→User→Create

3. Use the following information to create the account:

   Login shell: /usr/lib/uucp/uucico
   Home directory: /usr/spool/uucppublic

   Passwords are optional, but recommended, for UUCP logins.

Adding Entries for Remote Sites to the Systems File

The **Systems** file (*/usr/lib/uucp/Systems*) contains the information needed by the **uucico** daemon to establish a communications link to a remote computer. Each entry in the file represents a computer that can be called by your computer.

NOTE: After creating the **Systems** file, and each time you modify it, you must log in as user **mmdf** and execute the following commands:

```
cd /usr/mmdf/table
tools/uulist
dbmbuild
```

This ensures that the MMDF routing mechanism properly handles traffic for the new or modified sites.

In addition, the **Systems** file can be configured to prevent any computer that does not appear in this file from logging in on your computer. More than one entry may be present for a particular computer. The additional entries represent alternative communication paths that will be tried in sequential order.

NOTE: If you are setting up your system as a *dial-in* only (passive) site that never initiates calls, you only need to add the names of the systems that will be calling you.

Each entry in the **Systems** file has the following format (each field must be separated by a space):

sitename schedule device speed phone login-script

where:

sitename	Contains the node name of the remote computer.
schedule	Is a string that indicates the day-of-week and time-of-day when the remote computer can be called.
device	Is the device type that should be used to establish the communications link to the remote computer.
speed	Indicates the transfer speed of the device used in establishing the communications link.

phone	Provides the phone number of the remote computer for automatic dialers.
login-script	contains login information (also known as a "chat script").

The Schedule Field

The *schedule* consists of three subfields. The first, *day*, is required. The other two, *time* and *retry*, are optional. The syntax is as follows:

day[*time*][*;retry*]

The *day* subfield can contain the following keywords:

Su Mo Tu We Th Fr Sa	For individual days.
Wk	For any weekday (Mo Tu We Th Fr).
Any	For any day.
Never	For a passive arrangement with the remote computer. If the *schedule* field is **Never**, your computer never initiates a call to the remote computer. The call must be initiated by the remote computer. In other words, your computer is in a passive mode in respect to the remote computer (see discussion of **Permissions** file).

The optional *time* subfield should be a range of times in 24-hour clock format, such as 0800-1230. If no *time* is specified, any time of day is assumed to be allowed for the call. A time range that spans 0000 is permitted. For example, **0800-0600** means all times are allowed other than times between 6 am and 8 am.

For example, the following permits calls on Mondays, Wednesdays, and Fridays between the hours of 9 am and noon (the *schedule* field is in boldface for clarity):

```
grebe MoWeFr0900-1200 ACU D1200 14087672676 \
    ogin: nuucp ssword: Crested
```

You can also specify more than one set of *day* and *time* entries by separating them with commas. This is useful for more complex specifications. The following example allows calls from 5:00 pm to 8:00 am, Monday through Thursday, and calls any time Saturday and Sunday.

The following example would be an effective way to call only when phone rates are low, if immediate transfer is not critical:

```
gorgon Wk1700-0800,SaSu ACU D1200 14087672676 \
    ogin: nuucp ssword: DontLook
```

The optional subfield, *retry*, is available to specify the minimum time (in minutes) before a retry, following a failed attempt. The subfield separator is a semicolon (;). For example, the following is interpreted as "call any time, but wait at least 9 minutes before retrying after a failure occurs":

```
Any;9
```

NOTE: By default, UUCP uses an "exponential backoff" method to retry failed calls. After the initial failure, a second call is made in 5 minutes. This interval expands as the number of unsuccessful attempts increases. The retry field is used to override the default.

The Device Field

The *device* field selects the device type, in most cases an ACU (Automatic Calling Unit). For example, the keyword used in the following field is matched against the first field of **Devices** file entries:

Systems:
```
gorgon Any ACU D1200 14087672676 \
  ogin: nuucp ssword: DontLook
```

Devices:
```
ACU tty2A - D1200 hayes
```

The Speed Field

This field can contain a letter and speed (for example, C1200, D1200) to differentiate between classes of dialers (refer to the discussion on the **Devices** file, *speed* field). Some devices can be used at any speed, so the keyword **Any** can be used. However, we recommend that you specify the

actual range of speeds that can be used. (If **Any** is used in both **Systems** and **Devices** entries, 1200 is assumed.) For example, this field must match the *speed* field in the associated **Devices** file entry:

```
Systems:   gorgon Any ACU D2400-9600 14087672676 \
             ogin: nuucp ssword: DontLook

Devices:   ACU ttyla - D2400-9600 hayes2400
```

If information is not required for this field, use a hyphen (-) as a place holder for the field.

The Phone Field

This field is used to provide the phone number used for the modem dialer. The phone number is made up of an optional alphabetic abbreviation and a numeric part. If an abbreviation is used, it must be one that is listed in the **Dialcodes** file. For example:

```
Systems:    gorgon Any ACU D1200 CA2676 \
              ogin: nuucp ssword: DontLook

Dialcodes:  CA 9=408767
```

In this string, an equal sign (=) tells the ACU to wait for a secondary dial tone before dialing the remaining digits. A dash in the string (-) instructs the ACU to pause 2 seconds before dialing the next digit.

If your computer is connected to a LAN switch or port selector, you can access other computers that are connected to that switch. The **Systems** file entries for these computers will not have a phone number in the *phone* field. Instead, this field contains the token that must be passed on to the switch so it knows which computer your computer wishes to communicate with. (This is usually just the system name.) The associated **Devices** file entry should have a **\D** at the end of the entry to prevent translation using the **Dialcodes** entry.

The Login-Script Field

The login-script is used to open communications between modems, plus recognize and send proper login and password sequences. The script is given as a series of space-separated fields and subfields of the following format:

expect send

where *expect* is the string that is received, and *send* is the string that is sent when the *expect* string is received.

The *expect* field can be made up of subfields of the following form:

expect[-*subsend-subexpect*]...

where the *subsend* is sent if the prior *expect* is not successfully read and the *subexpect* following the *subsend* is the next expected string. To make this distinction clear: the send-expect sequence sends a string if the expect string is received, the subsend-subexpect sends only if the prior expect string is not received within 10 seconds.

For example, with "login--login", the UUCP program expects "login". If a "login" is received, it goes on to the next field. If it does not get "login", it sends nothing followed by a carriage return, then looks for "login" again. If no characters are initially expected from the remote computer, the characters "" (null string) should be used in the first *expect* field. Note that all *send* fields are sent followed by a carriage return unless the *send* string is terminated with a \c.

If an *expect* string starts with a dash, it is interpreted as a null *expect* string followed by a *subsend* string. For example, "--login:" sends a carriage return and then expects a "login:".

The *expect* string need not be complete; only the trailing characters must be specified, as in "ogin:". This avoids difficulties with login strings that use an uppercase letter as in "Login:" or "Password:", and also difficulties when the line is shared by dial-in and dial-out.

Creating Login Scripts

This section explains in greater detail how to create a login (chat) script.

Consider the following sample **Systems** file entry:

```
terps Any ACU 1200 18005211980 "" \r ogin:-BREAK-ogin: \
  uucpx word: ichore
```

This is how this script would work during connection:

1. Nothing is expected initially.
2. A carriage return is sent and the script waits for the prompt "ogin:" (login:).
3. If it does not receive "ogin:", send a BREAK signal.
4. When "ogin:" is finally received, send the login name **uucpx**.
5. When the prompt "word:" (for Password:) is received, send the password "ichore".

Login (chat) scripts often require some experimentation. There are cases that require one or more BREAK sequences before presenting a login (this is often true with variable speed modems). If you cannot obtain the necessary login sequence from the system administrator for a given site, it is a good idea to connect with the site manually. You can accomplish this using **cu** and find out what must be sent to generate a login prompt. (You can also connect with a system using a **uutry** for debugging; see "Debug Transmissions" under "Troubleshooting" for details.) There are several escape characters that cause specific actions when sent during the login sequence, some of which correspond to keystrokes; these should be included in the script where necessary. See Table 7.3.

Table 7.3.

Login (Chat) Script Escape Sequences

Character	Description
\N	Sends a null character (ASCII NUL).
\b	Sends or expects a backspace character.
\c	If at the end of a string, suppresses the carriage return that is normally sent. Ignored otherwise.
\d	Delays two seconds before sending or reading more characters.
\p	Pauses for approximately ¼ to ½ second.
\E	Starts echo checking. (From this point on, whenever a character is transmitted, it waits for the character to be received before doing anything else.)
\e	Turns echo check off.
\n	Sends or expects a new-line character.
\r	Sends or expects a carriage-return.
\s	Sends or expects a space character.
\t	Sends or expects a tab character.
\\	Sends or expects a \character.
EOT	Sends EOT (end of transmission or **Ctrl**-d)
BREAK	Sends a BREAK signal.
\K	Same as BREAK.
\ddd	Collapses the octal digits (ddd) into a single character.
""	Expects a null string.

Limiting Access with the Permissions File

If other machines will be dialing into your system, the **Permissions** file (*/usr/lib/uucp/Permissions*) specifies the permissions that remote computers have with respect to login, file access, and command execution. There are options that restrict the remote computer's ability to request files and its ability to receive files queued by the local site. Other options specify the commands that a remote site can execute on the local computer.

Structuring Permissions File Entries

Each entry is a logical line with physical lines terminated by a \ to indicate continuation. Entries are made up of options delimited by spaces. Each option is a name-value pair in the following format:

name=value

Note that no spaces are allowed within an option assignment.

Comment lines begin with a crosshatch sign (#) and they occupy the entire line up to a newline character. Blank lines are ignored (even within multi-line entries).

There are two types of **Permissions** file entries:

LOGNAME	Specifies the permissions that take effect when a remote computer calls your computer.
MACHINE	Specifies permissions that take effect when your computer calls a remote computer.

Permissions File Restrictions

When using the **Permissions** file to restrict the level of access granted to remote computers:

- All login IDs used by remote computers to log in for UUCP communications must appear in only one LOGNAME entry.
- Any site that is called whose name does not appear in a MACHINE entry, has the following default permissions/restrictions:
  - — Only local send and receive requests are executed.
  - — The remote computer can send files to your computer's */usr/spool/uucppublic* directory.
  - — The commands sent by the remote computer for execution on your computer must be one of the default commands, usually **rmail**.

NOTE: When a remote machine calls you, unless you have a unique login and password for that machine, you do not know if the machine is who it claims to be.

Permissions Options

This section lists some of the available options. See the examples at the end of this chapter and the **permissions**(F) manual page for details.

REQUEST
Specifies whether the remote computer can request to set up file transfers from your computer.

SENDFILES
Specifies whether your computer can send the work queued for the remote computer. When a remote computer calls your computer and completes its work, it may attempt to take work your computer has queued for it.

READ and **WRITE**
Specify the various parts of the file system that **uucico** can read from or write to. The READ and WRITE options can be used with either MACHINE or LOGNAME entries.

NOREAD and **NOWRITE**
Specify exceptions to the READ and WRITE options or defaults.

COMMANDS
Specifies the commands in MACHINE entries that a remote computer can execute on your computer. This affects the security of your system; use it with extreme care.

VALIDATE
Used in conjunction with the COMMANDS option when specifying commands that are potentially dangerous to your computer's security. It provides a certain degree of verification of the caller's identity.

Adding Dial-Out Entries to the Devices File

The **Devices** file (*/usr/lib/uucp/Devices*) contains information for all the devices that can be used to establish a link to a remote computer. Devices are Automatic Call Units, direct links, or network connections. This file works closely with the **Dialers**, **Systems**, and **Dialcodes** files. Before you make changes in any of these files, you should be familiar with them all. A change to an entry in one file may require a change to a related entry in another file.

Each entry in the **Devices** file has the following format:

type ttyline dialerline speed dialer-token

where:

type Can contain one of two keywords (**direct** or **ACU**), the name of a Local Area Network switch, or a system name.

ttyline Contains the device name of the port associated with the **Devices** entry. For example, if the automatic dial modem for a particular entry was attached to the */dev/tty1A* line, the name entered in this field would be *tty1A*.

dialerline This option is useful only for 801 type dialers, which do not contain a modem and must use an additional line. Unless you have an 801 dialer, simply enter a hyphen (-) as a placeholder.

speed is the speed or speed range of the device. Can also contain an indicator for distinguishing different dialer classes.

dialer-token This field contains pairs of dialers and tokens, each representing a dialer and an argument to be passed to it. The *dialer* portion can be the name of an automatic dial modem, or **Direct** for a direct link device.

The Type Field

This field can contain one of two keywords (**Direct** or **ACU**), the name of a Local Area Network switch, or a system name:

Direct This keyword indicates a direct link to another computer or a switch for **cu** connections.

ACU This keyword indicates that the link to a remote computer is made through an Automatic Call Unit. This modem can be connected either directly to your computer or indirectly through a Local Area Network (LAN) switch.

LANswitch can be replaced by the name of a LAN switch. **micom** and **develcon** are supplied with caller scripts in the **Dialers** file.

sysname indicates a direct link to a particular computer. (*sysname* is replaced by the name of the computer.) This means that the line associated with this **Devices** entry is for a particular computer in the **Systems** file.

For example the keyword "gorgon" used in the *Type* field **Devices** is matched against the third field of the **Systems** file entry:

```
Devices:   gorgon ttyla - 1200 hayes1200

Systems:   gorgon Any ACU 1200 14087672676 ogin: nuucp \
             ssword: DontLook
```


The Speed Field

In most cases, this is simply the speed of the device, if the keyword **ACU** or **Direct** is used in the *type* field. However, *speed* can contain a letter and a speed (for example, C1200, D1200) to differentiate between classes of dialers (Centrex or Dimension PBX). This is necessary because many larger offices may have more than one type of telephone network: one network may be dedicated to serving only internal office communications, while another handles the external communications. It is necessary to distinguish which lines are used for internal communications and which are used for external communications. The keyword used in the *speed* field of the **Devices** file is matched against the fourth field of **Systems** file entries, for example:

```
Devices:   ACU ttyla - D1200 hayes1200

Systems:   gorgon Any ACU D1200 3251 ogin: nuucp \
             ssword: DontLook
```

Some devices can be used at any speed, so the keyword **Any** can be used in the *speed* field. If **Any** is used, the line matches any speed requested in a **Systems** file entry. If this field is **Any** and the **Systems** file *speed* field is **Any**, the speed defaults to 1200 bps. If a device can be used at a range of speeds, then the speed field can specify this range (for example, 1200-9600 or D1200-9600). This is preferable to the use of **Any**.

The Dialer-Token Field

NOTE: For best results, dialer programs are preferred over **Dialers** entries. The following entry is an example of an entry using a dialer binary:

```
ACU  ttynn  -  300-2400  /usr/lib/uucp/dialHA24
```

The following binary types are provided in *usr/lib/uucp*:

Binary File	Modem
dialHA12	Hayes Smartmodem 1200 or compatible
dialHA24	Hayes Smartmodem 2400 or compatible
dialVA3450	Racal Vadic 3451 modem
dialTBIT	Telebit Trailblazer Modem

The source is provided for these dialer binaries; you can adapt and compile your own dialers if desired.

Structuring Dialer-Token Entries

The *dialer-token* can be structured four different ways, depending on the device associated with the entry:

- **Simple modem connection.** If an automatic dialing modem is connected directly to a port on your computer, the *dialer-token* field of the associated **Devices** file entry only has one pair. This pair would normally be the name of the modem. This name is used to match the particular **Devices** file entry with an entry in the **Dialers** file. Therefore, the *dialer* field must match the first field of the following **Dialers** file entry:

```
Devices:   ACU ttyla - 1200 ventel

Dialers:   ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
```

Notice that only the *dialer* portion (**ventel**) is present in the *dialer-token* field of the **Devices** file entry. This means that the *token* to be passed on to the dialer (in this case the phone number) is taken from the *Phone* field of a **Systems** file entry. (\T is implied; see the last item, "Modems used with a local network switch.") Backslash sequences are described later.

- **Direct links.** If a direct-link is established to a particular computer, the *dialer-token* field of the associated entry contains the keyword **direct**. This is true for both types of direct link entries, **direct** and *sysname* (refer to discussion on the *type* field).

- **Local network switches.** If a computer that you wish to communicate with is on the same local network switch as your computer, your computer must first access the switch and the switch can make the connection to the other computer. In this type of entry, there is only one pair. The *dialer* portion is used to match a **Dialers** file entry following:

```
Devices:   develcon tty13 - 1200 develcon \D

Dialers:   develcon "" "" \pr\ps\c est:\007 \E\D\e \007
```

As shown, the *token* portion is **\D**, which indicates that it is retrieved from the **Systems** file without translation. The **Systems** file entry for this particular computer will contain the token in the *phone* field; this is normally reserved for the phone number of the computer (refer to **Systems** file, *phone* field). The **\D** ensures that the contents of the *phone* field is not interpreted as a valid entry in the **Dialcodes** file.

- **Modems used with a local network switch.** If an automatic dialing modem is connected to a switch, your computer must first access the switch and the switch will make the connection to the automatic dialing modem. This type of entry requires two *dialer-token-pairs*. The following *dialer* portion of each pair (fifth and seventh fields of entry) are used to match entries in the **Dialers** file:

```
Devices:   ACU tty14 - 1200 develcon vent ventel

Dialers:   develcon "" "" \pr\ps\c est:\007 \E\D\e \007
           ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
```

In the first pair, **develcon** is the switch and **vent** is the token that is passed to the **develcon** switch to tell it which device to connect to your computer. This token would be unique for each LAN switch because each switch can be set up differently. Once the ventel modem is connected, the second pair is accessed, where ventel is the dialer and the token is retrieved from the **Systems** file.

The following are two escape characters that can appear in the *dialer-token* field:

\T Indicates that the *Phone* field should be translated at this stage, using the **Dialcodes** file. This escape character is normally placed in the **Dialers** file for each caller script associated with an automatic dial modem (penril, ventel, and so on). The translation will not take place until the caller script is accessed.

\D Indicates that the *Phone* field should not be translated using the **Dialcodes** file. If no escape character is specified at the end of a **Devices** entry, **\D** is assumed by default when a **Dialers** script is to be used (which can itself contain a **\T** to translate the number). **\T** is assumed if a built-in or dialer binary is to be used (because there is then no later opportunity to translate the number).

Using the Same Port for Dialing In and Out

It is possible to dial in and out on the same line without enabling/disabling the line or running a special version of **getty**. All that is necessary is to first create an entry for a line in the **Devices** file (dial-out) and then an entry in */etc/inittab* (dial-in) for the same line. When access to a dial-out line is requested on a shared port, **getty** runs a special program, **uuchat**, that automatically reinitializes the port when the call is complete. **uuchat** uses special dialer scripts found in the **Dialers** file that begin with an ampersand. This means there are actually two entries for some dialers. For example, the dialer for the Hayes Smartmodem 2400 (or compatible) consists of two entries: **hayes2400** and **&hayes2400**, the latter of which is used when reinitializing a shared port to dial-in. In the case of the dialer binaries in */usr/lib/uucp*, these programs are automatically invoked with the **-h** (hangup) switch that reinitializes the port to dial-in.

Administering Your UUCP System

This section discusses the various shell scripts that are used to supervise and maintain UUCP. Consult the section on "Administration and Maintenance Commands" for details on all commands available to the system administrator. Included is an extended description of the */usr/spool/uucp* work directory and a special subsection on troubleshooting.

UUCP Maintenance Shell Scripts

There are several aspects of system operation that are governed by shell scripts running as daemons:

- How often the UUCP directory is checked for work (**uudemon.hour**).
- Polling of sites that are passive (do not originate calls) (**uudemon.poll(2)**).
- Sending of status information to the UUCP administrator (**uudemon.admin**).
- Cleaning of the UUCP spool directory (**uudemon.clean**).

These scripts can be customized and are discussed in **uudemon**(ADM).

Generating Log Reports on UUCP Usage: uulog

The **uulog** program displays log information on UUCP usage according to remote machine. All usage of the programs UUCP, **uuto**, and **uux** are logged in special log files, one per machine. See the **uucp**(C) manual page for more information about **uulog**.

The UUCP Spool Directory

The following is a comprehensive discussion of all files and subdirectories of the UUCP spool directory. These files are created in spool directories to lock devices, hold temporary data, or keep information about remote transfers or executions.

TM. (temporary data file)

These data files are created by UUCP processes under the spool directory (i.e., */usr/spool/uucp/system*) when a file is received from another computer. The *system* directory has the same name as the remote computer that is sending the file. The names of the temporary data files have the format:

TM.*pid.ddd*

where *pid* is a process-ID and *ddd* is a sequential three digit number starting at 0.

When the entire file is received, the **TM.***pid.ddd* file is moved to the pathname specified in the C.*sysnxxxx* file (discussed below) that caused the transmission. If processing is abnormally terminated, the **TM.***pid.ddd* file may remain in the *system* directory. These files should be automatically removed by **uuclean**.

LCK. (lock file)

Lock files are created in the */usr/spool/uucp* directory for each device in use. Lock files prevent duplicate conversations and multiple attempts to use the same calling device. The names of lock files have the format:

LCK..*str*

where *str* is either a device or computer name. These files may remain in the spool directory if the communications link is unexpectedly dropped (usually on computer crashes). The lock files will be ignored (removed) after the parent process is no longer active. The lock file contains the process ID of the process that created the lock. The lock file is always named using the "a" (non-modem control) suffix to avoid possible conflicts if the same line is specified both modem-control and non-modem-control. For example, the lock on */dev/tty1A* is named *LCK..tty1a*.

C. (work file)

Work files are created in a spool directory when work (file transfers or remote command executions) is queued for a remote computer. The names of work files have the format:

C.*sysnxxxx*

where *sys* is the name of the remote computer, *n* is the ASCII character representing the grade (priority) of the work, and *xxxx* is the four-digit job sequence number assigned by UUCP. Work files contain the following information:

- Full pathname of the file to be sent or requested
- Full pathname of the destination or user/filename
- User login name
- List of options
- Name of associated data file in the spool directory. If the **uucp -c** or **uuto -p** option was specified, a dummy name (**D.0**) is used
- Mode bits of the source file
- Remote user's login name to be notified upon completion of the transfer

D. (data file)

Data files are created when it is specified in the command line to copy the source file to the spool directory. The names of data files have the following format:

D.*systmxxxxyyy*

where *systm* is the first five characters in the name of the remote computer, *xxxx* is a four-digit job sequence number assigned by **uucp**. The four-digit job sequence number may be followed by a sub-sequence number, *yyy* that is used when there are several **D.** files created for a work (**C.**) file.

X. (execute file)

Execute files are created in the spool directory prior to remote command executions. The names of execute files have the following format:

X.*sysnxxxx*

where *sys* is the name of the remote computer, *n* is the character representing the grade (priority) of the work, and *xxxx* is a four digit sequence number assigned by UUCP. Execute files contain the following information:

- Requester's login and computer name
- Name of file(s) required for execution
- Input to be used as the standard input to the command string
- Computer and file name to receive standard output from the command execution
- Command string
- Option lines for return status requests

Troubleshooting

The procedures that follow describe how to solve common UUCP problems.

Check for Faulty ACU/Modem

There are two ways you can check if the automatic call units or modems are not working correctly:

- Run **uustat -q**. This command yields counts and reasons for contact failure.
- Run **cu -x9 -l***line*. This permits you to use a specific line and print debugging information during the attempt. Note that this command is only permitted for those who have write access to the **Devices** file, to protect the modem from interference from unqualified users.

Check the Systems File

If you are having trouble contacting a particular machine, ensure that the information in your **Systems** file is current. Some things that could be out of date are:

- Phone number
- Login
- Password

Debug Transmissions

If you are unable to contact a particular machine, you can check out communications to that machine using **uutry** and **uucp**. Do the following:

1. Make contact using this command line:

   /usr/lib/uucp/uutry -r *machine*

   where *machine* is the node name of the problem machine. This command does the following:

   - Starts the transfer daemon (**uucico**) with debugging. You get more debugging information if you are **root**.
   - Directs the debugging output to /tmp/*machine*.
   - Prints the debugging output to your terminal (**tail -f**). Press the **Del** key to end output.

   You can copy the output from /tmp/*machine* if you wish to save it.

2. If **uutry** fails to isolate the problem, attempt to queue a job with the following command:

   uucp -r *file machine*!*/dir/file*

   where *file* is the file you want to transfer, and *machine* is the machine you want to copy to, and *dir/file* is the destination location on the other machine. (Remember that the ! must be escaped (\!) if you are using **csh**.) The **-r** option will queue a job without starting a transfer.

3. Next, use **uutry** again. If you still cannot solve the problem, you may need to call support personnel. Save the debugging output; it will help diagnose the problem.

Check Basic Information

There are several commands you can use to check for basic communications information:

uuname Use this command to list the machines you are set up to contact.

uulog Use this command to display the contents of the log directories for particular hosts.

uucheck -v Run this command to check for the presence of files and directories needed by **uucp**. This command also checks the **Permissions** file and outputs information on the permissions you have set up.

Keeping Traffic and Congestion under Control

The UUCP filesystem can be choked by traffic if a connection goes down, but unless your site is running a full USENET feed or your system connects with a number of systems, UUCP should prove self-sustaining. If UUCP is used more frequently on your system, this section discusses how to ensure that the system does not become stopped, congested, or affect the general performance of your system.

Crowded Directories and Lack of Space

The **uudemon.clean** script is the best way to prevent the UUCP spool directory from growing too large. To see how much disk storage is currently used by UUCP, use the **du**(C) command:

```
du /usr/spool/uucp /usr/spool/uucppublic
```

The current amount of disk space used in each directory is displayed in 512-byte blocks. Divide this number by two for the size in 1K bytes.

The **uudemon.admin and uudemon.clean** scripts send a great deal of mail to the **uucp** account. You should check and clear the mail file periodically.

Running Out of Processes

On systems with a large amount of traffic, you can get error messages indicating that there are too many processes. If you use the **ps**(C) command, you may notice a number of **uucico** or **uuxqt** processes running. You can establish a new limit on the number of these processes by editing the files **Maxuuscheds**(F) and **Maxuuxqts**(F) in */usr/lib/uucp*.

Evaluating Apparent Stoppages

If users complain that UUCP mail is not getting through and the spool directory is filled with old jobs, it is time to check for the source of the stoppage. UUCP provides an extensive set of error messages and log files that should allow you to trace the cause and remedy the situation.

- Use the **uulog**(ADM) command to study traffic on a per-system basis. Error messages in the *.Admin/errors* are called ASSERT errors. These usually involve filesystem problems.
- Find out the status of currently queued jobs using the **uustat -q** command. This command also indicates the number of failed connection attempts.

Error messages are explained in "UUCP Error Messages" in this chapter. Each message is documented with a suggested remedy.

Complete UUCP Examples

This section includes two complete working examples of a UUCP system and the database files.

Example 1: System gomer

The following system (gomer) has:

- 1200 baud modem on tty4B
- direct connection to system (poker) on tty4D for call out only.
- There are three valid uucp logins:

nuucp	The public login for email. No password required.
ubarn	The on site login for system (poker).
upay4	The private login for email and file transfers.

All lines beginning with # are comments and are not required. Most examples are partial listings and may contain other entries. Micnet is not installed. The modem answers at 1200 baud first and is set up for both call in and out.

NOTE: The lines from */etc/passwd* are included here for informational purposes. **Never edit the /etc/passwd file with a text editor**; this could cause serious problems. Always use the **sysadmsh**(ADM) **Accounts→User→Create** or **Accounts→User→Modify** selections to create or alter UUCP login accounts.

/etc/passwd

```
uucp:*:5:5:Uucp admin:/usr/lib/uucp:
nuucp::201:5:public:/usr/spool/uucplogins/nuucp:/usr/lib/uucp/uucico
upay4:*:202:5:private:/usr/spool/uucppublic:/usr/lib/uucp/uucico
ubarn:*:203:5:poker:/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

/etc/group

```
uucp:x:5:uucp,nuucp,ubarn,upay4
```

/etc/systemid

```
gomer
gomer
```

/etc/inittab

```
t4B:2:respawn:/etc/getty tty4B 2
t4b:2:respawn:/etc/getty tty4b m
t4D:2:respawn:/etc/getty tty4D m
t4d:2:respawn:/etc/getty tty4d 2
```

/usr/lib/uucp/Devices

```
# 300-1200 baud hayes 1200 baud modem.
# The Direct tty4b entry is for using cu to call out.
ACU     tty4B  -  300-1200  dialHA12
Direct  tty4b  -  300-1200  dialHA12
poker   tty4d  -  9600      direct
```

/usr/lib/uucp/Permissions

```
# Public uucp login for mail only.
# Can send mail, transfer files to/from uucppublic, and get
# a directory (ls) listing.
LOGNAME=nuucp     MACHINE=OTHER \
       COMMANDS=rmail:ls:uucp \
       READ=/usr/spool/uucppublic:/usr/tmp \
       WRITE=/usr/spool/uucppublic:/usr/tmp \
       SENDFILES=yes   REQUEST=yes
# Private uucp login for mail and file transfer.
# Only dingbat, ogre, grinch, ... can use this login.
LOGNAME=upay4     VALIDATE=dingbat:ogre:grinch:gomer:blitzen \
       COMMANDS=rmail:ls:uucp:who:uux \
       READ=/  WRITE=/ \
       NOREAD=/etc \
       SENDFILES=yes   REQUEST=yes
# Local trusted connection to gomer
# Only gomer can use this login.
LOGNAME=ubarn     VALIDATE=gomer \
       COMMANDS=ALL \
       READ=/  WRITE=/ \
       SENDFILES=yes   REQUEST=yes
```

/usr/lib/uucp/Systems

```
# local calls
dingbat Any ACU 1200 4444444 ogin:-BREAK-ogin:-BREAK-ogin: \
  uubig word: wetrot
# long distance (evening calls only)
grinch Any1800-0700 ACU 2400 18888888 "" \r ogin:-BREAK-ogin: \
  -BREAK-ogin:nuucp
uunet Any1800-0700 ACU 2400 17031111111 ogin:-BREAK-ogin: \
  -BREAK-ogin:xytpq sword: grm5q
# systems that call in as nuucp (for mail) but NOT call out.
daboss Never
sales Never
guru2 Never
```

Example 2: System dingbat

The following system (dingbat) has:

- 2400 baud modem on tty1A.
- There are two valid uucp logins:

nuucp	The public login for email. No password required.
uubig	The private login for email and file transfers.

All lines beginning with # are comments and are not required. Most examples are partial listings and may contain other entries. Micnet is not installed. The modem answers at 2400 baud first and is setup for both call in and out.

/etc/passwd

```
uucp:*:5:5:Uucp admin:/usr/lib/uucp:
nuucp:*:201:5:public:/usr/spool/uucplogins/nuucp:/usr/lib/uucp/uucico
uubig:*:202:5:private:/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

/etc/group

```
uucp:x:5:uucp,nuucp,uubig
```

/etc/systemid

```
dingbat
dingbat
```

/etc/inittab

```
t1A:2:respawn:/etc/getty ttyl1A 2
t1a:2:respawn:/etc/getty ttyl1a m
```

/usr/lib/uucp/Devices

```
# 300-2400 baud hayes 2400 baud modem.
# The Direct entry is for using cu.
ACU  ttyl1A  -  300-2400  dialHA24
Direct  ttyl1A  -  300-2400  dialHA24
```

/usr/lib/uucp/Permissions

```
# Public uucp login for mail only.
# Can send mail, transfer files to/from uucppublic, and get
# a directory (ls) listing.
LOGNAME=nuucp      MACHINE=OTHER \
       COMMANDS=rmail:ls:uucp \
       READ=/usr/spool/uucppublic:/usr/tmp \
       WRITE=/usr/spool/uucppublic:/usr/tmp \
       SENDFILES=yes    REQUEST=yes
# Private uucp login for mail and file transfer.
# Only ogre, grinch, ... can use this login.
LOGNAME=uubig      VALIDATE=ogre:grinch:gomer:blitzen \
       COMMANDS=rmail:ls:uucp:who:uux \
       READ=/  WRITE=/ \
       NOREAD=/etc \
       SENDFILES=yes    REQUEST=yes
```

/usr/lib/uucp/Systems

```
# local calls
gomer Any ACU 1200 3333333 ogin:-BREAK-ogin:-BREAK-ogin: \
  upay4 word: dryrot
# long distance (evening calls only)
grinch Any1800-0700 ACU 1200 18888888 "" \r ogin: \
  -BREAK-ogin:-BREAK-ogin: nuucp
# systems that call in as nuucp (for mail) but NOT call out.
daboss Never
damgr Never
guru2 Never
```

Sample Commands

Sending mail to another system and have it send the mail back.

```
mail othersystem!mysystem!mylogin        (Bourne/korn shell)
mail othersystem\!mysystem\!mylogin      (C-shell)
```

Printing Your System's full mail address.

```
echo "`uuname -l`\!`logname`"
```

Displaying the Systems You Can Call.

```
uuname
```

Forcing a call to another system and save the debug output in background.

```
/usr/lib/uucp/uucico -r1 -x7 -Sother 2>/tmp/uulog$$ &
```


UUCP Error Messages

This section lists the error messages associated with UUCP. There are two types of error messages. ASSERT errors are recorded in the */usr/spool/uucp/.Admin/errors* file. STATUS errors are recorded in individual machine files found in the */usr/spool/uucp/.Status* directory.

ASSERT Error Messages

When a process is aborted, ASSERT error messages are recorded in */usr/spool/uucp/.Admin/errors*. These messages include the filename, SCCS ID, line number, and the text listed in these messages. In most cases, these errors are the result of filesystem problems. The "errno" (when present) should be used to investigate the problem. If "errno" is present in a message, it is shown as ( ) in this list.

Error Message	Description/Action
CAN'T OPEN	An open( ) or fopen( ) failed. Check for the presence of the file and permissions.
CAN'T WRITE	A write( ), fwrite( ), fprint( ), etc. failed. Check for the presence of the file and permissions.
CAN'T READ	A read( ), fgets( ), etc. failed. Check for the presence of the file and permissions.
CAN'T CREATE	A create( ) call failed. Check permissions.
CAN'T ALLOCATE	A dynamic allocation failed.
CAN'T LOCK	An attempt to make a LCK (lock) file failed. In some cases, this is a fatal error.
CAN'T STAT	A stat( ) call failed. Check for the presence of the file and permissions.
CAN'T CHMOD	A chmod( ) call failed. Check for the presence of the file and permissions.
CAN'T LINK	A link( ) call failed. Check for the presence of the file and permissions.
CAN'T CHDIR	A chdir( ) call failed. Check for the presence of the file and permissions.
CAN'T UNLINK	A unlink( ) call failed.
WRONG ROLE	This is an internal logic problem.
CAN'T MOVE TO CORRUPTDIR	An attempt to move some bad C. or X. files to the */usr/spool/uucp/.Corrupt* directory failed. The directory is probably missing or has wrong modes or owner.
CAN'T CLOSE	A close( ) or fclose( ) call failed.
FILE EXISTS	The creation of a C. or D. file is attempted, but the file exists. This occurs when there is a problem with the sequence file access. Usually indicates a software error.

Error Message	Description/Action
`No uucp server`	A TCP/IP call is attempted, but there is no server for UUCP.
`BAD UID`	The uid cannot be found in the */etc/passwd* file. The filesystem is in trouble, or the */etc/passwd* file is inconsistent.
`BAD LOGIN_UID`	Same as previous.
`ULIMIT TOO SMALL`	The ulimit for the current user process is too small. File transfers may fail, so transfer is not attempted.
`BAD LINE`	There is a bad line in the **Devices** file; there are not enough arguments on one or more lines.
`FSTAT FAILED IN EWRDATA`	There is something wrong with the Ethernet media.
`SYSLST OVERFLOW`	An internal table in *gename.c* overflowed. A big or strange request was attempted.
`TOO MANY SAVED C FILES`	Same as previous.
`RETURN FROM fixline ioctl`	An ioctl, which should never fail, failed. There is a system driver problem.
`BAD SPEED`	A bad line speed appears in the **Devices/Systems** files (Class field).
`?ERMISSIONS file: BAD OPTION`	There is a bad line or option in the **Permissions** file.
`PKCGET READ`	The remote machine probably hung up. No action need be taken.
`PKXSTART`	The remote machine aborted in a non-recoverable way. This can generally be ignored.
`SYSTAT OPEN FAIL`	There is a problem with the modes of */usr/lib/uucp/.Status*, or there is a file with bad modes in the directory.

Error Message	Description/Action
TOO MANY LOCKS	There is an internal problem!
XMV ERROR	There is a problem with some file or directory. It is likely the spool directory, because the modes of the destinations were suppose to be checked before this process was attempted.
CAN'T FORK	An attempt to fork and exec failed. The current job should not be lost, but are attempted later (**uuxqt**). No action need be taken.

UUCP STATUS Error Messages

Status error messages are messages that are stored in the */usr/spool/uucp/.Status* directory. This directory contains a separate file for each remote machine that your system attempts to communicate with. These individual machine files contain status information on the attempted communication, whether it was successful or not. What follows is a list of the most common error messages that can appear in these files.

Error Message	Description/Action
OK	Things are OK.
NO DEVICES AVAILABLE	There is currently no device available for the call. Check to see that there is a valid device in the **Devices** file for the particular system. Check the **Systems** file for the device to be used to call the system.
WRONG TIME TO CALL	A call was placed to the system at a time other than what is specified in the **Systems** file.
TALKING	Self explanatory.
LOGIN FAILED	The login for the given machine failed. It could be a wrong login/password, wrong number, a very slow machine, or failure in getting through the *dialer-token* script.

Error Message	Description/Action
CONVERSATION FAILED	The conversation failed after successful startup. This usually means that one side went down, the program aborted, or the line (link) was dropped.
DIAL FAILED	The remote machine never answered. It could be a bad dialer or the wrong phone number.
BAD LOGIN/MACHINE COMBINATION	The machine called us with a login/machine name that does not agree with the **Permissions** file. This could be an attempt to masquerade!
DEVICE LOCKED	The calling device to be used is currently locked and in use by another process.
ASSERT ERROR	An ASSERT error occurred. Check the */usr/spool/uucp/.Admin/errors* file for the error message and refer to the section "ASSERT Error Messages."
SYSTEM NOT IN Systems	The system is not in the **Systems** file.
CAN'T ACCESS DEVICE	The device tried does not exist or the modes are wrong. Check the appropriate entries in the **Systems** and **Devices** files.
DEVICE FAILED	The open of the device failed.
WRONG MACHINE NAME	The called machine is reporting a different name than expected.
CALLBACK REQUIRED	The called machine requires that it calls your system.
REMOTE HAS A LCK FILE FOR ME	The remote site has a LCK file for your system. They could be trying to call your machine. If they have an older version of UUCP, the process that was talking to your machine may have failed leaving the LCK file. If they have the new version of UUCP, and they are not communicating with your system then the process that has a LCK file is hung.

Error Message	Description/Action
`REMOTE DOES NOT KNOW ME`	The remote machine does not have the node name of your system in its **Systems** file.
`REMOTE REJECT AFTER LOGIN`	The login used by your system to log in does not agree with what the remote machine was expecting.
`REMOTE REJECT, UNKNOWN MESSAGE`	The remote machine rejected the communication with your system for an unknown reason. The remote machine may not be running a standard version of UUCP.
`STARTUP FAILED`	Login succeeded, but initial handshake failed. Check communication parameters: data word size, parity, stop bits, etc.
`CALLER SCRIPT FAILED`	This is usually the same as `DIAL FAILED`. However, if it occurs often, suspect the caller script in the **Dialers** file. Use **uutry** to check.

Glossary

ALIAS. An alternate host name, created as a convenience for addressing a host on a local network whose unique primary name is long and/or complicated.

ARP. Address Resolution Protocol is used by Ethernet for address mapping.

ARPA. Now called DARPA, stands for Defense Advanced Research Projects Agency. ARPANET is the network based on the work sponsored by this agency. See also DDN.

BIND. Berkeley Internet Name Domain. Also: bind. To fix an association between a name and an object. In networking, used to explicitly assign a network address to a socket.

BRIDGE. A simplified gateway used to connect local networks that use the same internal protocols and exhibit the same interface to attach stations.

BROADCAST NETWORK. A system in which messages are sent to all hosts simultaneously, rather than from point to point. Each node then "grabs" the transmissions intended for them.

BSD. Berkeley Software Distribution.

BUS. A set of one or more parallel signals implemented in hardware in a standard manner so that multiple devices can access it and communicate over it.

CACHE. To store temporarily in memory to improve access performance. Also, that which is stored temporarily in memory.

CACHING-ONLY SERVER. A server that is not authoritative for any domain. This server services queries and asks other servers that have the authority for the information needed.

CCITT. The Comite Consultatif Internationale de Telegraphie et Telophonie. A communications organization that sets international usage standards. In English: International Telegraph and Telephone Consultative Committee. See X.25.

CLIENT. A computer or executing program that sends a request to a server, and waits for a response. The term "client" is generally used in the context of NFS.

CLONING DEVICE. A cloning device provides for dynamic allocation of resources by means of a single pathname.

CONNECTION. A connection is a logical communication path.

CONNECTIONLESS. A packet delivery system in which packets sent from one machine to another may follow different paths. It is called unreliable because delivery is not guaranteed. Packets may be delivered out of sequence, duplicated, or lost. However, connectionless delivery may be desirable due to its low transport overhead.

CONSUMER. A computer or executing program that receives and uses information. A subset of client. The term "consumer" is generally used in the context of LM/X.

DAEMON. A daemon is a system service. It is a program that is active in the background but not connected to a terminal. Also: demon.

DARPA. Department of Defense Advanced Research Project Agency, formerly called ARPA. This agency sponsored the network architecture research project upon which ARPANET is based. ARPANET is a large governmental internetwork, called the Internet, part of which is the Defense Data Network (DDN).

DATAGRAM. Basic transfer unit of IP. Consists of a header, containing Internet source and destination addresses, and data. Also called a packet. Datagram implies that delivery will be connectionless. Also: dgram.

DATA LINK LEVEL. Data link level is the communications protocol for the physical media-link used to transport the data.

DDN. Defense Data Network. A set of communications capabilities that link together computer systems within the Department of Defense (oD). The DDN allows users of these computer systems to send mail and files between systems and to access other computers on the network in interactive terminal sessions. The DDN is part of the DARPA Internet. See also Internet.

DESTINATION. The destination address, an internet header field.

DESTINATION ADDRESS. Network and host identifiers.

DNS. Domain Naming System.

DOMAIN. A naming category in DNS, a hierarchical naming scheme. A domain is a set of machines usually grouped by geographic location, organization, or activity (for example, EDU for educational machines, COM for machines in commercial use).

ETHERNET. Originally, a heavily shielded, half-inch diameter coaxial cable developed by Xerox Corporation, Digital Equipment Corporation, and Intel Corporation, for use in local area networks.

FLOW CONTROL. Flow control is the function and process of regulating the traffic and amount of data between flowing nodes so that neither node is sent more data than it can handle at a given time.

GATEWAY. A protocol translator device connecting two local networks, or a local to a long-haul network. Gateways can be thought of as communication paths for the exchange of data between networks.

HOST. A host is a computer that acts as client and/or server. It is, specifically, a source or destination of messages from the point of view of the communication network.

IAB. Internet Activities Board.

ICMP. Internet Control Message Protocol. CMP is used by a gateway or destination host to communicate with a source host, for example, to report an error in datagram processing. ICMP uses the basic support of IP as if ICMP were a higher level protocol. However, ICMP is actually an integral part of IP, and must be implemented by every IP module.

IEN. Internet Engineering Notes.

INTERNET. When capitalized, Internet refers specifically to the internet built by DARPA. Otherwise it refers to any internet.

INTERNET ADDRESS. A 32-bit universal identifier assigned to each host on the Internet.

INTERNETWORKING. The connection of networks using different hardware and/or software protocols by means of devices called gateways, for the purpose of forwarding data from one network to another. Internetworking allows several networks to function cooperatively as a single, virtual network.

LAYER. A conceptual model in protocol software in which each machine in a network can be thought of as being stacked in tiers, in which each tier, or level, handles one aspect of the process of transferring data.

LOOPBACK INTERFACE. Used for diagnostic purposes, loopback interface is software, without any associative hardware, that receives information and sends it right back to its point of origin.

MASTER SERVER. A master server is the authority for a particular domain and maintains all data corresponding to it.

NETWORK INTERFACE. Device drivers and associated hardware that allow TCP/IP software to communicate with a particular network.

NETWORK MASK. A bit mask that specifies the portion of an Internet address that is to be considered the network part for that network.

PORT. A port, or port number, is a 16-bit address used by TCP/IP to identify a socket on a particular machine.

PRIMARY MASTER SERVER. A server that loads its data from a file on disk. In a multiple master situation, this server may also delegate authority to other servers in its domain.

PROCESS. A process is a program in execution. A source or destination of data from the point of view of the Transmission Control Protocol (TCP), or other host-to-host protocol.

PROTOCOL. A set of rules for communications, including standards for message format.

RFC. Request For Comments. A document containing proposals, ideas, observations, as well as general information and accepted Internet protocol standards. RFC is usually followed by a number, which refers to a particular edition or iteration of the notes, and is available across the Internet.

ROOT. root is the login name of the super-user. The super-user is the user who has the widest form of machine privileges.

ROUTING TABLE. A collection of configuration information that allows for the dynamic and adaptive transfer of data from point to point, automatically, via the best available path.

SECONDARY MASTER SERVER. A server that is delegated authority and receives its data for a domain from a primary master server. A secondary master server functions as a master server or backup when the primary master server is unavailable.

SERVER. Any program that accepts requests over the network, performs a service, and returns the result to the machine making the request.

SLAVE SERVER. A server that always forwards queries it cannot satisfy locally to a fixed list of forwarding servers. In slave mode the server forwards each query to each of the forwarders until an answer is found or the list of forwarders is exhausted.

SOCKET. A socket provides a point of access to network software that allows use of the network.

TCP. Transmission Control Protocol is a transport level, connection-oriented protocol that provides reliable end-to-end message transmission over an internetwork.

UDP. User Datagram Protocol. A connectionless mode, user-level transport protocol for transaction-oriented applications. UDP datagrams include a protocol port number, enabling the sender to specify a particular application on the remote machine.

X.25. X.25 is a circuit-switched network protocol used commonly in Europe and less so in the United States. X.25 is based on a three-layer, peer-communications protocol standard defined by the International Telegraph and Telephone Consultative Committee (CCITT).

Administering ODT-DOS

Contents

Chapter 1 Introduction 1
Who Should Use This Guide 1
Organization of This Guide 1
ODT-DOS Guides 2
Installing ODT-DOS 2
Release Notes 2

Chapter 2 Administering ODT-DOS 3
Using the dosadmin Program 4
Adding And Deleting User Accounts 4
Administering DOS Applications 4
Administering the System Console 6
Administering COM Ports 11
Administering DOS Printers 11
Backing Up the ODT-DOS Filesystem 15
Administering Disk and Diskette Drives 15
Administering the Physical DOS Partition 16
Administering Virtual DOS Partitions and Virtual Floppy Disks 19
Installing Plug-In Cards in Your Computer 25
Making New DOS Images 31
System Files Affected by System Administration 35

Chapter 3 Installing DOS Applications 37
Installing DOS Applications Using dosadmin 37
Installing Copy-Protected DOS Applications 50
Removing DOS Applications 54

OPEN DESKTOP™ Software

OPEN DESKTOP Administration and User Documentation

Processed: Wed Dec 20 18:09:01 PST 1989
Document number : 12/21/89 1.0.0C

ODT-DOS is based on technology developed for Merge 386 by Locus Computing Corporation.

12/21/89-1.0.0D
Processed: Wed Dec 20 11:42:11 PST 1989

Chapter 1

Introduction

This guide explains how to administer ODT-DOS. Administering ODT-DOS is no different in most respects from administering separate, conventional DOS and UNIX systems. The administrator's responsibilities include installing and maintaining system hardware and software, regularly backing up system data, assisting users, and informing them of changes to the system.

This guide supplements the system administration instructions in your DOS and UNIX documentation. You should be familiar with that documentation because this guide is not a comprehensive description of the system administrator's responsibilities. In general, you administer ODT-DOS by using UNIX procedures to accomplish UNIX tasks and DOS procedures to accomplish DOS tasks.

Who Should Use This Guide

This guide is for the Open Desktop system administrator, the person responsible for maintaining the day-to-day operation of the system. This guide covers only the administrative procedures that are necessary to manage the combined DOS and UNIX environment of Open Desktop. It supplements the documentation on your computer hardware, DOS, and the UNIX System.

Organization of This Guide

This guide has two additional chapters:

Chapter 2, Administering ODT-DOS, tells you how to manage user accounts, set up and administer computer hardware used by DOS, and configure your computer's resources to meet the combined needs of DOS and UNIX System users.

Chapter 3, Installing DOS Applications, provides hints for installing DOS applications for personal or public use, installing copy-protected applications, setting up DOS applications for use from the UNIX shell, and removing DOS applications.

ODT-DOS Guides

Other guides that describe ODT-DOS operation and administration include:

- *Using ODT-DOS* in the *Open Desktop™ User's Guide.*
- The optional Open Desktop documentation.

Installing ODT-DOS

This guide assumes that you have installed ODT-DOS according to the instructions in the *Open Desktop Installation Guide.*

Release Notes

Be sure to read the *Open Desktop Release Notes* for up-to-date information on supported hardware and software, as well as information on product changes since this guide was printed.

Chapter 2
Administering ODT-DOS

This chapter covers the topics that are essential for using your computer as a combined DOS and UNIX machine. The topics include:

- using the **dosadmin** program,
- adding and deleting user accounts,
- administering DOS applications
- administering the system console,
- administering COM ports,
- administering DOS printers,
- backing up the ODT-DOS filesystem,
- administering disk and diskette drives,
- administering the physical DOS partition,
- administering virtual DOS partitions and floppy disks,
- installing plug-in cards,
- making new DOS images, and
- system files affected by system administration.

To administer ODT-DOS effectively, you should be familiar with the contents of *Using ODT-DOS* in the *Open Desktop User's Guide*, in addition to this chapter.

Most of the descriptions in this chapter assume that you are logged in as root or you are the super user. The UNIX # prompt is therefore shown in most examples. Examples that apply to any user are shown with appropriate DOS or UNIX ($) prompts.

Using the dosadmin Program

The **dosadmin** menu system provides an easy way to change the values of three important DOS characteristics: memory, DOS startup files, and DOS device files. Throughout this guide you will use the **dosadmin** menu to perform various system administration tasks.

Adding and Deleting User Accounts

ODT-DOS requires no special procedures for adding or deleting user accounts. Any user with a valid UNIX account can log into ODT-DOS. You are not required to be logged in as root or have any special permissions, for example, to run DOS or use DOS and UNIX commands.

In general, follow the instructions in the *Administering ODT-OS* in the *Administrator's Guide* for adding, deleting, and administering user accounts.

Although no special configuration files or setup procedures are required to use DOS, Open Desktop users may want to customize the way DOS runs. Users who follow the instructions in *Using ODT-DOS* can alter the behavior of DOS in many ways without help from the system administrator. If users at your site are unfamiliar with the DOS or UNIX systems, however, they may require your assistance. *Using ODT-DOS* contains hints for system administrators who want to modify ODT-DOS defaults or help users configure their own individual environments.

Administering DOS Applications

Occasionally you may run into problems with DOS applications running exactly as they were intended under ODT-DOS. This section covers methods that you can use to fix these problems.

Keyboard Buffer and DOS Applications

Some DOS applications that buffer keystrokes may not work as expected in the ODT-DOS environment. These applications include applications, such as SuperKey, that create keyboard macros by mapping multiple keystroke to a single key.

You can correct this problem by putting the line:

```
device=\usr\dbin\ansi.sys
```

in a CONFIG.SYS file that is interpreted when you run DOS. This approach has a side effect, however. To improve system response, ODT-DOS, by default, puts DOS applications that poll the keyboard to sleep while they wait for keyboard input. When you include the *device=\usr\dbin\ansi.sys* line in your CONFIG.SYS file, ODT-DOS no longer puts applications that poll the keyboard to sleep. Your computer is therefore likely to be more heavily loaded, especially if you run multiple DOS applications at once.

Applications that Poll the Keyboard

DOS applications that poll the keyboard can consume system resources even when they are idle by entering a polling loop. If not compensated for, these applications reduce system performance because when they poll the keyboard, less CPU time is available to other concurrently running processes. By default, ODT-DOS corrects this problem by putting many applications that enter a polling loop to sleep until there is keyboard input.

This method works poorly unless you disable the **pollsleep** feature. To disable **pollsleep**, type at the DOS prompt:

merge set pollsleep off

To restore the default condition, type:

merge set pollsleep omerge set pollsleep on

Network Applications and DOS Drives

Some DOS applications, especially applications designed to work in a network environment, may not recognize ODT-DOS drives that access the shared DOS/UNIX file system (that is, drives C:, D:, or J:) as valid DOS drives.

By default, ODT-DOS treats these drives and files that reside on them as "remote" when an application queries the operating system for information about local and remote drives or files. If a DOS application expects these ODT-DOS drives or files to be "local," the application may fail. If you encounter such a failure, try using the **merge set drive local** or **merge set handle local** command.

The **merge set drive local** command causes ODT-DOS to treat drives that access the shared DOS/UNIX file system as "local." To use this command, start a DOS environment and type the following at your DOS prompt:

```
merge set drive local
```

The command is effective for the duration of the DOS environment. If you want to switch back to the default treatment of drives without exiting the DOS environment, type:

```
merge set drive remote
```

The **merge set handle local** and **merge set handle remote** commands work like the **merge set drive local** and **merge set drive remote** commands, but they cause DOS to interpret files rather than drives as either local or remote. Use the **merge set drive** and **merge set handle** commands individually or in combination according to the requirements of your DOS applications.

Administering the System Console

ODT-DOS works with any system console that consists of a monochrome or color monitor with a PC-compatible keyboard, connected to a monochrome, Hercules, CGA, EGA, or VGA display adapter. *Using ODT-DOS* describes how to use the console from the point of view of the ODT-DOS user. This section tells you how to connect and configure the console. The topics covered here are:

- Setting Up the Console
- Changing Console Display Adapter Cards
- Using Extended Video Modes
- Using the UNIX MultiScreen™ Facility with DOS
- Redefining the ODT-DOS Switch-Screen Key Sequence

Setting Up the Console

No special procedures are required to set up the system console for use with ODT-DOS. The only requirement for using ODT-DOS is that you must use one of the display adapters that ODT-DOS recognizes. These include monochrome, Hercules, CGA, EGA, and VGA adapters. Refer to the *Open Desktop Release Notes* for a specific list of supported display adapters.

ODT-DOS and Display Modes

When you use ODT-DOS, you can run DOS applications on ASCII or PC scancode terminals or on a console that uses a monochrome, Hercules, CGA, EGA, or VGA display adapter and a compatible display. In addition, by using different DOS *images*, ODT-DOS can simulate the characteristics of display adapters less powerful than your actual physical display adapter. (Refer to "Making New DOS Images" later in this chapter for more information on DOS images.)

This flexibility can lead to unexpected problems if you are not careful how you install, configure, and run graphics applications. For example, an application may fail to run if you have configured it to use a different type of display than you really use. If your system uses more than one type of display, you may find that some applications run correctly on some displays but not on others.

Some applications automatically detect the type of display adapter being used and adjust themselves to display graphics data correctly. Other applications do not work properly unless you identify your display adapter when you install them. When you install these applications, be sure you follow the application manufacturer's instructions and configure the applications to work with the display adapter you intend to use.

If you run DOS processes from an ASCII terminal, identify your display as a monochrome display adapter (MDA). If you use a graphics display adapter, you should normally configure your DOS applications to use the type of adapter in your computer, provided the application is compatible with that type of adapter. For example, if you use a VGA card, configure your DOS applications to use a VGA card.

In rare cases, you may run into difficulty because you are trying to use an incompatible combination of physical display adapter, DOS image, and application. By default, ODT-DOS uses DOS images that correspond to your physical display — VGA images for VGA displays, CGA images for CGA displays, and so on. Most DOS applications that are configured for less powerful display adapters run correctly, automatically, when you run them on more powerful adapters. For example, an application configured for CGA displays

may run correctly on a VGA display. However, some applications fail when run on a more powerful display than they are configured for. If you run into this restriction, you can work around it in either of two ways:

- Reconfigure the application so it expects to be run on the display adapter you are using. For example, if your application expects to be run on a CGA card and you use a VGA card, reconfigure the application to expect a VGA card.
- Use a DOS image that corresponds to the display adapter the application expects to use.

For example, if you have a VGA card but your application expects to use a CGA card, you can start a DOS environment with the command:

dos +acga

Then start your application. If you want to start the application directly from the UNIX shell, you can also use a command such as:

dos +acga *appl*

where *appl* is the command that starts your application. These commands cause ODT-DOS to use a CGA image instead of the default image, and your application views your computer hardware as though you actually have a CGA card.

Note that this technique works only if your physical display is capable of using the DOS image you request. The following table shows usable combinations of physical display adapters and DOS images. Fields with a dash (-) indicate unusable combinations.

	Capability By Physical Display Type				
Option	**Monochrome**	**Hercules**	**CGA**	**EGA**	**VGA**
None	mono	herc	CGA	EGA	VGA
+amono	mono	mono	mono	mono	mono
+aherc	-	herc	-	-	-
+acga	-	-	CGA	CGA	CGA
+aega	-	-	-	EGA	-
+arega	-	-	-	EGA	-
+avga	-	-	-	-	VGA

Refer to *Using ODT-DOS* for more information on these options and the characteristics of graphics displays.

Changing Console Display Adapter Cards

If you change your console display adapter card after you install ODT-DOS, you may need to make a new DOS image. See "Making New DOS Images," later in this chapter, for further information.

Using Extended Video Modes

Whether or not you can use extended video modes on EGA and VGA cards depends on how your DOS images were created. If you have an EGA or VGA display and want to run high-resolution graphics applications, you need to create an EGA or VGA DOS image. You can run DOS applications that require monochrome or CGA displays using your EGA or VGA card even if you do not make an EGA/VGA image.

To make an EGA/VGA image, your computer must have a 100 percent IBM-compatible EGA or VGA card. If your card is not fully IBM-compatible, the procedure for make an EGA or VGA image may fail. When the procedure fails, symptoms range from an error message stating that an image cannot be created to locking up the system. Refer to the lsit of supported display adapters in the *Open Desktop Release Notes* for information on tested and certified EGA and VGA cards.

For information on how to create a new DOS image, refer to "Making New DOS Images," later in this chapter.

Using the UNIX MultiScreen Facility with DOS

ODT-DOS is compatible with the standard UNIX MultiScreen facility. If your console can display multiple, independent screens, you can run DOS commands and applications or a DOS environment in any screen.

When you run multiple DOS processes in separate screens, the default key sequence for switching between screens is **Alt-F***n*, where F*n* is a function key (such as F1 or F2). **Alt-F2** switches to screen 2, **Alt-F3** switches to screen 3, and so on. The UNIX system also uses the **Alt-F***n* convention for switching between screens, so you can switch between any active DOS and UNIX screens using **Alt-F***n*.

Redefining the ODT-DOS Switch-Screen Key Sequence

You can redefine the switch-screen key sequence that applies to DOS screens if **Alt-F***n* does not suit your needs. For example, you may have a DOS application that does not work properly with the default switch-screen sequence because it assigns **Alt-F***n* a special meaning. You can redefine the switch-screen key sequence to use a function key together with any combination of the **Ctrl**, **Alt**, and **Shift** keys, or to use only a function key. To redefine the switch-screen key sequence, use the **switchkey** command with the syntax:

switchkey [-cas]

In this syntax, **c** stands for the **Ctrl** key, **a** stands for the **Alt** key, and **s** stands for the **Shift** key. To switch screens any time after you use the **switchkey** command, use the keys you specified with **switchkey** in addition to a function key. For example, to specify that you want to require **Ctrl** and **Shift** along with a function key, type:

$ switchkey -cs

Thereafter, to switch screens, press **Ctrl-Shift-F***n*.

To specify that you want to use only function keys, without **Ctrl**, **Alt**, or **Shift**, use **switchkey** with only a hyphen as an argument:

$ switchkey -

Thereafter you can use F1, F2, F3, and so on, without additional keys, to switch from your current DOS screen to a different one. The **switchkey** command with no arguments displays the current switch-screen key sequence.

The switch-screen key sequence you define with **switchkey** applies only when you are viewing a DOS screen. When you are viewing a UNIX screen, the existing UNIX switch-screen sequence is effective.

ODT-DOS and the X Window System both use the same default switch-screen key sequence and are affected the same way when you use **switchkey**.

Any switch-screen key sequence that you define with **switchkey** applies to any DOS or X Window sessions run on the console where you ran **switchkey**. DOS and X Window sessions that are currently running when you redefine the switch-screen key sequence are also affected. Your specified key sequence remains effective until you issue another **switchkey** command or reboot your computer. To make sure your preferred key sequence remains effective even after a reboot, include the **switchkey** command in */etc/profile* or your home directory *.profile*.

Administering COM Ports

On Open Desktop, both the UNIX environment and DOS can use COM (serial) ports, but only one process (either UNIX or DOS) can access a particular COM port at one time. If you plan to use a COM port for DOS work, follow these procedures:

1. Make sure no **getty** is running on the physical device (for example, */dev/tty1a* or */dev/tty2a*) that you want to make available to DOS as a COM port. To do this, log in as root or become the super user and run the **disable** command. For example, to disable the **getty** process on */dev/tty2a*, type:

   ```
   # disable /dev/tty2a
   ```

2. Use the **chmod** command to set permissions of the device to be readable and writable. If you want to use */dev/tty2a*, you would type:

   ```
   # chmod 666 /dev/tty2a
   ```

The COM port is now ready for use with DOS. To return the COM port for use with UNIX, type:

```
# enable /dev/tty2a
```

For more information on using COM ports with DOS, refer to *Using ODT-DOS*

Administering DOS Printers

Using ODT-DOS describes, from the user's point of view, how to print using DOS commands or applications. This section explains how to set up and administer DOS printers. It covers the following topics:

- configuring the default DOS printer,
- changing the default DOS printer,
- adding DOS printers, and
- administering printers directly attached to DOS.

Configuring the Default DOS Printer

By default, ODT-DOS sends all DOS printing via the UNIX spooler to a printer named **doslp**. The ODT-DOS installation routine automatically attempts to configure your default UNIX printer, if one exists, so it can also be used as the default DOS printer, **doslp**. To determine whether **doslp** has been configured, type the following **lpstat** command:

```
# lpstat -p doslp
```

This command tells you whether or not **doslp** is properly configured. If it is not, you need to configure **doslp** yourself.

To configure **doslp**, follow the instructions for configuring printers in the *Administering ODT-OS*. You need to know the UNIX device name of the printer you want to use as **doslp** (for example */dev/lp0*). Use standard UNIX procedures to perform the following operations:

1. Assign the printer name **doslp** to your preferred printing device.

2. Choose the appropriate printer interface program as described below. (The interface program is also known as a printer model.)

Selecting a Printer Interface Program

ODT-DOS supplies a printer interface program named **dosmodel**. It is the same as the **standard** model, except that it does not print banner pages at the beginning of each print job, and it does not output a form feed at the end of each print job.

If ODT-DOS automatically configured **doslp** during the installation procedure, it used **dosmodel**. Any printer that recognizes the **standard** model can also use **dosmodel** for printing DOS output or any other output that does not require banner pages or form feeds.

You can choose the **standard** model for **doslp** if you prefer to have banner pages and automatic form feeds.

If your printer does not recognize the **standard** model, you should not use either the **standard** model or **dosmodel**. Instead, choose a model that is appropriate for your printer.

Changing the Default DOS Printer

The default DOS printer is always named **doslp**. To change the physical printer that **doslp** refers to, follow the standard UNIX procedures for configuring printers. Assign the name **doslp** to your preferred printing device and choose an appropriate interface program.

Adding DOS Printers

You can use printers other than **doslp** for DOS printing. You can select any convenient printer name and any appropriate interface program when you configure a printer.

To use any available UNIX printer for DOS printing, use the ODT-DOS **printer** command to correlate a print stream with a particular UNIX printer and print command. The syntax for selecting a print stream, a printer, and a UNIX print command is:

printer [*print_stream*] **unix** "*print_command*"

where *print_stream* is LPT1, LPT2, or LPT3 and *print_command* is a UNIX command that processes the specified print stream. If you do not specify a print stream, **printer** assumes LPT1 by default. Use the **printer** command in the DOS environment. Assume, for example, that you want to direct DOS printer output sent to LPT2 to a UNIX printer named "laser." Follow these steps:

1. Start a DOS environment if you haven't already started one.
2. Use the following **printer** command to direct print stream LPT2 to to the printer named "laser":

   C> **printer lpt2 unix "lp -dlaser"**

   In this command, the **-d** ("destination") option to the **lp** command identifies the printer named "laser."
3. To send DOS printer output to the printer, name the DOS print stream in your DOS print command.
   For example:

   C> **copy letter.txt lpt2**

You can direct printer output from DOS applications to any UNIX printer using the same procedures.

If you want a **printer** command to be effective every time you use the DOS environment, you can include it in an AUTOEXEC.BAT file.

Note that the *print_command* that you specify when you use the **printer** command is typically lp -d*printer*, where *printer* is the name of a UNIX printer. However, *print_command* can be *any* UNIX command that you choose to use to process a print stream.

Administering Printers Directly Attached to DOS

Under some circumstances, Open Desktop users may not want to use the UNIX spooling system when they use DOS printing. In these cases, users can attach a printer directly to a DOS process. The printer is then under the direct control of DOS, and not available for UNIX printing. Appendix C in *Using ODT-DOS* explains the procedures that users must follow to attach a printer directly to a DOS process.

If a user wants to attach a printer directly that is currently configured for UNIX printing, you need to disable UNIX printing on that printer. Follow these procedures:

1. Log in as root or become the super user.

2. Disable UNIX printing by using the **disable** command. For example, if you want to disable UNIX printing on the printer named **doslp**, issue the command:

   ```
   # disable doslp
   ```

   The printer is now available for direct attachment to any user's DOS process. Refer to Appendix C in *Using ODT-DOS* for instructions on directly attaching a printer to DOS.

   While UNIX printing is disabled, users can continue to spool print jobs to that printer. However, the jobs are not printed until the printer is re-enabled for UNIX use.

3. When the user finishes the printing that requires directly attaching the printer to DOS, you can use the **enable** command to re-enable UNIX printing. For example, to re-enable the printer **doslp**, issue the command:

   ```
   # enable doslp
   ```

   NOTE: A single physical printer may have more than one printer name associated with it. If the UNIX printer you are disabling has more than one UNIX printer name, then, in step 2, you should issue the **disable** *printer_name* command for each printer name correlated with that printer. Similarly, in step 3 of this procedure, you may need to issue multiple **enable** commands.

Backing Up the ODT-DOS Filesystem

To guard against permanently losing important data, you should regularly back up all data on your fixed disk. ODT-DOS requires no special procedures for system backups. Simply use UNIX backup procedures for the shared DOS/UNIX filesystem and DOS backup procedures for the physical DOS partition (drive E:).

Consult your DOS and UNIX documentation for specific procedures recommended for your system.

NOTE: If they are stored on a physical or virtual DOS partition, some copy-protected DOS applications cannot be backed up and restored using the BACKUP and RESTORE commands. For more information, consult your DOS user's manual and the instructions for your applications.

Administering Disk and Diskette Drives

Your ODT-DOS system may have one or more diskette drives that can be assigned to either the DOS or the UNIX environment. A diskette drive mounted as a UNIX device is assigned to the UNIX environment and is not accessible as a DOS diskette drive. If you insert a diskette containing a UNIX filesystem into a diskette drive and mount it as a UNIX device, any files on that diskette become part of the shared DOS/UNIX filesystem. You can then use DOS drive C: or J: to access files on the mounted diskette in the same way you access files in the shared DOS/UNIX filesystem on the fixed disk.

To be directly accessible to DOS as DOS drives A: or B:, diskette drives must not be mounted as UNIX devices or in use by UNIX programs (such as **cpio**). If you have more than one diskette drive on your system, you must not have any of them mounted as UNIX devices or otherwise in use by the UNIX system when you attempt to access a diskette drive as DOS drive A: or B:. ODT-DOS prevents a DOS process from using all diskette drives as long as any other process is using any diskette drive.

Sharing Diskette Drives

When diskette drives are not mounted as UNIX devices or otherwise used by the UNIX system, DOS users can access them on a first-come, first-served basis. Whenever a diskette drive has not been used for five seconds or more, it is available to the first DOS process that accesses that drive.

Administering the Physical DOS Partition

The physical DOS partition is a portion of the fixed disk formatted under DOS and reserved exclusively for DOS files. You use the DOS partition under ODT-DOS as DOS drive E:. A physical DOS partition is useful on Open Desktop for the following reasons:

- Some copy-protected DOS applications that cannot be installed in the shared DOS/UNIX filesystem can be successfully installed in the DOS partition.
- You can use files and applications contained in the DOS partition under "raw" DOS, that is, by shutting down the UNIX System and booting DOS.

If you do not already have a physical DOS partition and you choose not to create one, you can still use all ODT-DOS features except drive E:. You can also create and use a virtual DOS partition any time after you install ODT-DOS without removing or reinstalling any files that are currently on your fixed disk. Virtual DOS partitions offer advantages similar to those of a physical DOS partition. Refer to "Administering Virtual DOS Partitions and Virtual Floppy Disks," later in this chapter, for more information.

This section tells you how to create, format, and administer the physical DOS partition.

Creating and Formatting the Physical DOS Partition

If you want to have a physical DOS partition on your Open Desktop computer, you should create one at the time you install the UNIX system, before you install ODT-DOS. Use the utilities packaged with your computer hardware to create and format a DOS partition. Your DOS partition should have a minimum size of 2.5 megabytes. Some DOS copy-protection schemes will not install on a partition smaller than 2.5 megabytes. If you have already installed ODT-DOS, do not have a physical DOS partition, and wish to create one, you may have to back up your fixed disk, create the DOS partition, and reinstall your UNIX system and all applications. Refer to the documentation packaged with your computer for further information.

Default Protection of the DOS Partition

The UNIX filesystem protection mechanisms apply in only a limited way to the DOS partition. Because it is not a UNIX filesystem, access to individual DOS files in the partition is not governed by UNIX user or group ownership or by UNIX read, write, or execute permissions.

However, the DOS partition itself is accessed as a UNIX file or device and can be protected like any other UNIX file or device.

Following are the considerations that affect access to and administration of the DOS partition:

- Although multiple users can read files on the DOS partition at the same time, only one DOS process can write to the DOS partition at one time. In addition, when any DOS process writes to the DOS partition, no other DOS processes can read files on the partition until the process that is writing exits.

- The DOS partition contains a DOS filesystem and is located on a portion of the fixed disk that is physically distinct from the shared DOS/UNIX filesystem.

- The DOS partition is accessed as a UNIX device (sometimes also called a special file). The device name is */dev/dsk/dos*. This device appears to the UNIX system as a single UNIX file, but it can contain within it any number of DOS files and directories.

- When you install ODT-DOS, the DOS partition (that is, the special file */dev/dsk/dos*) is owned by root and is readable and writable by everyone. You can check the ownership and permissions of the DOS partition by typing:

  ```
  # ls -l /dev/dsk/dos
  ```

  The system then shows root as the owner and displays the following permissions:

  ```
  brw-rw-rw-
  ```

  The initial "b" identifies **/dev/dsk/dos** as a block-type special file (as opposed to character-type). UNIX execute permission for **/dev/dsk/dos** is not necessary.

Although it is owned by root, the DOS partition by default is considered to be a public resource that any user can access. All users have read permission so they can inspect the contents of the partition or execute DOS programs stored there. All users have write permission so they can install DOS applications on the partition. Write permission is granted to all users for an additional reason: even if users do not need to create files within the partition, some DOS applications require write permission when they run and will not work if run from an unwritable filesystem.

Changing the Protection of the DOS Partition

If you need to restrict access to the DOS partition, you can use one or both of the following methods:

- Use the DOS ATTRIB command to make specific files in the DOS partition read-only (that is, remove DOS write permission).
- Change the UNIX permission mode of */dev/dsk/dos* to restrict access to the partition.

Using the DOS ATTRIB command to make a DOS file read-only reduces the risk of that file being accidentally removed or corrupted. The protection offered by DOS ATTRIB is limited, however, because anyone who can access a DOS file can also use ATTRIB to make the file writable.

To make a DOS file read-only, use the ATTRIB command to assign the read-only (**R**) attribute as follows:

```
E> attrib +r filename
```

When the read-only attribute is assigned, the file cannot be deleted or changed. To make the file writable again, type:

```
E> attrib -r filename
```

If you want to restrict access to the DOS partition in a more general way, you can use the UNIX command **chmod** to assign any desired permission mode to */dev/dsk/dos*. For example, to deny access to users outside the group the partition belongs to, you could type:

```
# chmod o-rw /dev/dsk/dos
```

Administering Virtual DOS Partitions and Virtual Floppy Disks

The physical DOS partition is a section of the fixed disk that is formatted under DOS and reserved for DOS files. ODT-DOS also allows you to create virtual DOS partitions, which are UNIX files formatted as DOS volumes and used to store DOS files. You can use a virtual partition like a physical DOS partition to install copy-protected DOS applications that cannot be installed on the shared DOS/UNIX filesystem. Virtual partitions have the following additional advantages:

- You can create a virtual DOS partition easily at any time after you install ODT-DOS. You can therefore avoid the time-consuming process of reconfiguring your fixed disk to create a physical DOS partition if your system does not already have one.

- Users can create and own virtual partitions. By using personal virtual partitions, they can avoid ownership and permission mode conflicts that may arise when all users share the physical DOS partition.

Virtual DOS partitions differ from physical DOS partitions in the following ways:

- Virtual partitions are actually part of the UNIX filesystem. They are UNIX files that contain DOS filesystems. Because virtual partitions are really UNIX files, their contents can be backed up or restored along with other UNIX files using standard UNIX backup commands.

- You cannot access a virtual partition when you have shut ODT-DOS down and booted standard DOS.

After you create a virtual partition, you can attach it as a DOS drive (such as E:).

ODT-DOS also allows you to create a UNIX file and format it under DOS for use as a virtual DOS floppy drive. You can use virtual floppies much as you would use physical DOS diskette drive A: or B:.

The following sections show how to create and administer virtual DOS partitions IX "DOS" "partition" "virtual" and how to create virtual floppies.

Using dosadmin to Create a Virtual DOS Partition

To create a virtual DOS partition, use the following **dosadmin** procedure:

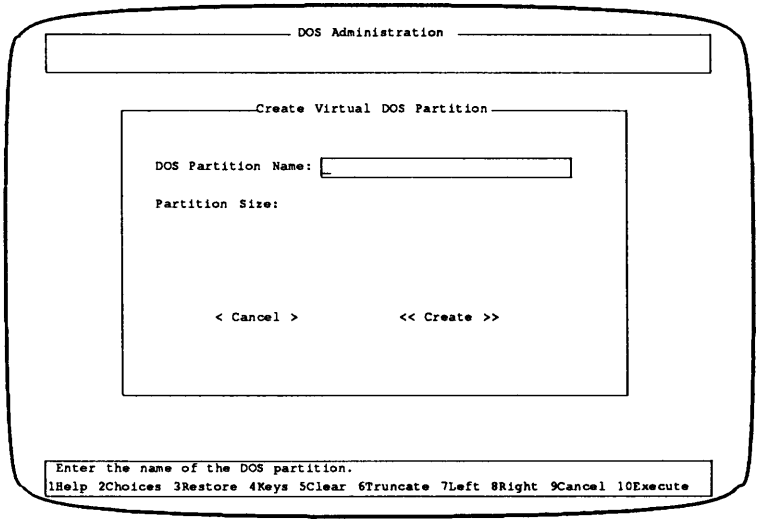
1. Start **dosadmin** by typing:

   $ dosadmin

   (You may start **dosadmin** from either the DOS or UNIX command line.) The **dosadmin** "DOS Administration" main menu appears.

2. Press the **Left** or **Right Arrow** key (← or →) to move the highlighted field to "Merge".

3. Press the **Up** or **Down Arrow** key (↑ or ↓) to move the highlighted field to the "Create Virtual Partition" field. Press **Enter** (↵) to select this item and display the following screen:



4. In the "DOS Partition Name" field, type the full path of the file you want to use as your virtual DOS partition. You supply the pathname using either DOS style (including the DOS drive name and using backslashes as path separators) or UNIX style. For example, to create a virtual DOS partition named */usr/joe/vpart*, type:

   ```
   /usr/joe/vpart
   ```

   Then press **Tab** (⇆) to move the highlight to the "Partition Size" field.

5. Enter the size of the virtual partition, in kilobytes (1024 kilobytes equals 1 megabyte; the recommended minimum size is 2.5 megabytes). For example, to create a 2.5-megabyte virtual DOS partition, type:

   ```
   2560
   ```

   Then press **Tab** to move to the "< Cancel >" field.

6. If you choose to cancel the operation and not create the virtual partition, press **Enter** while "< Cancel >" is highlighted. Otherwise, check your entries for accuracy. If any information on the screen is incorrect, press **Tab** to move to the field you want to change and retype the entry. (Press **F5** to clear a field completely.)

   When the information is correct, press **Tab** to highlight the "<< Create >>" field and then press **Enter**. ODT-DOS creates the virtual DOS partition you have selected.

Using Virtual Partitions

To use a virtual DOS partition, you must attach it as a drive between E: and Z: when you start DOS. For example, to use the */usr/joe/vpart* DOS partition as drive F:, you would start the DOS environment by typing:

```
$ dos +af:=/usr/joe/vpart
```

For the duration of the DOS environment, */usr/joe/vpart* is accessible as DOS drive F:. You can use the **dosopt** command to automatically attach a DOS partition whenever you run DOS. Refer to the descriptions of the **dosopt** command and the **±a** option in Appendix A of *Using ODT-DOS* for further information.

Note that the DOS drive name for a physical DOS partition, if you have one, is "E:". Only one partition at a time is available as drive E:. If you have a physical DOS partition, it is available by default as drive E: when you start a DOS process. If you want to use a virtual partition as drive E:, you can use the command **dos +ae:=***pathname* whether or not you have a physical DOS partition.

You should not use drive J: to attach a virtual partition since drive J: has a specific function in Open Desktop. Drive J: works exactly like drive C:, but you can have different current directories on the two drives. You can also use drive J: to simplify the use of some older DOS applications that require both the application and data or text files to be in the current directory.

Also, when specifying a drive name, be sure you redefine LASTDRIVE in your CONFIG.SYS file if you use a drive letter late in the alphabet. The default LASTDRIVE is "N".

Administering Virtual DOS Partitions

Most of the information covered under "Administering the Physical DOS Partition" in this chapter also applies to virtual DOS partitions. Virtual DOS partitions differ from physical DOS partitions in the following ways:

- A virtual DOS partition is a single file within the shared DOS/UNIX filesystem. It contains a DOS filesystem within it. It is not useful from the UNIX environment. You can use standard UNIX tools to move it around, back it up, or delete it. Deleting it frees up space on the shared DOS/UNIX filesystem.

- When you specify the size of a virtual partition at the time you create it, the partition does not immediately consume the disk space you specify. As you add files, the partition expands to consume the disk space required by the files up to the maximum amount you specified when you created the partition.

  Once the space is used, however, you cannot free space in the UNIX filesystem by removing files from the partition. Furthermore, if you back up and restore the partition using UNIX utilities, the partition always consumes the full amount of UNIX filesystem space specified when you created it, whether or not you have filled it with DOS files.

Using dosadmin to Create a Virtual DOS Floppy

Virtual floppy disks are UNIX files that have been formatted under DOS and contain DOS volumes in sizes that correspond to standard DOS diskettes. You may use them just as you would physical diskettes to store DOS files. You may even transfer DOS system files to a virtual floppy (using the DOS SYS command) and boot from it.

NOTE: When you boot from a virtual floppy, you normally cannot access the shared DOS/UNIX filesystem.

To create a virtual DOS floppy, use the following procedure:

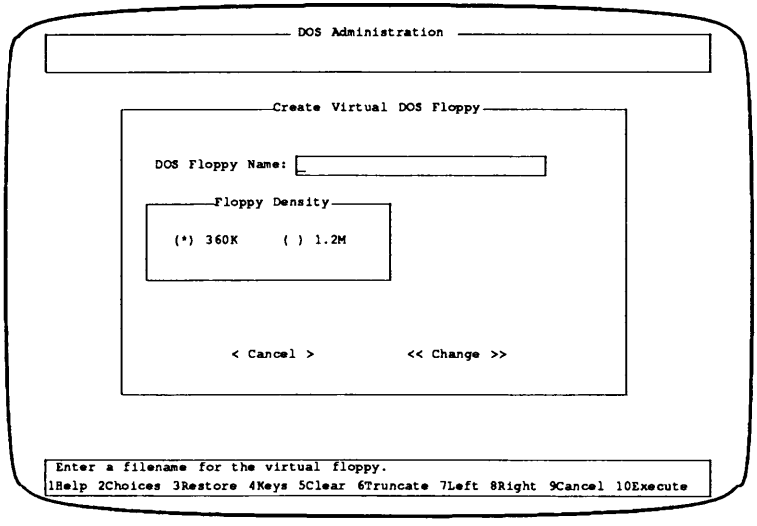
1. Start **dosadmin** by typing:

   $ dosadmin

   (You may start **dosadmin** from either the DOS or UNIX command line.) The **dosadmin** "DOS Administration" main menu appears.

2. Press the **Left** or **Right Arrow** key (← or →) to move the highlighted field to "Merge".

3. Press the **Up** or **Down Arrow** key (↑ or ↓) to move the highlighted field to the "Create Virtual Floppy" field. Press **Enter** (↵) to select this item and display the following screen:



4. Enter the full pathname of the file you want to use for your virtual DOS floppy in the "DOS Floppy Name" field. You supply the pathname using either DOS style (including the DOS drive name and using backslashes as path separators) or UNIX style. For example, to create a virtual DOS floppy named */usr/joe/vflop*, type:

   ```
   /usr/joe/vflop
   ```

   Then press **Tab** (⇆) to move to the "Floppy Density" field.

5. The field for low-density virtual floppies should be highlighted. This is the default for virtual floppies. To create a low-density floppy, press **Tab** (⇆) to move to the next field.

   If you want to create a high-density floppy, press the **Left** or **Right Arrow** key (← or →) to highlight the high-density value. Press **Enter** or the **Space** to select this option; an asterisk (*) appears between the parentheses. Then press **Tab** (⇆) to move to the next field.

6. If you decide to cancel the operation and not create a virtual floppy, press **Enter** while the "< Cancel >" field is highlighted. Otherwise, check your entries for accuracy. If any information on the screen is incorrect, press **Tab** to move to the field you want to change and retype your entry. (Use **F5** to clear a field completely.)

   When the information is correct, press **Tab** to highlight the "<< Create >>" field, and then press **Enter**.

ODT-DOS creates and formats the virtual floppy as you have specified. To use the virtual floppy, you must attach it as floppy drive A: or B: when starting DOS. For example, to use the **/usr/joe/vflop** virtual floppy as drive B:, you would start the DOS environment by typing:

```
$ dos +ab:=/usr/joe/vflop
```

When you attach a virtual floppy like this, you cannot access a physical DOS diskette drive with the same name (drive B: in this example). You can, however, access physical diskette drives with different names.

To boot a virtual floppy, use the **dos -l** ("no load image") option and use the **+a** option to attach the virtual floppy to your DOS process. The syntax is:

dos -l +a*drive_letter*:=*pathname*

where *drive_letter* is either a or b and *pathname* is the full path of the UNIX file containing the virtual floppy. For example:

$ dos -l +aa:=/usr/joe/vflop

Refer to Appendix A in *Using ODT-DOS* for further information on the ±**l** option.

Installing Plug-In Cards in Your Computer

ODT-DOS allows DOS to use standard hardware devices in a convenient way. Standard DOS hardware devices include displays, disk drives, COM ports, expanded memory (EMS), a mouse, printer ports, and the game port. A few pointers on administering several of these devices appear earlier in this chapter. Also refer to *Using ODT-DOS* (especially the description of the ±**a** option) for information on using standard devices.

This section tells you how to install, configure, and use other DOS devices. DOS can use other devices only by communicating directly with those devices, without the intervention of the UNIX system. This form of device access is called direct attachment.

When you add, remove, or change computer hardware, you should make new DOS images. See "Making New DOS Images," later in this chapter, for more information.

Using Directly Attached DOS Devices

Install your directly attached DOS device according to the manufacturer's instructions.

To use a directly attached DOS device, you must use the **+a** option to attach it to the DOS process when you start DOS. The syntax for direct device attachment is:

+a*device_specification*

where *device_specification* consists of the following fields:

io_port_range (typically in the range of 0-3ff, hex)
interrupt_level (3-7, 9-12, 14, or 15)
memory_mapped_io_range (typically c0000-effff, hex)
dma_channel (0, 1, 2, 3, 5, 6, 7)

The device specification fields must be listed in the order shown, but only those needed must be specified. Fields are separated by dots (.). The first field has a leading dot (between it and +a). When a field is not relevant, a single dot is used. An example of a command that starts the DOS environment and directly attaches a device is:

```
# dos +a.2ff.5..
```

Following are more complete descriptions of each of the device specification fields:

- **io_port_range** has this recursive definition:

  X[-*Y*][,*io_port_range*]

  where the form *X* specifies an I/O port address *X* (in hex) and the form *X-Y* specifies an I/O port address range, from low *X* (in hex) to high *Y* (in hex), that the device uses. As the syntax shows, single or multiple ports or ranges may be given.

- **interrupt_level** is a single decimal number that denotes which interrupt level is used.

- **memory_mapped_io_range** has this recursive definition:

  X-Y[,*memory_mapped_io_range*]

  where *X* and *Y* are two hex numbers, separated by a hyphen, denoting the range of the memory-mapped I/O. The first number is the lowest address and the second is the highest address. Each range must start on a 4K boundary and be a multiple of 4K. As the syntax shows, multiple ranges may be specified.

- **dma_channel** is a single decimal number that denotes which DMA channel is used.

Determine the values for **io_port_range, interrupt_level, memory_mapped_io_range**, and **dma_channel** by consulting the hardware technical specifications for the devices being used. If you have difficulty determining the required values for your hardware, contact the hardware manufacturer. Following are examples showing how to use the **+a** option to specify a directly attached device:

- **Directly attached COM1.**[1] The COM1 interrupt is 4, and the ports range from 3f8 to 3ff. The device specification is therefore:

  ```
  .3f8-3ff.4..
  ```

  The last two fields, memory-mapped I/O range and DMA channel, are blank because these fields are not relevant for a COM port. You might use this device specification in commands such as:

  ```
  # dos +a.3f8-3ff.4..
  # dos +a.3f8-3ff.4.. xtalk
  ```

- **A hypothetical hardware device called "widget" that uses memory-mapped I/O in addition to ports and interrupts.** The device has the following characteristics:

  Port ranges: 2f8-2ff and 3f0
  Interrupt channel: 2
  Memory mapped I/O range: c0000-c03ff

  The device specification for this device is:

  ```
  .2f8-2ff,3f0.2.c0000-c03ff.
  ```

  You can attach this device to DOS when you start the DOS environment with the command:

  ```
  # dos +a.2f8-2ff,3f0.2.c0000-c03ff.
  ```

[1] This example is for tutorial purposes only. ODT-DOS is configured by default to attach directly the COM1 and COM2 ports when you use the **+adcom1** or **+adcom2** DOS option.

If you have an application called "colorama" that uses this device, you can invoke the application and attach the device at the same time with the command:

```
# dos +a.2f8-2ff,3f0.2.c0000-c03ff. colorama
```

Incorrect use of device specifications for directly attached devices can crash Open Desktop. Therefore, only the system administrator (logged in as root or super user) is allowed to specify directly attached devices on the command line, as illustrated in the preceding examples. To enable other users to access directly attached devices, you must define tokens for the devices in the *dosdev* file. The next section describes this procedure.

Setting Up /etc/dosdev

The ODT-DOS device specification file */etc/dosdev* serves several purposes. *dosdev* includes specifications for several standard devices, including printers, COM ports, and expanded memory. It also includes comments and example "template" device specifications that you might find useful as you set up *dosdev* specifications for other DOS devices.

Normally, you should not alter existing *dosdev* specifications. You can add your own device specifications to *dosdev*, though. The primary reasons for specifying a directly attached device in *dosdev* are:

- To allow ODT-DOS users to directly attach devices. When you specify a device in *dosdev*, users other than root or super user can directly attach devices.

- To create a short, easily remembered alias (or token) for a device specification. You and other users can use this token with the **dos +a** option instead of fully specifying the device as described in the previous section.

Before specifying a directly attached device in *dosdev*, you should test the device specification using the **+a** option as described in the previous section. Because incorrect use of device specifications for directly attached devices can crash Open Desktop, you should conduct your tests when users will not be inconvenienced if you need to reboot the system.

The syntax for specifying a directly attached device in *dosdev* is:

token **d** *device_specification comments*

Tokens may not contain spaces, commas, or equal signs and may not start with a slash or dot. Tokens also may not resemble a drive letter specification (a single letter followed by a colon). Aside from these restrictions, you can use any alphanumeric characters in tokens. Tokens can be up to 32 characters in length. For example, the *dosdev* line for the directly attached COM1 port discussed in the previous section is:[1]

```
dcom1 d .3f8-3ff.4.. Direct attach COM device.
```

A possible *dosdev* entry you could create for the hypothetical hardware device discussed in the previous section is:

```
widget d .2f8-2ff,3f0.2.c0000-c03ff. Direct attach widget.
```

These lines can be anywhere in *dosdev*. As long as these lines exist in *dosdev*, users can directly attach COM1 or widget with commands such as:

```
$ dos +adcom1
$ dos +awidget
```

Users can also use the **dosopt** command to cause specific applications or the **dos** command to request access to a directly attached device automatically. Use the token defined in *dosdev* when issuing the **dosopt** command. For example:

```
$ dosopt +awidget dosenv.def
```

When you use **dosopt** like this, ODT-DOS looks up specification for the token (''widget'' in this example) in *dosdev* when you run DOS, not when you assign the option with **dosopt**. Therefore, if you change the device specification associated with the token in *dosdev*, ODT-DOS always uses the current specification. Refer to *Using ODT-DOS* for further information on **dosopt**.

Note that directly attached devices cannot be shared. As long as one process is using a directly attached device, other processes are prevented from using the device until the process controlling the device exits.

[1] Recall that this example is for tutorial purposes only. You do not need to add this line to *dosdev*, because it is already there by default.

Mapping Device Names

Open Desktop allows DOS to access a hardware device via different device specifications than the physical device uses. For instance, assume you have a communications program that uses COM1 by default. If you decide you want the application to use the physical COM2 port instead, you can use the mapping feature to make the application treat COM2 as though it were COM1, without reconfiguring the application or physically moving hardware. This feature is useful whenever you want DOS to view a hardware device as though were a different physical device.

To make DOS view a hardware device as though it had different specifications, you map the virtual device specification (the specification that DOS sees) to the physical device specification (the actual physical characteristics of the device). The syntax for mapping device names is:

+a*virtual_device_specification=physical_device_specification*

virtual_device_specification and *physical_device_specification* can be:

- Complete device specifications (including fields for I/O port range, interrupt level, memory-mapped I/O range, and DMA channel).
- Tokens that are defined in *dosdev*.

For example, the tokens **dcom1** and **dcom2** are defined by default in *dosdev*, so you can start the DOS environment and map COM1 to COM2 with the command:

```
$ dos +adcom1=dcom2
```

This command causes DOS to treat the physical COM2 port as though it were COM1.

There is no difference in performance when you map virtual and physical DOS devices like this, except for I/O ports, for which it is more efficient to make the virtual and physical port ranges the same.

Making New DOS Images

The *DOS image* is a file that reflects the configuration of your virtual PC environment. The DOS image is a frozen picture, or snapshot, of DOS after it has been booted and loaded into memory and begun running. This image includes information DOS needs about the system configuration, derived from the hardware installed on the system and the contents of the file */usr/lib/merge/config.sys* at the time the image is made. It allows quick startup of a DOS program from UNIX. The default DOS images (one each for monochrome, CGA, EGA, and VGA display adapters) are contained in the files */usr/lib/merge/mono.img*, */usr/lib/merge/cga.img*, */usr/lib/merge/ega.img*, and */usr/lib/merge/vga.img*.[1] These DOS image files are used by all ODT-DOS users whenever they run any DOS process unless they specifically request a custom image when they start DOS.

There are several reasons to make new DOS images. Typically, when new images are required, the system administrator creates a new default DOS image for each type of display adapter used on Open Desktop. The new default images are then available automatically to all DOS users. The following subsections describe when you need to create new DOS images and how to create them. The topics are:

- Hardware Changes that Require New DOS Images
- Changing the CONFIG.SYS STACKS Command
- Using **dosadmin** to Make New DOS Images

Hardware Changes That Require New DOS Images

The most common reason to make new DOS images is that you have changed your computer hardware. New DOS images are required whenever you make hardware changes that change the BIOS data area or interrupt vectors. You may not know whether a particular hardware change has had these effects. Making new DOS images is simple, however, and it never hurts to make new images even when they are not required. We therefore recommend that you make new default DOS images for all types of displays used on your system after you make any hardware change.

[1] Hercules display adapters also use the */usr/lib/merge/mono.img* file.

Standard ROMs and On-Card ROMs

When you create a DOS image for an EGA or a VGA display adapter card, ODT-DOS uses the data in the display adapter ROM. ODT-DOS can use either the actual physical ROM on the card while making the image, or a standard ROM supplied (as a file) with the ODT-DOS software.

The images created during the ODT-DOS installation procedure use standard ROM files, rather than the on-card ROMs, whenever standard ROMs exist. When you make a new image following the initial ODT-DOS installation, you may be able to choose between a standard ROM and an on-card ROM. When both a standard ROM and an on-card ROM exist for a particular card, you must specify whether you want to use the standard ROM or the on-card ROM. Consider the following trade-offs as you make your choice:

- The image-making process is more reliable when you use standard ROMs. Making an image using some on-card ROMs can fail in unpredictable ways. Symptoms may range from an error message stating that an image cannot be made to the system locking up .

- When you make an image for a display adapter using a standard ROM, you are restricted to using the standard IBM video modes for that card. If you have a display adapter that provides nonstandard functionality and you want to take advantage of that functionality, you must make an image using the on-card ROM.

  NOTE: ODT-DOS may not operate correctly when you use nonstandard video modes. Because extended video modes can vary in unpredictable ways (in their treatment of registers in particular), it is impossible to predict how ODT-DOS will work when you use these modes.

To create a new DOS image, follow the procedures in "Using **dosadmin** to Make New DOS Images," later in this guide.

Changing the config.sys STACKS Command

ODT-DOS allows you to use nearly all *config.sys* commands as you would on a conventional DOS computer. Commands in the root directory *config.sys* file affect all users when they start a DOS process. Commands in a user's home directory *config.sys* file affect only that user's DOS processes.

On Open Desktop, however, DOS does not interpret the *config.sys* STACKS command unless it is incorporated into the DOS image you are using. DOS ignores STACKS commands in the root directory and home directory *config.sys* files and any other *config.sys* files you specify with the +e option at DOS runtime.

To incorporate a STACKS command into one or more DOS images, you first include the STACKS command you want to use in the file */usr/lib/merge/config.sys*. This *config.sys* file is used only in the DOS image-creation process. It includes configuration information (such as device driver definitions) that are needed whenever users run DOS processes.

WARNING: Do not alter any of the lines that exist in */usr/lib/merge/config.sys* when you install ODT-DOS. You can edit this file to include any additional *config.sys* instructions that you want incorporated into a DOS image, but altering any of the factory default lines is likely to result in unexpected and undesirable ODT-DOS behavior. The only reason you are likely to need to modify this file is to add or change a STACKS command.

After adding to */usr/lib/merge/config.sys*, create one or more new DOS images by following the instructions in the next section. You need to make a new image for each type of display to which you want the new */usr/lib/merge/config.sys* to apply.

Using dosadmin to Make New DOS Images

To make new DOS images, follow these steps:

1. If you are creating a new image because you have made hardware changes, make sure that the hardware that requires the new DOS image (for example, a new VGA card) has been installed on the system.

2. Log in as root or become the super user.

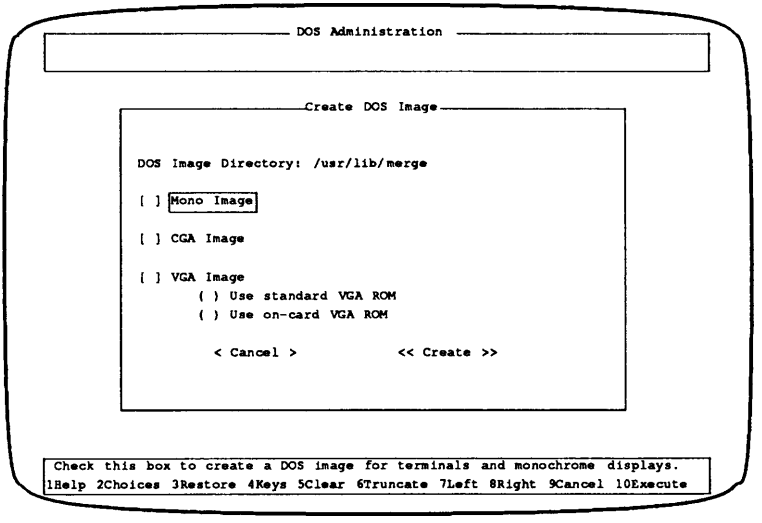
3. Start **dosadmin** by typing:

   ```
   # dosadmin
   ```

   (You may start **dosadmin** from either the DOS or UNIX command line .) The **dosadmin** "DOS Administration" main menu appears.

4. Press the **Left** or **Right Arrow** key (← or →) to move the highlighted field to "ODT-DOS." A submenu opens under "ODT-DOS."

5. If the "Create DOS Image" field is not highlighted, press the **Up** or **Down Arrow** key (↑ or ↓) to move the highlighted field to "Create DOS Image." Then press **Enter** (↵) to select this item and display the following screen:



The "DOS Image Directory" shows */usr/lib/merge* as the default directory for system images. Below this field are fields for each type of image you can create. These fields vary, depending on your computer hardware. For example, fields for VGA images appear only if you have a VGA card installed.

6. If you want to create DOS images in a directory other than */usr/lib/merge*, enter your preferred directory name in the "DOS Image Directory" field. If this field is not already highlighted when you want to change it, use the **Tab** key (⇆) to move the highlight. The directory must be specified by full pathname and may be entered either in UNIX format, using slashes (/) as path separators, or in DOS format, specifying a DOS drive (C: or D:) and using backslashes (\).

NOTE: To use a DOS image stored in a directory other than */usr/lib/merge*, you must explicitly request the image using the DOS **+l** option when you run DOS. See the description of the **+l** option in *Using ODT-DOS* for further information.

7. Now check the box adjacent to the name of each image you want to create. Use the **Tab** key to move the highlight from one field to another, and press **Enter** or **Space** to check the selection box for each image you want to create. You can select any or all of the displayed image types. Each image you select is created in the directory specified in the "DOS Image Directory" field in a file named to correspond to the image type — *mono.img*, *cga.img*, *ega.img*, or *vga.img*.

8. If your system allows you to create images that use either a standard ROM or an on-card ROM, the **dosadmin** screen offers you a choice between the two types of ROM. The default is to use the standard ROM, so by default this selection has an asterisk next to it. If you want to use the on-card ROM, use the **Down Arrow** key (↓) to move the highlight to the "Use on-card ROM" field. Then press **Enter** or the **Space** to select this field. An asterisk appears between the parentheses.

9. If you decide not to complete this procedure, press **Tab** to move the highlight into the "< Cancel >" field and press **Enter**. Otherwise, to complete the procedure, press **Tab** until "<< Create >>" is highlighted, then press **Enter**.

System Files Affected by System Administration

For your reference, the following is a list of ODT-DOS files affected by common administrative procedures. These system files are used by the general ODT-DOS user community. Most of these files are described earlier in this chapter. Refer to Chapter 3 in this guide for further information on */usr/lib/merge/sdfile*, */etc/dosenv.def*, and */etc/dosapp.def*.

- */autoexec.bat*
- */config.sys*
- */etc/dosenv.def*

- */etc/dosapp.def*
- */etc/dosdev*
- */usr/lib/merge/mono.img*
- */usr/lib/merge/cga.img*
- */usr/lib/merge/config.sys*
- */usr/lib/merge/ega.img*
- */usr/lib/merge/vga.img*
- */usr/lib/merge/sdfile*

Be sure you inform all affected users (via mail or message of the day) whenever you modify any of these files.

Chapter 3

Installing DOS Applications

This chapter tells you how to install and configure DOS applications on the Open Desktop fixed disk and how to remove them. In general, you can install most applications simply by entering the DOS environment as described in *Using ODT-DOS* and then following the application manufacturer's instructions for installing on a fixed disk. ODT-DOS, however, supplies a menu system called **dosadmin** that automatically configures applications so they can be used from both the UNIX shell and the DOS environment. The **dosadmin** utility also provides simple ways of keeping track of installed applications and tailoring them.

A few copy-protected applications cannot be installed on the shared DOS/UNIX filesystem. You can install these applications on the DOS partition as described later in this chapter.

Some applications cannot be installed on the fixed disk at all because they boot their own operating systems from the distributed diskette to run the program. To run these applications, use the ODT-DOS **dosboot** command as described in *Using ODT-DOS*.

This chapter includes the following sections:

- Installing DOS Applications Using **dosadmin**
- Installing Copy-Protected DOS Applications
- Removing DOS Applications

Installing DOS Applications Using dosadmin

If you want to use DOS applications only from the DOS environment, you do not need to read this section. You can start a DOS environment and install your applications according to the manufacture's instructions. The **dosadmin** procedures described here configure DOS applications so they can be used conveniently from the UNIX shell. If you choose to install a DOS application without using **dosadmin**, you can still use **dosadmin** to configure the application for use from the UNIX shell at a later time.

The system administrator can use **dosadmin** to install public applications in a directory accessible to all users. Individual users can use **dosadmin** to install personal DOS applications in their own directories.

You can use **dosadmin** to install most DOS applications that are designed to be installed on a personal computer fixed disk. The exceptions are DOS applications using copy-protection schemes that require installation on an actual DOS filesystem or on a system running DOS as its native operating system. For further information on this subject, see "Installing Copy-Protected Programs," later in this chapter.

Following is an outline of the procedure for installing applications using **dosadmin**:

- After logging into ODT-DOS, type **dosadmin**.
- Select the "Install Applications" menu and answer some pre-installation questions concerning the application.
- Install the application on the fixed disk according to the application manufacturer's instructions.
- Respond to post-installation **dosadmin** prompts concerning system configuration required by the application.

Following are step-by-step instructions for using **dosadmin** to install DOS applications on the Open Desktop fixed disk. Before installing any application, you should read through these instructions to make sure you have the necessary information at hand. You should also be familiar with the application manufacturer's installation instructions.

You follow the same general procedures whether you are the system administrator installing an application in a public directory or a user installing a personal application in your home directory. The next section, "Installing Public DOS Applications," tells the system administrator how to install applications for use by all system users. The following section, "Installing Personal DOS Applications," tells you how to install a personal application in your own directory.

The screen displays and procedures described in these sections use Lotus 1-2-3 as a typical example. Your responses to **dosadmin** prompts and the precise appearance of your display will differ, depending on your application and how you want to configure it.

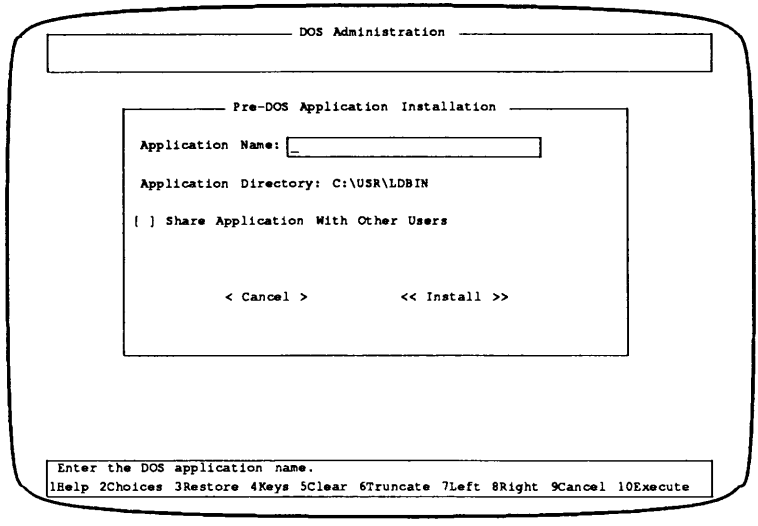
Installing Public DOS Applications

The Open Desktop system administrator can use **dosadmin** to install DOS applications intended for shared use by all system users. To install a public DOS application, follow these procedures:

1. Log in as root.
2. Type:[1]

   # dosadmin

   The **dosadmin** main menu is displayed.
3. The "Applications" submenu should be highlighted. If it is not, press the **Left** or **Right Arrow** key (← or →) until it is.
4. Press the **Down Arrow** key (↓) to move the highlight to "Install" and press **Enter** (↵). The "Install Applications" menu appears, and your display looks like this:



[1] You can run **dosadmin** from either the DOS environment or the UNIX shell. If you start **dosadmin** while using the DOS environment, your prompt before starting **dosadmin** and after exiting **dosadmin** is a DOS prompt, such as **C>**. When you start **dosadmin** from the UNIX shell, your prompt before and after using **dosadmin** is **#**. The actual **dosadmin** procedures and displays are identical, whether you start **dosadmin** from the UNIX shell or in the DOS environment.

5. Type in the name of the application you are installing. The name you type here is entered in the application database, allowing you to access the application easily, by name, with the List Applications, Remove Applications, and Tailor Applications menus. The name you type is associated in the database with one executable file. With applications (such as Lotus 1-2-3) that include more than one executable file (LOTUS.COM and 123.COM), it is useful to include the name of the command you normally use with that application. For example, if you normally use Lotus by typing **lotus**, you might enter **Lotus 1-2-3 (lotus)** as the name of the application. If you normally use Lotus by typing **123**, you could enter **Lotus 1-2-3 (123)** as the name of the application.

   If you make a mistake while typing, use the **Bksp** key (←) to back up and correct it, or use the **F5** key to clear the field and retype your entry.

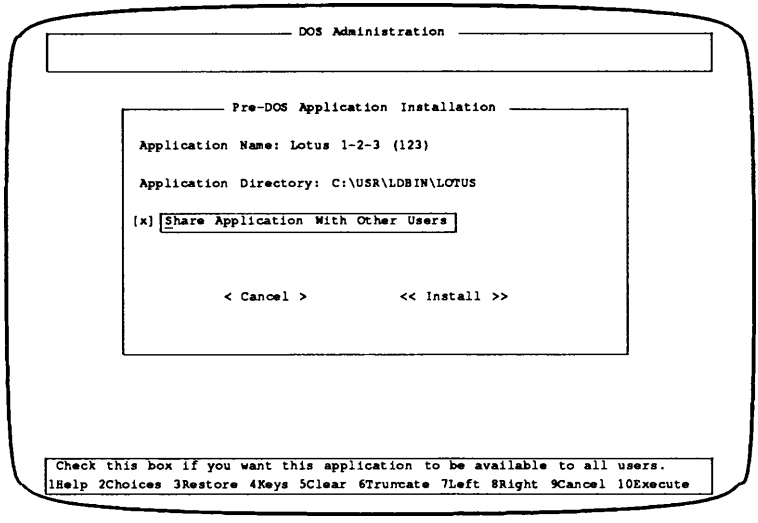
6. Press the **Tab** key (⇆) to move to the "Application Directory" field. "C:\USR\LDBIN" is already filled in. We recommend you install public DOS applications in this default directory.

7. If you want to install the application in \USR\LDBIN (the default), skip to step 8.

   If you want to change the default drive or directory for installing the application, enter your choice here. You may, for example, want to install the application in a subdirectory, such as \USR\LDBIN\LOTUS. To specify a subdirectory like this, use the right arrow key (→) to move the cursor to the right of the displayed "C:\USR\LDBIN", and type in the name of the subdirectory. If the directory you specify does not already exist, **dosadmin** creates it during the installation procedure.

8. Press the **Tab** key (⇆) to highlight the "Share Applications With Other Users" field.

9. Because you are installing a public application, press the **Space** or **Enter** key to check the "Share Application with Other Users" box. The display looks like this:

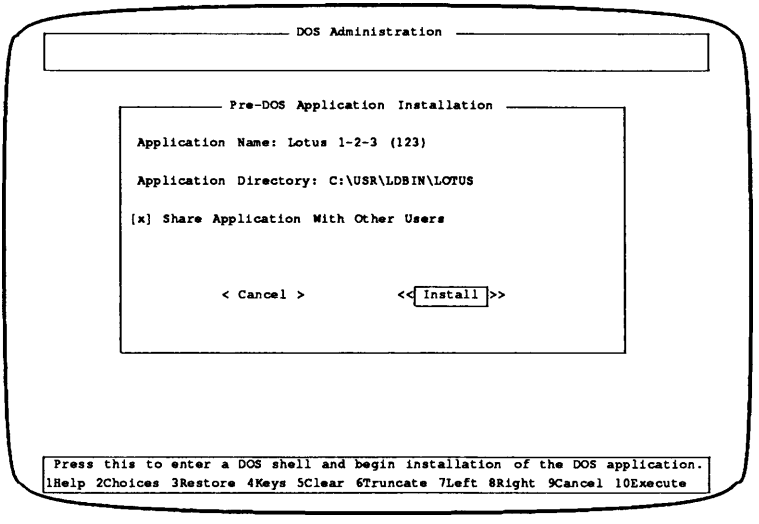


When you check the "Share Applications" box, **dosadmin** sets permission modes so all users can run the application.

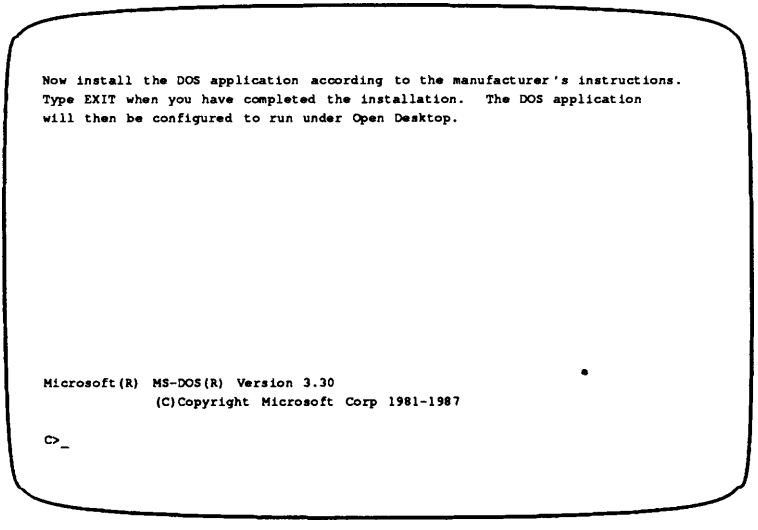
10. Move to the "< Cancel >" field by pressing the **Tab** key. When "< Cancel >" is highlighted, you can press **Enter** to cancel the installation procedure and return to the top-level **dosadmin** menu.

    If you choose to continue the installation, check your entries for accuracy. If you want to make any changes, you can return to any field by pressing the **Tab** key repeatedly, and then type in your corrections.

11. When you have confirmed your entries, press the **Tab** key until "<< Install >>" is highlighted like this:



12. Press **Enter** to save your choices and start the next part of the installation process. Your display is now similar to this:



13. Continue the installation procedure by following the application manufacturer's instructions for installation on a personal computer fixed disk. Your current working directory and drive are those you specified in the "Application Directory" field of the Pre-DOS Application Installation menu. If the manufacturer's instructions require you to identify the drive or directory in which you are installing the application, you should normally enter the same values you used in the Pre-DOS Application Installation menu. However, if you change your mind about the drive or directory, you can correct the **dosadmin** database later.

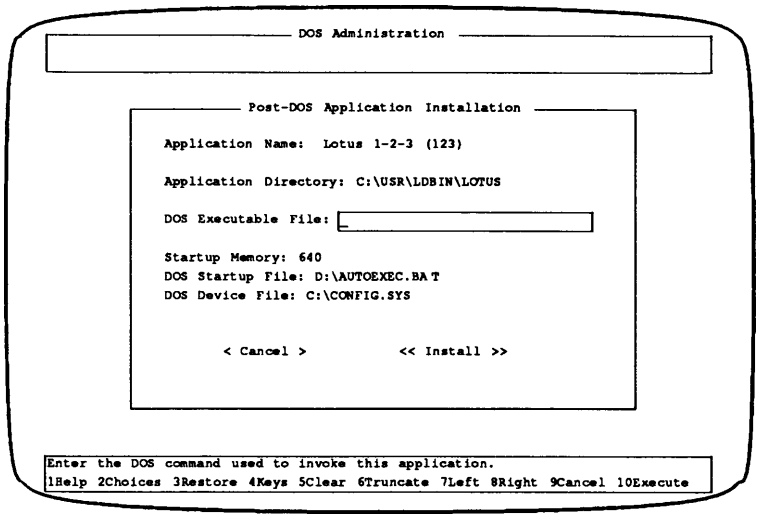
14. When you have completed installation according to the manufacturer's instructions, type:

    C> **exit**

    WARNING: Be sure you type **exit** and not **quit** when you have finished the manufacturer's recommended installation procedure. If you type **quit**, you have to start the installation procedure over from the beginning.[1]

[1] However, if you have finished installing the application before accidentally typing **quit**, you can skip step 13 when you redo the installation procedure. Since the application is already installed, there is no need to install it again.

The Post-DOS Application Installation menu appears, as follows:



15. The "DOS Executable File" field is highlighted. Type in the command you use when you run the program. For example, type **123** if this is the command you use to run Lotus 1-2-3.

    If you use more than one command with this application, see "Configuring DOS Applications for Use from the UNIX Shell" later in this chapter.

16. Check all fields of the display for accuracy. Most applications will run correctly without further changes to this display, but you should now correct or change the displayed values if necessary. To make changes, press the **Tab** key repeatedly until the field you want to change is highlighted. Then type in the correct information.

Following are points you should consider as you review the values for each field:

- **Application Name:** The displayed name is the name you entered during the pre-installation phase. The **dosadmin** database associates the application name shown here with the command shown in the "DOS Executable File" field.

- **Application Directory:** This field shows the drive and directory you entered during the pre-installation phase. If you installed the application on a different drive or in a different directory for any reason, you should correct this field now. The directory name displayed must be that of the directory containing the executable DOS file (for example, 123.COM).

- **DOS Executable File:** This is the command you type when using the application. The **dosadmin** database associates the command name shown here with the application name shown in the "Application Name" field.

- **Startup Memory:** This is the amount of memory (in Kbytes) reserved for this application when you run it from the UNIX shell. The value displayed is 640 Kbytes, unless factory defaults have been changed. All applications run correctly with 640 Kbytes of memory, but not all applications need this much memory. If your application runs correctly with less memory, you can increase ODT-DOS efficiency by specifying a smaller value than 640. Refer to the application manufacturer's instructions for the recommended memory, and enter it here.

- **DOS Startup File:** The file named in this field (D:\AUTOEXEC.BAT by default) is executed automatically whenever you run the application from the UNIX shell. When you are logged in as root, D:\AUTOEXEC.BAT is the root directory AUTOEXEC.BAT in the shared DOS/UNIX filesystem. That is, D:\AUTOEXEC.BAT is just another name for C:\AUTOEXEC.BAT or **/autoexec.bat**. If you have created or added to the D:\AUTOEXEC.BAT file while installing this application, it will be executed as expected when you run the application.

Users' home directory AUTOEXEC.BAT files run automatically in addition to D:\AUTOEXEC.BAT when they execute the DOS application from the UNIX shell as long as the default D:\AUTOEXEC.BAT file is displayed in the "DOS Startup File" field. If you want only the root directory AUTOEXEC.BAT to run when users invoke this application, replace the displayed D:\AUTOEXEC.BAT with C:\AUTOEXEC.BAT.

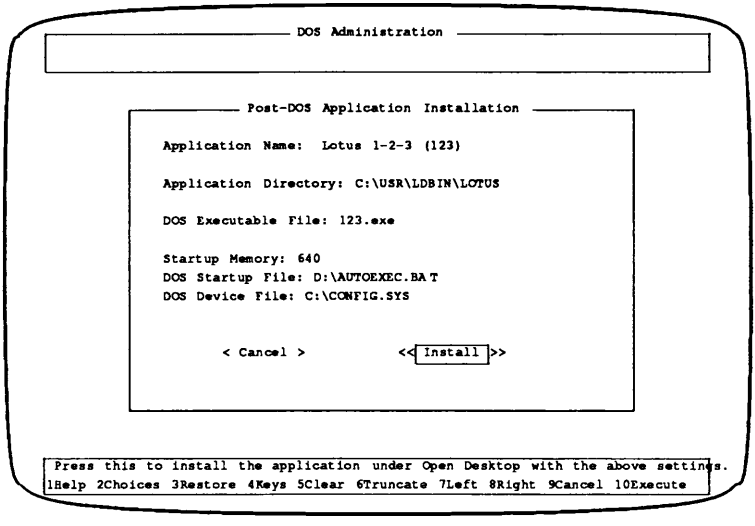
If you have created a file other than D:\AUTOEXEC.BAT that should run every time you use this application, enter the name of the file in this field. For example, if your application is supposed to run the file C:\USR\LDBIN\APPL.BAT every time you invoke it, type in this field:

```
C:\usr\ldbin\appl.bat
```

If you do not want any AUTOEXEC.BAT file to run automatically when you use this application, erase the field by pressing the **F5** key.

- **DOS Device File:** This field lists the device configuration file interpreted by DOS when you run your application from the UNIX shell. The system default for this field is a single file—C:\CONFIG.SYS—when you install a public application. You can accept or change this value as appropriate for the application you are installing. Note that, even though **dosadmin** displays only the system default device file, when users run the application, two device files, C:\CONFIG.SYS and their home-directory CONFIG.SYS, are interpreted if they both exist. (Home directory CONFIG.SYS files do not exist until users create them.)

17. When the display shows correct information, press the **Tab** key until "Install" is highlighted. The display should resemble this:



18. Press **Enter** to complete the installation procedure. An "In Progress" message appears while **dosadmin** configures your application. Then the top-level **dosadmin** menu returns.

    If you want to install another application, press **Enter** to redisplay the Install Applications menu. Otherwise, to return to your system prompt, press the **Left** or **Right Arrow** key (← or →) until "Exit" is highlighted, and press **Enter**.

Defining the Search Path

After you install a DOS application, you may want to update the search path if you have installed the application in a directory that is not already in users' search paths. You can update the search path for all users by changing the **PATH** definition in **/etc/default/login**.

The UNIX search path applies by default both at the UNIX shell and in the DOS environment.

Installing Personal DOS Applications

The procedures for installing personal DOS applications are almost identical to the procedures described in the previous section for public DOS applications. To install personal applications, refer to the instructions in the previous section, "Installing Public DOS Applications," and note the following differences:

- **Step 1:** Log in using your normal login name. Your prompt is **$** instead of **#**.
- **Steps 6 and 7:** The default directory for installing personal DOS applications is D:\ (your home directory). Normally, you should not change this field to show a different drive from drive D:. You may, however, want to install your DOS application in a subdirectory subordinate to your home directory. You can specify a subdirectory by filling in this field appropriately.
- **Step 9:** You are not required to share the application with other users. If you want it to be executable by others, press the **Space** or **Enter** key to check the "Share Application With Other Users" box. Otherwise, just press **Tab** and continue with the rest of the application installation procedures.
- **Step 16:** When you install personal DOS applications, the "DOS Startup File" field displays D:\AUTOEXEC.BAT by default, just as it does when you install public DOS applications. When you are not logged in as root however, D:\AUTOEXEC.BAT is your home directory AUTOEXEC.BAT rather than the root AUTOEXEC.BAT. If the "DOS Startup File" field displays "D:\AUTOEXEC.BAT," DOS interprets both the root and your home directory AUTOEXEC.BAT files (if they both exist) when you run this application from the UNIX shell. If you want a different startup file to be interpreted when you run this application, you can change this field as described in step 16 under "Installing Public DOS Applications."

  When you install personal DOS applications, the "DOS Device File" field by default shows two files: C:\CONFIG.SYS and D:\CONFIG.SYS. These files are the system default CONFIG.SYS file in the root and your home directory CONFIG.SYS file. DOS interprets both files, if they are both listed and both exist, when you run this application from the UNIX shell. (If no filename is displayed, then no device configuration file is interpreted when you run this application.) If your application requires device configuration information from a CONFIG.SYS file, it must be listed here.

Even if your application does not require a CONFIG.SYS file, you should normally not delete the displayed default filenames. If your system hardware changes in the future, one or more of the listed files may contain configuration information required when you run any DOS program.

Defining the Search Path

After you have installed a personal DOS application, you may want to update your search path. You can change the UNIX search path that is defined whenever you log in by modifying your home directory **.profile** file. The UNIX search path applies by default in the DOS environment also.

You can also use standard DOS methods to define your search path in the DOS environment.

Configuring DOS Applications for Use from the UNIX Shell

You can use **dosadmin** to configure DOS applications that are already installed so they can be used from the UNIX shell. This procedure is useful under the following circumstances:

- You previously installed a DOS application that has never been configured for use from the UNIX shell.
- You previously installed an application such as Lotus 1-2-3 that has more than one executable file and has configured only one file for use from the UNIX shell.
- You want to change the characteristics previously assigned to an application with **dosadmin**.

To configure an already-installed application using **dosadmin**, you simply follow all the steps listed under "Installing Public DOS Applications" with one exception: do not complete step 13, installing the application according to manufacturer's instructions, since the application is already installed. Instead, just type **exit** at the DOS prompt (step 14) and continue with the Post-DOS Application Installation menu.

During this procedure, **dosadmin** displays the application's current values for startup memory, startup file, and device files. You can change these values before exiting **dosadmin** if you wish.

When you exit **dosadmin**, the application is added to the **dosadmin** database, and you can access the application using the **dosadmin** List Applications, Remove Applications, and Tailor Applications menus.

How dosadmin Configures Applications

dosadmin automatically accomplishes the following operations when you use it to install DOS applications:

- Adds the name of the application to a database of installed applications. The database is used by other **dosadmin** functions, including the List Applications, Remove Applications, and Tailor Applications menus.
- Links DOS executable files to UNIX files without **.bat**, **.com**, or **.exe** extensions.
- Sets UNIX permission modes so the application can be used from both the DOS environment and the UNIX shell.
- Adds appropriate **dos** options. (Refer to *Using ODT-DOS* for further information on **dos** options.)

NOTE: When you install DOS applications using **dosadmin**, **dosadmin** creates a file named **sdfile** that stores information about the applications. If you are logged in as a user and are installing personal applications in a directory you own, **sdfile** is stored in your home directory. If you are the system administrator logged in as root, **sdfile** is stored in **/usr/lib/merge**. You need not be concerned with the contents of **sdfile**, but be sure you do not accidentally delete it.

Installing Copy-Protected DOS Applications

A few DOS applications are designed by the manufacturer to prevent installation on more than one machine or execution by more than one user at a time. You can use these applications with Open Desktop, but the installation procedures differ according to the type of copy protection used.

Many different copy-protection methods are in use, and it is impossible to recommend installation procedures that apply universally. There are, however, two common classes of copy-protected applications for which we can supply general recommendations that apply in most cases. These classes are:

- applications that use "key disks."
- applications that use special installation utilities.

Procedures for installing applications in each of these classes are described in the following sections.

Using a Key Disk

An application that uses a key disk allows you to install part of the application on the system fixed disk but requires you to insert a protected diskette in the diskette drive whenever you use the application.

Such applications generally require no special installation procedures to work on Open Desktop. You can install them by running a DOS environment and following the manufacturer's instructions or the instructions in the previous section, "Installing DOS Applications Using **dosadmin**." When you run an application that requires a key disk, you use the key disk just as you would on a conventional personal computer running DOS.

Using Special Installation Procedures

Another class of copy-protected applications uses special installation (and sometimes also removal) procedures to prevent illegal copying of the application. These applications typically include a special installation utility (often invoked with a command such as "install"). They are intended to be installed in their entirety on the system fixed disk so that no key disk is required. You can install such applications on Open Desktop but, because different copy-protection methods are in use, you may have to try more than one installation procedure. The procedures, listed in the order you should try them, are:

- If you do not intend to use the application from the UNIX shell, run a DOS environment and install the application according to the manufacturer's instructions.
- Install the application according to the standard instructions under "Installing DOS Applications Using **dosadmin**," earlier in this chapter.

- Install the application on DOS drive E: (the DOS partition). This procedure is described in the next section, "Installing DOS Applications on the DOS partition."

- Shut down Open Desktop, boot DOS, and install the application on the DOS partition. This procedure is also described in the next section.

Installing DOS Applications on the DOS Partition

This section describes two methods of installing DOS applications on the DOS partition. These procedures are particularly useful when you install copy-protected applications that cannot be installed using **dosadmin**.

Installing DOS Applications under Open Desktop

This procedure is required for a few copy-protected DOS applications that must be installed on an actual DOS filesystem rather than on the shared DOS/UNIX filesystem. This installation method works only if you have a DOS partition on Open Desktop. The DOS partition can be a physical partition (that can be booted and run independently of Open Desktop), or it can be a virtual partition. If your system does not have a DOS partition, you can create one as described in Chapter 2 of this guide.

NOTE: These instructions assume you are installing an application on the physical DOS partition (drive E:). If you are installing on a virtual partition, you must first attach the partition to your DOS process as described in "Administering Virtual DOS Partitions and Virtual Floppy Disks" in Chapter 2 of this guide.

1. Log in to Open Desktop. If you are installing an application in a virtual partition that you own, just log in as you usually do. If you plan to install an application in the physical DOS partition, you must log in as a user who has permission to read and write the physical DOS partition. (By default, all users have read and write permission.)

2. Start the DOS environment by typing (at the UNIX prompt):

   ```
   $ dos
   ```

   Your prompt changes to the DOS prompt **C>**.

3. Change your working drive to drive E: by typing:

```
C> e:
```

4. Make sure you are working in the directory in which you want to install the application. If necessary, use the DOS CD (change directory) command to move to the desired directory.

5. Install the DOS application according to the manufacturer's instructions. For example, if the application has an installation program named INSTALL, invoke it by entering (at your DOS prompt):

```
E> install
```

(If the manufacturer provides no other installation instructions, use the DOS COPY command to transfer all files from drive A: to drive E:.)

The application is now usable from DOS drive E:.

Installing Applications under "Raw" DOS

Use this procedure for the few copy-protected DOS applications with special timing requirements that make them impossible to install while ODT-DOS is running.

NOTE: This method works only if your system has a physical DOS partition that can run DOS as a stand-alone operating system.

Since you boot your system under DOS during this procedure, you must have one of the following:

- A bootable DOS system diskette. If you do not already have such a diskette, you can create one by inserting a diskette into drive A: and typing:

```
$ dos format a: -s
```

- A bootable DOS partition. (It is bootable if you formatted it using the FORMAT /S option.)

Follow these procedures:

1. Shut down the UNIX system according to the instructions in the *Administering ODT-OS*.

2. Reboot the system from a bootable DOS diskette or from the DOS partition.

3. Follow the manufacturer's instructions to install the application on the DOS partition. The DOS partition is accessible as drive C: when you are running raw DOS.[1]

4. When you are finished installing the application, reboot the UNIX system.

The application should now be usable like any other application on the DOS partition. When ODT-DOS is running, the DOS partition is drive E:.

Removing DOS Applications

Following is an outline of the procedures you follow to remove installed DOS applications from the Open Desktop fixed disk. Each of these steps is described in the following sections.

- Remove the application according to the manufacturer's instructions.
- Remove UNIX links to DOS executable files.
- Remove the **dosadmin** database entry for the application, if there is one.
- Redefine the search path, if necessary.

[1] A few DOS applications must be installed and run on the same drive. Since you access the DOS partition as drive E: when ODT-DOS is running and as drive C: when you have booted raw DOS, you should not install these applications directly on drive C:. While your system is running raw DOS, before installing one of these applications, use the DOS SUBST command as follows:

```
C> subst e: c:\
```

The DOS partition is then accessible as drive E:. Continue with the installation procedure, referring to the DOS partition as drive E:.

Removing the Application from the Fixed Disk

Remove a DOS application from the Open Desktop fixed disk by following the manufacturer's instructions. If no instructions are provided, it is usually safe to remove the application's files using the UNIX **rm** or DOS DEL command.

Note that some copy-protected applications require special removal procedures. If you do not follow the required procedure, you may not be able to reinstall the application at a future time.

Deleting UNIX Links

If the DOS application you have removed was configured for execution from the UNIX shell, there are UNIX files that were linked to DOS executable files. (For example, **ws** would be linked to WS.COM.) When you remove DOS executable files (files with extensions ending in .COM, .EXE, or .BAT), the corresponding UNIX links are not automatically removed. When you remove a DOS application, you should also delete any UNIX links to the DOS files you are removing. To delete these links, you simply use the UNIX **rm** or DOS DEL command:

```
$ rm ws
```

Removing the dosadmin Database Entry

If you used **dosadmin** to install or configure your application when you installed it, there is an entry for the application in the **dosadmin** database. When you remove an application, you should remove its database entry. To remove a **dosadmin** database entry for an application, proceed as follows:

1. From either the UNIX shell or the DOS environment, type:

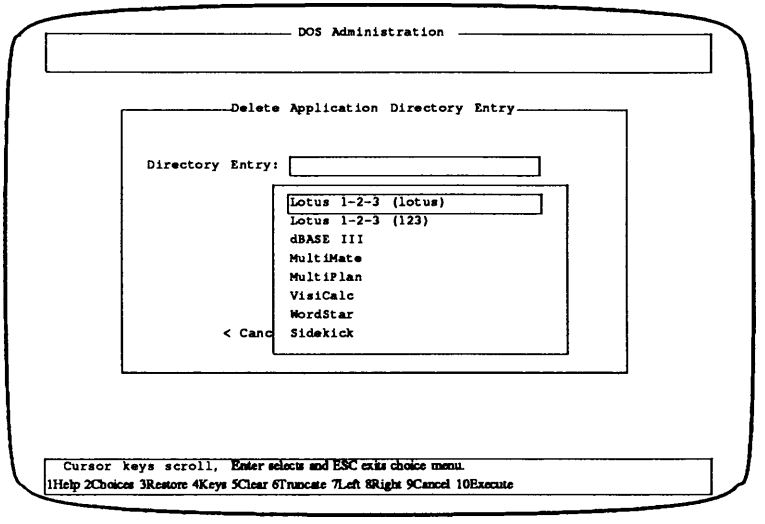
   dosadmin

   The main **dosadmin** menu screen is displayed.

2. If the "Applications" menu is not already highlighted, use the **Left** or **Right Arrow** key (← or →) to highlight it.

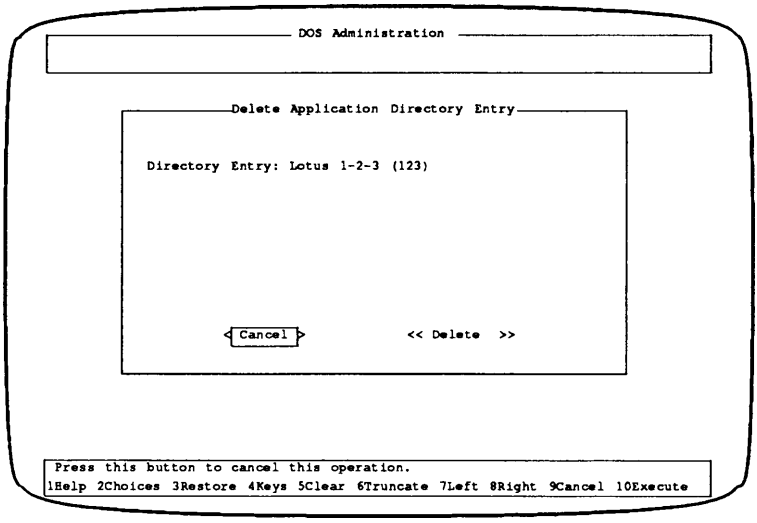
3. Press the **Down Arrow** key (↓) to move the highlighted field to "Remove."

4. Press **Enter** (↵) to display the "Remove Application" menu. The menu displays a list of the applications currently installed in the **dosadmin** database, like this:



5. Select the application you want to remove by pressing the **Down Arrow** key (↓) until the application name is highlighted. If there are more installed applications than there is room to display, you can press the **Down Arrow** key when you reach the bottom of the window to scroll the list and display additional names. To move up in the list, use the **Up Arrow** key (↑), which also scrolls when you reach the top of the window.

6. When the application you want to remove is highlighted, press **Enter**. The application you have chosen is displayed in the "Directory Entry" field, like this:



7. If you choose to cancel the operation and leave the application in the **dosadmin** database, press **Enter** while "< Cancel >" is highlighted. Otherwise, to remove the application, press **Tab** (⇆) to move the highlighted field to "<< Delete >>."

8. Press **Enter**. The application is then removed from the **dosadmin** database, and the Applications menu reappears on your screen.

9. Press the **Left** or **Right Arrow** key (← or →) to move the highlighted field at the top of the menu to "Exit."

10. Press **Enter** to exit **dosadmin** and return to your system prompt.

Redefining the Search Path

After you have removed a DOS application, you may want to redefine your search path. Depending on whether the application you removed was a public or a personal application and on how you define your search path, you might want to modify the **PATH** definitions in **/etc/default/login**, your home directory **.profile** file, or an AUTOEXEC.BAT file.

Administering ODT-DATA

ODT-DATA is based on technology developed by **INGRES CORPORATION**, and includes the following **INGRES** components:

- INGRES/DBMS and SQL Terminal Monitor
- INGRES/User Interfaces
  - Query-by-Forms
  - Report-by-Forms
  - Report Writer
  - Menu
  - Forms Runtime Systems and VIFRED
- INGRES/NET with TCP/IP Support
- INGRES/WindowView
- INGRES/ESQL Preprocessor for C

12/21/89-1.0.0
Processed: Wed Dec 20 11:08:19 PST 1989

Contents

Chapter 1: Introduction 1
Introduction to Release 6 2
Organization of This Document 2
Associated Publications 4

Chapter 2: Overview of Installation Tools 5
ODT-DATA Installation Utility—iibuild 5
ODT-DATA Installation and DBMS Server Start Up 5
The Installation Shut Down Utility—shutserver 6

Chapter 3: Configuration Decisions 7
Configuration Requirements 7
General Suggestions for Avoiding Problems 11

Chapter 4: Installing ODT-DATA 13
Manual Initialization 13

Chapter 5: Maintenance Utilities 19
The Server (iidbms) Maintenance Utility—iimonitor 19
Shared Memory and Semaphores Report Utility—csreport 22
The Locking Facility Report—lockstat 24
The Logging Facility Report—logstat 26

Chapter 6: Installation Reference Material 31
ODT-DATA Installation and Server Start Up Utility 31
ODT-DATA Installation Shutdown 41

Chapter 7: Troubleshooting with Log Files 45
ODT-DATA Log Files 45

Appendix A: ODT-DATA Startup Files 47
Installation-Wide Startup Files 47
Database-Specific Startup File 47
User-Specific Startup File 48

Appendix B: Authorizing User Access to ODT-DATA and Databases 49
Database Access 49
Defining the Terminal 50
Invoking accessdb 51
Using accessdb 51
Functions in accessdb 52
Summary of Accessdb 59

Appendix C: ODT-DATA Environment Variables 61
Setting Installation Wide Environment Variables 61
Setting User Defined Environment Variables 62
Environment Variable List 62

Appendix D: ODT-DATA System Recovery 71
Using finddbs 71

Appendix E: Running ODT-DATA under the Network File System 75
Configuration Scenarios 75

Glossary 81

Chapter 1
Introduction

ODT-DATA is a relational database management and application development system, based on technology developed for INGRES by Relational Technology, Inc.

This book describes the procedures for installing ODT-DATA and authorizing user access to specific databases. It also provides information for the ODT-DATA system administrator.

The ODT-DATA system administrator is the individual responsible for the ODT-DATA installation and the authorization of user access to ODT-DATA. The ODT-DATA system administrator is also responsible for installing ODT-DATA updates and new releases, but does not necessarily have responsibility for databases maintained under the ODT-DATA system. A database administrator (DBA) is designated for each database. The DBA is usually the user who creates a database. The DBA is responsible for the creation of shared tables and authorization of user access to shared tables. The DBA is also responsible for backup and recovery of the database.

Important Note

ODT-DATA installation requires access as "root" on UNIX®. Traditionally, access to "root" privileges presumes a knowledge of the UNIX operating system. This publication assumes that anyone installing ODT-DATA has such a knowledge of UNIX.

If you possess the "root" password and do not have minimal knowledge of UNIX refer to the UNIX sections of the Open Desktop™ *User's Guide* and *Administrator's Guide*. If you are unfamiliar with UNIX, be sure that someone with UNIX expertise can assist you with the installation of ODT-DATA.

Read this publication before proceeding.

Introduction to Release 6

The ODT-DATA DBMS server architecture provides access to the DBMS through a shared process. With the server architecture comes a logging and locking facility that is configured during the ODT-DATA installation. The DBMS logging facility is implemented by two processes. The components of the logging system include: a process to handle online recovery and a process to archive journaled data. An installation-wide logging file keeps track of all ODT-DATA transactions and calls within the DBMS server. The logging facility logs ODT-DATA transactions, manages the logging file, and insures that log records are accessible to the recovery and archiver routines.

The **recovery** process (**dmfrcp**) handles online recovery from system failures and inconsistencies generated by user actions. Consistency points are written into the logging file to allow online recovery when a problem is detected. This permits other users to continue working in the database while the inconsistency is corrected.

The **archiver** process (**dmfacp**) removes completed transactions from the logging file and writes them to the corresponding journal files for the particular database. This process sleeps until sufficient portions of the logging file are ready to be archived. The ODT-DATA system administrator can tune the values that define the logging system.

The recovery and archiver processes are daemon processes started at boot time. They are available on the system at all times and reduce the startup times required to begin an ODT-DATA session. They also reduce the resource requirements, since sessions can use a single process to service DBMS requests, the server.

Organization of This Document

This book contains the following items:

- Chapter 1 introduces this book.
- Chapter 2 describes various installation tools:
  - the ODT-DATA installation utility (**iibuild**),
  - the ODT-DATA installation start up utility (**iistartup**), and
  - The ODT-DATA installation shut down utility (**shutserver**).

- Chapter 3 discusses various system requirements and configuration decisions.
- Chapter 4 describes how to install ODT-DATA
- Chapter 5 discusses maintenance utilities, including:
  - server monitoring and control utility (**iimonitor**),
  - shared memory and semaphores report utility (**csreport**),
  - locking facility report (**lockstat**), and
  - logging facility report (**logstat**).
- Chapter 6 discusses miscellaneous topics in installation:
  - DBMS server creation utility,
  - the logging and locking facility, and
  - ODT-DATA Release 6 shutdown procedure.
- Chapter 7 discusses troubleshooting ODT-DATA with log files.
- Appendix A discusses ODT-DATA startup files.
- Appendix B discusses authorizing user access to ODT-DATA and databases.
- Appendix C discusses recovery of the ODT-DATA master database.
- Appendix D discusses ODT-DATA environment variables.
- Appendix E discusses running ODT-DATA under Network File System (NFS™).

Associated Publications

The table below lists all the ODT-DATA books available with each Open Desktop product:

Open Desktop:
■ *Administering ODT-DATA*
■ *Using ODT-DATA*
Open Desktop Development System:
■ *ODT-DATA Embedded SQL User's Guide*
■ *ODT-DATA Embedded Open SQL Forms Reference Manual*
■ *ODT-DATA Open SQL Reference Manual*
■ *ODT-DATA Embedded SQL Companion Guide for C*
Open Desktop Server Upgrade:
■ *Administering an ODT-DATA NET Server*
Open Desktop Optional Documentation:
■ *Using ODT-DATA Through Forms and Menus*
■ *ODT-DATA Report-Writer Reference Manual*
■ *ODT-DATA SQL Reference Manual*

Chapter 2

Overview of Installation Tools

This chapter provides an overview of the installation tools. The following tools are located in the *$II_SYSTEM/utility* directory and can only be executed by the "ingres" user. Only the "ingres" user has access to these tools.

ODT-DATA Installation Utility—iibuild

The ODT-DATA Release 6 installation utility is **iibuild**. This script is run by the "ingres" user during the ODT-DATA installation procedure described in Chapter 4. The **iibuild** utility performs the following tasks during installation:

- ensures that the ODT-DATA environment is set up for installation
- checks that there are sufficient system resources for installation
- sets up the ODT-DATA installation, including ownership and permissions
- installs system logging and locking resources
- installs the ODT-DATA log file
- installs and starts logging, locking, and the DBMS server
- creates the database database (**iidbdb**)

ODT-DATA Installation and DBMS Server Start Up

You can use **iistartup** to start up a single server or an installation. The **iistartup** utility is used by the **iibuild** script to start up your ODT-DATA installation during the ODT-DATA installation procedure.

Depending upon the option used to invoke the utility, **iistartup** will either start up an installation automatically or prompt you through your choice of the following tasks:

- install shared memory resources
- configure and create a log file
- configure logging and locking parameters
- initialize a log file
- start up the recovery process
- start up the archiver process
- configure the DBMS options and start up the server

If you choose a configuration option, you are prompted for parameters. Refer to Chapter 6 for configuration parameters and **iistartup** syntax.

The Installation Shut Down Utility—shutserver

shutserver is a utility that can be used to shutdown individual servers or the entire installation. **shutserver** will prompt for the following options:

- Bring down server process(es) designated by communication address
- Reconfigure the server(s)
- Stop the installation's archiver and recovery processes
- Change the logging and locking facility parameters
- Deinstall shared memory

Refer to Chapter 6 for **shutserver** syntax and emergency shutdown information.

Chapter 3
Configuration Decisions

This chapter outlines the system requirements to configure and install ODT-DATA and gives suggestions for avoiding problems with your ODT-DATA installation.

Configuration Requirements

This section discusses the requirements for disk space, memory, semaphores, and swap space, as well as the required ODT-DATA environment variables.

Disk Space Requirements

Approximately 32 Mbytes of disk space are required to load your ODT-DATA installation media. In addition, a minimum of 4,096 Kbytes are required for your log file. You must also estimate the space needed for databases, journals and checkpoints.

Memory Requirements

To run ODT-DATA, you must configure your UNIX kernel with sufficient shared memory resources to support ODT-DATA installation, locking and server requirements. Your ODT-DATA installation requires 208,000 bytes of shared memory; 8,192 bytes for a system segment and a minimum of 204,800 bytes for a lock segment. In addition, each DBMS server requires a shared memory segment. The size is calculated as '16,384 + (8,708 * max_connected_sessions)'. The value for "max_connected_sessions" is the same used for the server parameter of that name. See Chapter 6 for server parameter information.

For example, to configure shared memory for an installation with a single server, using the default maximum of 25 connected sessions.

Allow 8,192 bytes of shared memory for the installation segment.

Allow a minimum of 204,800 bytes of shared memory for the locking segment.

Calculate the shared memory for the server segment::

16,384 + (8,708 * 25) = 234,084 bytes of shared memory

Add:

8,192 + 204,800 + 234,084 = 447,076

The total shared memory requirement for this installation is a minimum of 447,076 bytes.

The minimum shared memory configuration possible is 238,084 bytes. This would be single server installation, configured for a maximum of one connected session.

Semaphore Requirements

Your ODT-DATA installation requires a semaphore set of 21 semaphores (5 + maximum_number_servers, which is a hard-wired limit of 16). In addition, each server requires a separate set of 10 semaphores. A minimum system configuration requires 31 semaphores.

Swap Space Requirements

A minimum of 16 megabytes of swap space is recommended for an ODT-DATA installation with one DBMS server. You need to add an additional 3 to 5 megabytes for each additional DBMS server, depending on the configuration of maximum connected sessions.

Environment Variables

The following environment variables are set during the ODT-DATA installation procedure. These are set to defaults by **iibuild** during the install process. In the standard installation the defaults are set for you. If you want a manual installation you need to decide what to set them to before running **iibuild**.

II_SYSTEM

In the standard installation the default location for your ODT-DATA installation is set for you. The environment variable II_SYSTEM is hard coded to */usr*. Before you begin a manual installation you need to determine the location for your ODT-DATA installation, so the environment variable II_SYSTEM is used by ODT-DATA programs to locate the DBMS installation. Before you begin the installation procedure, this variable must be set in the "ingres" user's environment to the full path name of the ODT-DATA installation directory.

Because II_SYSTEM is not set in the ODT-DATA symbol table, it must be set by users in their environment before they can run ODT-DATA.

Use the $II_SYSTEM directory only for ODT-DATA administration. Do not create user files or directories in the $II_SYSTEM directory or its subdirectories.

II_INSTALLATION

Your ODT-DATA installation, II_INSTALLATION, is identified by a unique two-letter identification code, which you select. Your code must be two alphanumeric characters such as "xx" or "x4", and the first character must be a letter.

II_LOG_FILE (Logging File Location)

ODT-DATA Release 6 uses an installation-wide logging file. This file handles all of the installation's concurrent ODT-DATA transactions. The logging facility writes pending transactions to the logging file, and the archiver facility moves completed transactions to the journal files when necessary. The full path name of the log file is *$II_LOG_FILE/ingres/log/ingres_log*.

The default value for II_LOG_FILE is the value of II_SYSTEM. Sites must determine the best place for this file to reside. Do not place the logging file on an I/O bound disk. Data acquisition times of the recovery and archiver routines will increase, slowing down all users on the ODT-DATA installation.

The size of the logging file is another important factor. The logging file must be large enough to handle all concurrent transactions without reaching saturation. It is designed as a circular file that wraps when the physical end-of-file is reached. If the logging file reaches the force-abort-limit, the oldest transactions are backed out dynamically. If it is not successful in freeing enough space and the log-full-limit is reached, the transaction system stops and backs out the oldest transactions. This could have severe performance implications and should be avoided.

The Raw Log File Option

The ODT-DATA log file can be configured as a "raw device." Utilizing a "raw device" for the log improves performance by:

- writing directly to disk, bypassing the UNIX cache
- writing larger disk blocks

If you plan to use a raw device, you must complete the following steps before running the **iibuild** installation script:

- The raw log device must be created as a character special device.
- The raw device cannot contain a filesystem.
- The device must be owned by the "ingres" user.
- The "root" user must run the *$II_SYSTEM/utility/mkrawlog* script to link the *ingres_log* file to the designated raw device file.

II_DATABASE

The II_DATABASE variable is set during ODT-DATA installation and may not be changed later, even during updates, so select your location carefully. The II_DATABASE environment variable points to the default location for database files. This environment variable also determines the location of the ODT-DATA master database, the **iidbdb**. Any database except the **iidbdb** may be created on alternate locations and moved later.

On multi-disk systems, II_DATABASE should not be set to a directory on an I/O-bound system disk because ODT-DATA database scans should not compete with system operations (such as system calls) for I/Os. Do not set II_DATABASE to a full filesystem, or to one that will become full as databases are added. Full disks become fragmented, and disk performance degrades.

II_CHECKPOINT and II_JOURNAL

The II_CHECKPOINT and II_JOURNAL variables are set during ODT-DATA installation and cannot be changed later, even during updates. II_CHECKPOINT is the environment variable set to the location iicheckpoint , where ODT-DATA checkpoints reside. II_JOURNAL is set to the location iijournal, where ODT-DATA journal files reside. Checkpoints are static backups of a database, while journals are dynamic records of changes made to a database since the last checkpoint. A checkpoint provides for recovery up to the time the checkpoint was taken. Checkpoints and journals provide for complete recovery up to the time of failure.

On single-disk systems, checkpointing to disk provides little safety. Disks usually crash in an all-or-nothing fashion. On single disk systems, checkpointing to magnetic tape is recommended. Storing data and backups on the same device provides little insurance from a disk failure. You can put journals and checkpoints on the same device. Journals are used in recovery only if the associated checkpoint is also available.

General Suggestions for Avoiding Problems

The *$II_SYSTEM/ingres* directory will usually be the home directory of the ODT-DATA system administrator, the "ingres" user. This account should be used only for ODT-DATA administration. No user files or directories should be placed in the $II_SYSTEM directory or its subdirectories.

The files in the $II_SYSTEM directory and subdirectories are critical to ODT-DATA. Deleting or changing any of these files may corrupt your installation.

ODT-DATA uses operating system permissions to protect your data. Never alter the permissions of any ODT-DATA file, directory, or subdirectory.

Chapter 4
Installing ODT-DATA

Read the preceding chapters before installing ODT-DATA. Installation requires some knowledge of UNIX. ODT-OS and TCP/IP must be installed before you can install ODT-DATA. After the ODT-DATA software is installed, it must be initialized. TCP/IP must be running before you initialize ODT-DATA. To initialize ODT-DATA automatically, log in as *ingres* and respond affirmatively to the initialization prompt.

Therefore, before using ODT-DATA, you must do the following:

1. Installl ODT-OS;
2. Install ODT-NET TCP/IP;
3. Install ODT-DATA;
4. Activate TCP/IP;
5. Initialize ODT-DATA by logging in to the system as *ingres* and responding affirmatively to the initialization prompt.

The next section presents the manual initialization, in which you must answer a series of prompts.

Manual Initialization

The automatic ODT-DATA initialization is accomplished by logging in as user **ingres** and responding **y** at the prompt for initialization. This automatic initialization uses the defaults described in the table below and you are not prompted for configuration information such as log file size and number of log buffers. To initialize ODT-DATA manually, you must respond **n** to the prompt for initialization. You should then run the **iibuild** command manually to customize ODT-DATA as desired. You can accept the default entries or enter your own values.

ODT-DATA Default Initialization Parameters	
Parameter	**Value**
Log file size 32 Kbyte blocks	128
Number of log buffers	4
Maximum number of databases in logging system	32
Maximum number of transactions in logging system	32
Block size of the log file	4 Kbytes
Log-full-limit	95%
Percent of log file for each consistency point	5%
Maximum consistency point interval for invoking archiver	19%
Force-abort-limit	80%
Size of locks hash table	63
Size of resources hash table	63
Maximum number of locks in locking system	2000
Maximum number of lock lists in locking system	128
Maximum number of locks allowed per transaction	150
Maximum number of server connections	50
Maximum number of open cursors per session	16
Maximum number of open databases per server	25
Per session stack size in bytes	32768

Initialization Checklist

You need to collect all the following information, before you begin the ODT-DATA initialization procedure. Please fill in the blanks on this checklist.

Does your UNIX system configuration meet the minimum ODT-DATA installation requirements?

238084 bytes or more of shared memory (y/n)_____

31 or more semaphores (y/n)_____

16 Megabytes or more of swap space (y/n) _____

If any of your answers are *no*, read Chapter 3 before continuing.

- *Your media device name is :*_____
- *Your two-letter ODT-DATA installation code is :*_____
- Read the configuration discussion in Chapter 3, and choose default locations for your databases, checkpoints, journals, and logging file.

  *For your databases, set II_DATABASE to :*_____

  *For your checkpoints, set II_CHECKPOINTS to :*_____

  *For your journals, set II_JOURNAL to :*_____

  *For your the logging file, set II_LOG_FILE to :*_____

- *Do you want the log file for this installation to be a raw device?* (y/n)_____

See Chapter 3 for instructions on creating a raw log device and for log file information.

Read the discussion in Chapter 6 of this book for an explanation of logging and locking facility configuration and primary DBMS server information.

Logging Parameters

*Log file size in 32Kb blocks :*_____

*Number of log buffers in memory :*_____

*Maximum number of databases :*_____

*Maximum number of transactions :*_____

*Block size of the log file :*_____

*Log-full-limit in percentage :*_____

*Percentage of log file for each consistency point :*_____

*Maximum number of consistency points written to invoke the archiver :*_____

*Maximum consistency point interval for invoking archiver :*_____

*Force-abort-limit in percentage :*_____

Locking Parameters

*Size of the locks hash table :*_____

*Size of the resources hash table :*_____

*Maximum number of locks in locking system :*_____

*Maximum number of lock lists in locking system :*_____

Maximum number of locks allowed per transaction : _____

See Chapter 6 of this book for an explanation of the options used in the startup of the DBMS server.

*Maximum number of server connections allowed :*_____

*Maximum number of open cursors per session :*_____

*Maximum number of open databases per server :*_____

*The per session stack size in bytes :*_____

Do you want this to be a "sole server" (required for fast commit)? (y/n)_____

Do you want the write_behind option set? (y/n)_____

Do you want to create a dblist for this server? (y/n)_____

Choose the UNIX editor to be used by ODT-DATA. The default is the current setting of the UNIX variable $EDITOR

*Set ING_EDIT to :*_____

Do you have ODT-DATA NET? (y/n)_____

Chapter 5
Maintenance Utilities

Use the maintenance and report utilities listed in this chapter to monitor and administer your ODT-DATA installation.

The Server (iidbms) Maintenance Utility—iimonitor

Database access using ODT-DATA Release 6 is controlled by the DBMS server. Use the **iimonitor** utility to examine the status of a server and the connected sessions. Use the utility to control the server's execution, including shutting down sessions and DBMS server. The administrative options such as stopping the server are restricted to an ODT-DATA superuser. This utility will:

- Display DBMS server information
- Display active sessions for the DBMS server
- Stop the DBMS server
- Display session information
- Disconnect a session
- Suspend a session

The II_DBMS_SERVER Environment Variable

Set the II_DBMS_SERVER environment variable to the communications address of the server you want to monitor with the **iimonitor** utility. To examine the global value of $II_DBMS_SERVER, type:

```
$ ingprenv | grep II_DBMS_SERVER
```

If a new server is created without specifying **-nopublic**, II_DBMS_SERVER is set to the new server's address. Connection to the old server is not possible without explicitly setting II_DBMS_SERVER to the old server's address. For example, to examine an old server with the communications address "1367", do the following:

C shell example:

```
% setenv II_DBMS_SERVER 1367
```

Bourne shell example:

```
$ II_DBMS_SERVER=1367
$ export II_DBMS_SERVER
```

The iimonitor Utility

The **iimonitor** utility allows you to examine the status of a server and the sessions connected to it. Administrative options, like stopping the server, are restricted to the ODT-DATA superuser. The utility can be used to control the server's execution, including shutting down sessions or the server itself. At the operating system prompt, to start the **iimonitor** utility, type:

```
$ iimonitor <server name>
```

At the "IIMONITOR >" prompt the following list of commands are available:

HELP	Lists the available commands.	
SHOW SERVER	Displays information about the server, including the number of sessions currently active or connected to it, the state of the server, and the CPU usage in terms of quantity used.	
SHOW SESSIONS	Displays a list of active sessions and their current states. The session states are:	
	CS_EVENT_WAIT	Waiting for an event. The event type is shown in parentheses: (LOCK), (DIO) - disk io, (BIO) - frontend communication.
	CS_MUTEX	Awaiting a semaphore (access to a system data structure).

	CS_COMPUTABLE	Runnable and waiting for a chance to run.
	CS_INTERRUPT	Interruptable, either a lock or frontend I/O request.

SET SERVER SHUT	Prevents additional server connections, and shuts the server down when the current sessions finish. This command can only be run by the ODT-DATA superuser.
STOP SERVER	Stops the server immediately. Use this command only if absolutely necessary, for example, when a frontend program is hanging. This command can only be accessed by the ODT-DATA super user.

The following commands use the session id to perform actions on a specific server session. The session id is displayed in the **iimonitor** utility with the **show sessions** command.

FORMAT *session id*	Gives a synopsis of the ODT-DATA information about a session.
REMOVE *session id*	Disconnects a particular session. This command can only be run by the ODT-DATA super user.
SUSPEND *session id*	Suspends a compute-bound session to allow technical support personnel to trace the problem.
QUIT	Terminates the IIMONITOR session.

Shared Memory and Semaphores Report Utility—csreport

Use the **csreport** utility to display shared memory and semaphore information for your installation. The **csreport** utility displays:

- the maximum number of servers configured for your installation
- shared memory and semaphore information

To invoke the **csreport** utility, type the following at the operating system prompt:

```
$  csreport
```

Output from the csreport Utility

The following is a example of output from **csreport**. The format is subject to change.

```
!Installation version 610008
!Max number of servers 16
!Description of shared memory for control system:
!key 0x49065A7A: size 8192 attach 00000000
!Description of shared memory for logging & locking system:
!key 0x49065A7C: size 327680 attach 0E000000
!Semaphore information for installation:
!sysV semid 0, num sems 21, used sems 19
! 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
!Event system: used space 2212, length space 8192
!Server info:
!server 0:
!inuse 0, pid 4917, connect id 0, id_number 0, semid 0
!shared memory:
!key 0xFFFFFFFF: size 0 attach 00000000
!server 1:
!inuse 1, pid 125, connect id 0, id_number 1, semid 0
!shared memory:
!key 0xFFFFFFFF: size 0 attach 00000000
!server 2:
!inuse 1, pid 128, connect id 0, id_number 2, semid 0
!shared memory:
```

```
!key 0xFFFFFFFF: size 0 attach 00000000
!server 3:
!inuse 0, pid 3770, connect id 2159, id_number 3, semid 12
!shared memory:
!key 0x49065A8E: size 122880 attach 00000000
!server 4:
!inuse 0, pid 360, connect id 0, id_number 4, semid 0
!shared memory:
!key 0xFFFFFFFF: size 0 attach 00000000
!server 5:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 6:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 7:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 8:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 9:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 10:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 11:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 12:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
```

```
!server 13:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 14:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
!server 15:
!inuse 0, pid 0, connect id 0, id_number 0, semid 0
!shared memory:
!key 0x00000000: size 0 attach 00000000
```

The Locking Facility Report—lockstat

Use the **lockstat** utility to display locking status information. The **lockstat** utility displays information about installation lock status.

To invoke the **lockstat** utility, at the operating system prompt, type:

```
$ lockstat
```

Output from the lockstat Utility

The following example is a typical output from **lockstat.**

```
==========Tue Mar 28 16:35:43 1989 Locking System Summary==========
        Create lock list          181          Release lock list          176
        Request lock             7520          Re-request lock            280
        Convert lock             9482          Release lock              7105
        Escalate                   11          Lock wait                    3
        Convert wait                0          Convert Deadlock             0
        Deadlock Search             3          Deadlock                     0
        Cancel                      0
----------------------------------------Locks by lock list----------------------------
Id: 00300011 Tran_id: 000000000000002E R_llb: 00000000 R_cnt: 0 Wait: 00000000   Locks:
(2,0/75) Status: NONPROTECT,NOINTERRUPT
        Id: 01B00009 Rsb: 00AD001D Gr: IX Req: IX State: GR PHYS(1) KEY(DATABASE,iidbdb)
        Id: 015F000D Rsb: 00350020 Gr: X Req: X State: GR PHYS(1)
KEY(PAGE,DB=00000001,TABLE=[1,0],PAGE=3)
Id: 0032000C Tran_id: 0000000000000030 R_llb: 00000000 R_cnt: 0 Wait: 00000000 Locks:
(0,0/75) Status: NONPROTECT,NOINTERRUPT
Id: 00330001 Tran_id: 0000000000000005 R_llb: 00000000 R_cnt: 0 Wait: 00000000 Locks:
(4,0/75) Status: NONPROTECT,NOINTERRUPT
        Id: 00390033 Rsb: 00380052 Gr: IX Req: IX State: GR PHYS(1)  KEY(SV_DATABASE,iidbdb)
        Id: 0155000D Rsb: 01970040 Gr: N Req: N State: GR PHYS(1)  KEY(DB_TBL_ID,iidbdb)
        Id: 01E80009 Rsb: 00910012 Gr: N Req: N State: GR PHYS(1)  KEY(OPEN_DB,iidbdb)
        Id: 01940009 Rsb: 01950009 Gr: IS Req: IS State: GR PHYS(1)
KEY(SV_PAGE,DB=00000001,TABLE=[1,0],PAGE=3)
Id: 00340003 Tran_id: 0000000000000004 R_llb: 00000000 R_cnt: 0 Wait: 00000000 Locks:
(0,0/75) Status: NONPROTECT
Id: 00350001 Tran_id: 0000000000000001 R_llb: 00000000 R_cnt: 0 Wait: 00000000 Locks:
(1,0/75) Status: NONPROTECT,MASTER,NOIN
        Id: 018C000C Rsb: 00910012 Gr: N Req: N State: GR PHYS(1)  KEY(OPEN_DB,iidbdb)
----------------------------------------Locks by resource----------------------------
Id: 00350020 Gr: X Conv: X Value: <0000000000000000> KEY(PAGE,DB=00000001,TABLE=[1,0],PAGE=3)
        Id: 015F000D Llb: 00300011 Gr: X Req: X State: GR PHYS(1)
Id: 00380052 Gr: IX Conv: IX Value: <0000000000000000> KEY(SV_DATABASE,iidbdb)
        Id: 00390033 Llb: 00330001 Gr: IX Req: IX State: GR PHYS(1)
Id: 00910012 Gr: N Conv: N Value: <0000000000000001> KEY(OPEN_DB,iidbdb)
        Id: 018C000C Llb: 00350001 Gr: N Req: N State: GR PHYS(1)
        Id: 01E80009 Llb: 00330001 Gr: N Req: N State: GR PHYS(1)
Id: 00AD001D Gr: IX Conv: IX Value: <0000000000000000> KEY(DATABASE,iidbdb)
        Id: 01B00009 Llb: 00300011 Gr: IX Req: IX State: GR PHYS(1)
Id: 01950009 Gr: IS Conv: IS Value: <0000000000000000>
Y(SV_PAGE,DB=00000001,TABLE=[1,0],PAGE=3)
        Id: 01940009 Llb: 00330001 Gr: IS Req: IS State: GR PHYS(1)
Id: 01970040 Gr: N Conv: IX Value: <0000000000000000> KEY(DB_TBL_ID,iidbdb)
        Id: 0155000D Llb: 00330001 Gr: N Req: N State: GR PHYS(1)
==========
```

The Logging Facility Report—logstat

The **logstat** utility displays information about:

- the force-abort-limit set during installation
- the percent of the log file used
- open transactions
- open databases

To invoke the **logstat** utility, at the operating system prompt, type:

```
$ logstat
```

Output From the logstat Utility

The following example is a typical output from **logstat.**

Logstat utility information

```
==13-JUL-1988 15:04:13.71 Logging System Summary============================
Database add                  387    Database removes       383
Transaction begins     5408          Transaction ends       5405
Log read i/o's                506    Log write i/o's        7196
Log writes                    18174  Log forces             5840
Log waits                     8005   Log splits             197
Log group commit       5808          Log group count        5827
Check commit timer     5820          Timer write            5787
Inconsistent db                 0    Kbytes written 10551
----Current log file header-----------------------------------------
Block size:4096               Block count:8000               Buffer count:1
CP interval:400               Archive interval:7600          Abort interval:6400
Last Transaction Id:0000009114A00P0B
Begin: 8946:1668:248  CP: 8946: 2070:168                     End:<8946:2298:96>
Journal Window:,0,0..,0,0
Status: ONLINE,CPDONE

----List of active processes----------------------------------------------------
ID      PID      TYPE    OPEN_   DB      WRITE   FORCE   WAIT  BEGIN        END
------------------------------------------------------------------------------
0010018 0000004F SLAVE   2       18121   5386    6380    5405  5404 65537
001001D 0000004C SLAVE   1        0       0       0       803   2    1  65537
001001E 0000004B MASTER  1       53       0       0       822   1    0  65537
```

```
----List of active databases----------------------------------------------------------
Id:FFFF0001      Database:($archiver, $ingres)     Status:NOTDB
        Tx_cnt:2 Begin:3  End:1    Read:0          Write:53 Force:0  Wait:1625
        Location:  None
        Journal Window:,0,0..,0,0<0,0,0>..<0,0,0>
        Id:002A0014      Database:(getto60,pixar)  Status:
        Tx_cnt: 1                 Begin:42 End:41   Read:0   Write:155    Force:57 Wait:63
        Location:        II_DATABASE:[INGRES.data.getto60]
        Journal Window:,0,0..,0,0<0,0,0>..<0,0,0>
        Id:01560015      Database:(company,pixar)   Status:
        Tx_cnt:0         Begin:3  End:3    Read:0          Write:3  Force:2  Wait:3
        Location:                 IIDATABASE:[INGRES.data.company]
        Journal Window:,0,0..,0,0<0,0,0>..<0,0,0>
----List of active transactions-------------------------------------------------------
Id:0F980016  Tran_id:0000009114A00495      Database:002A0014 Process:00010018
First:8496,4,96          Last:8496,11,2180         Cp:8495,7662,100
        Write: 11                 Split:0  Force:3  Wait:4
        Status: ACTIVE,PROTECT
Id:00010019  Tran_id:000000911149FEF81     Database:FFFF001 Process:0001001E
First:8495,7662, 1196    Last:8495,7662, 1456      Cp:8495,7260,836
        Write:53 Split:0  Force:0  Wait:822
        Status:INACTlVE
Id:0002001B  Tran_id:000000911149FEF82     Database:FFFF0001 Process:001001D
First:,0,0   Last:,0,0            Cp:8495,839,96
        Write:0  Split:0  Force:0  Wait:801
        Status:INACTIVE
================================================================================
```

Determining Proximity to FORCE-ABORT-LIMIT

To determine how close you are to reaching the FORCE-ABORT-LIMIT, refer to previous example, "Logstat utility information," for a computation example and consider the following:

- The first boldfaced number highlights the "Abort interval" of 6,400. This figure refers to the number of blocks in the log file that must be filled before the FORCE-ABORT-LIMIT is reached.
- The second highlighted number is part of a numeric series following the word "Begin". The important value here (1,668) refers to the block marking the log file's Beginning of File (BOF).
- The third highlighted number is part of a numeric series following the letters "CP". The important figure (2,070) refers to the block marking the last consistency point. This consistency point contains a list of all open transactions and open databases between the BOF (1,668) and the CP (2,070).

- The fourth highlighted number is part of a numeric series following the word "End". The important figure (2,298) refers to the block marking the log file's End of File (EOF). This is the last block that contains transaction information.

To calculate the number of blocks in use, subtract the EOF from the BOF (2298 - 1668). Currently, 402 blocks are in use in the log file.

To determine how close you are to the FORCE-ABORT-LIMIT, subtract the total blocks used (402) from the abort interval (6400):

```
6400 - 402 = 5998
```

This log file still has plenty of room before reaching the FORCE-ABORT-LIMIT.

Also highlighted in this section is an area called "Status," which, in this case, states: "ONLINE, CPDONE." The status provided here is of the logging and recovery systems. **ONLINE** indicates that everything is fine. **CPDONE** indicates that a consistency point was taken and the status is fine. Other options that might appear as the status are:

cpneeded	The logging system is about to take a consistency point.
logfull	The log file is full and transactions will stall until the archiver process catches up.
force_ abort	The FORCE-ABORT-LIMIT has been reached; the oldest open transaction will be aborted.
recover	The recovery process is performing recovery.
archive	The archiver process is archiving journaled transactions to the journal files.
acp_ shutdown	The archiver is preparing to shutdown. (This status is displayed when you type "rcpconfig/shutdown".)

imm_ shutdown	The logging system has been told to shut down immediately. (This is displayed when you type "rcpconfig/imm_shutdown.")
start_ archiver	This is an important status indicating that the archiver has died and must be restarted by the DBA. The archiver will not restart automatically; if the DBA does not restart it, the log file will eventually reach full capacity.
purgedb	This status appears when a database has been closed by the last user who had it open; the archiver is archiving transactions that belong to this database.

Notifying Users of Imminent Shutdown

To shut down the system you must determine what databases are active and who the users are so that you can notify them of the imminent shutdown. In referring to the example of the logstat report displayed a few pages ago you can see the following.

The first database listed will always be owned by **$ingres**. In this case, **$archiver** is displayed as the database, but the status on this line indicates that **$archiver** is not a database. This entry shows that ODT-DATA is operating correctly.

The second database shown is "getto60", owned by "pixar". The ID for this database is 002A0014. Refer to the section entitled "List of active transactions". You will see that the first transaction listed belongs to "Database: 002A0014". The status of this database (getto60) is ACTIVE.

The third database shown is called "company,pixar", but since its ID number does not appear on the list of active transactions, "company" is not currently active. Both other transactions shown in this list are also inactive and belong to **$ingres**.

When EOF Is Less Than BOF

Because the log file is circular, it is not uncommon for the block marking the file's beginning to be a higher number than the block marking the file's end. The following example illustrates the "Current log file header" from the results of another **logstat** execution.

```
------Current log file header-----------------------------------------------------------
Block size: 4096                Block count: 8000               Buffer count: 1
CP interval:400                         Archive interval: 7600  Abort interval: 6400
Last Transaction Id:            0000009114A0049F
Begin: 8495:7662:100            Cp:8495:7662:100                End:8496:16:96
Journal Window: ,0,0..,0,0
Status:                                 ONLINE, CP DONE
------List of active processes---------------------------------------------------------
```

- The first boldfaced number highlights the "Block count" of 8,000, the number of 4K blocks in the log file.
- The "Abort interval" here is 6,400, as it was in the previous example.
- The log file's Beginning of File (BOF) is 7,662.
- The consistency point (CP) is the same as the BOF (7,662). The amount of space occupied in the log file by open transactions since the last consistency point has not reached the five-percent threshold. A new consistency point is taken at the five percent threshold.
- The End of File (16) is smaller than the BOF (7,662). Such a discrepancy is possible because the log file is circular. To calculate the number of blocks in use, subtract the BOF from the block count (8,000), then add the EOF to the total, as in the following example:

  (8000 - 7662) + 16 = 354

  To determine how close you are to the FORCE-ABORT-LIMIT, subtract the total blocks used from the abort interval:

  ```
  6400 - 354 = 6046
  ```

Chapter 6
Installation Reference Material

This chapter provides reference information for installing ODT-DATA and for the server start up facility.

ODT-DATA Installation and Server Start Up Utility

A new server installation can be started by invoking **iistartup** along with one of the following flags from the operating system prompt:

-b $II_SYSTEM

To start an installation automatically each time the system is rebooted, include the command **iistartup -b $II_SYSTEM** in the UNIX system startup file. This is */etc/rc* or */etc/rc.local* at most sites. The server is started using the parameters saved in the file *rundbms.opt* if it exists; otherwise the default parameters will be used. This flag may not be used from the operating system prompt.

-i

The command **iistartup -i** can be used from the operating system prompt to start up a new server installation.The user is given the option to use the parameters saved in the *rundbms.opt* file, use default parameters, or to reconfigure *rundbms.opt*.

-n

The command **iistartup -n** can be used from the operating system prompt to start a new server or installation.The user is given the option to use the parameters saved in the *rundbms.opt* file or the default parameters, or reconfigure *rundbms.opt*.

iirundbms and DBMS Server Parameters

Access to a Release 6 database is controlled by the server (iidbms). **iistartup** invokes **iirundbms** to configure the server using parameters saved in the file *$II_SYSTEM/ingres/files/rundbms.opt*. If the server configuration option is chosen, the user is prompted with a selected list of **iirundbms** parameters that will be written to **rundbms.opt** and used for server startup. If parameters other than those prompted are desired,, edit **iirundbms** with the required parameter list.

DBMS Server Parameters

The following **iirundbms** parameters are recognized:

-connected_sessions *n*
: The maximum number of server connections allowed. The default is 32, and *n* may not exceed 50.

-cursors_per_session *n*
: The number of simultaneously open cursors per session. The default is 16.

-dmf.option *n*
: The Data Manipulation Facility (DMF)manages the interface between the DBMS and stored data. The following **dmf** options are available:

  cache_size
  : The number of individual buffers (single data pages) in the buffer manager. The default is (128 + 4 *sessions).

  count_read_ahead
  : The number of group buffers (multiple data pages) in the buffer manager. The default is (4 + sessions/4).

  size_read_ahead
  : The size of the group buffers used in READ_AHEAD operations. The default is 8.

  tcb_hash
  : The size of TCB hash table used in lookups of table control blocks. Use values of power to 2 for optimal hashing operation. The default is 255.

wbstart Specifies the threshold for modified pages in the cache, after which write-behind starts. When there are wbstart-modified pages in the cache, the *write behind* threads wake-up and write-modified pages out of the cache until the number of modified pages drops below the **wbend** limit.The default is (**mlimit** - 15% of the cache size).

wbend Specifies the lower limit for modified pages. When it is reached the *write behind* threads go to sleep and wait for the **wbstart** limit to be reached. The default is(1/2 of the buffer manager (cache) size). If 1/2 of the cache is not less than **wbstart** then, the size defaults to (1/2**wbstart**).

flimit Specifies the minimum number of free pages that the buffer manager will try to keep available in the cache. When this limit is reached, the buffer manager begins doing synchronous writes of pages whenever needed. The default is (1/32 of the buffer manager (cache) size).

mlimit Specifies the maximum number of modified pages that can be left in the buffer manager. When this limit is reached, the buffer manager begins writing pages out of the cache each time a dirty page is unfixed. **Mlimit** must be greater than **wbstart**, if **wbstart** is specified and **mlimit** is not, then **mlimit** will default to halfway between **wbstart** and the size of the cache. The default is(3/4 of the buffer manager (cache) size).

memory Specifies maximum memory usage by the DMF for control blocks and caching. Calculate requirements based off total cache size (single and group buffers) and shade enough for control blocks. Keep in mind that each cache buffer requires 2K (ODT-DATA pages). The default is (500K, Value input * 1024).

-database _count *n*
The maximum number of open databases for the server. The default is the value of the **connected_session parameter.**

-dblist *dbname {dbname}*
Allows the specification of a list of databases that will be serviced by the DBMS server. By default all databases can be serviced by any DBMS server running in an installation. If you use the option **-dblist** *dbname* and the **-sole_server** option, that server will be the only one that can access the database *dbname*.

-fast _commit
Transaction I/O optimized by using a delayed write algorithm. Used only with **-sole_server** option.

-no public
Creates a private server. To access a private server, the environment variable II_DBMS_SERVER must be set to the value reported at startup. The global value of II_DBMS_SERVER is not updated when this option is used.

-qef.option *n*
The Query Execution Facility (QEF) manages query plans and execution. The following **qef** option is available:

qep_size
Translates to the QEF data size to be used to calculate the maximum memory that QEF will use to build data segment headers, containing all the runtime information needed for the query to be executed. DSH_MAXMEM = 2048 + ((sessions *cursors) * qep_size)

-qsf *n*
The Query Storage Facility (QSF) manages shared memory between facilities. The following QSF option is available:

pool_size
Increases or resets the QSF memory pool by allocating *n* bytes of memory to it. The pool is used to store all the query objects, regardless of type. It serves as the facility that manages a session's memory use. The default size for the QSF pool is ((number of connected sessions * 40K)+ 60K).

-scf.option *n*
The System Control Facility (SCF) is the central controlling facility. The following SCF option is available:

row _estimate	This is the number of rows to expect on the first pass of a select or **retrieve**. This is used to build message blocks to send MDEs (message data elements) across GCAto frontends. This reduces the amount of formatting done before queries are executed. If insufficient buffers to send the data from the query are formatted, the additional buffers needed are built after the query completes.
-sole_server	Forces databases accessed by this server to refuse connection requests from other servers. This option should be used when possible to reduce the overhead in managing multiple server connections to common databases.
-stack_size *n*	The per-session stack size in bytes. The default is 32,768 bytes.
-write _behind *n*	Allows the creation of write behind threads in a DBMS server. *Write behind* threads write dirty pages out of the buffer manager when the cache starts getting full. This keeps users from having to do synchronous writes to free up space in the cache to read in a new page. They also wake up when consistency points are taken to help the *fast commit* thread flush all the dirty pages out of the cache. There are no rules to determine the optimal number of *write behind* threads to allocate and there is no way to dynamically add new *write behind* threads to the server once it is invoked. As a guideline, look at the load on the system and determine the number of writes per second that need to be done (figuring in the number of devices to be written to). Each *write behind* thread can perform approximately 20 I/Os per second up to the capacity of the operating system. The value for *n* must be some number greater than 0.

The following options control UNIX process parameters:

-maximum_working_set *n*	Sets the server's resident size (RLIMIT_RSS)in bytes.
-priority *priority*	Adjusts the server process priority (PRIO_PROCESS).

NOTE: These options may be abbreviated to the minimal unique prefix. For example, **-con** can replace **-connected_sessions.**

The DBMS Server Process Communications Address

The **iirundbms** command sets the *installation-wide* ODT-DATA environment variable II_DBMS_SERVER to the communications address of the server process. The communications address is an Internet socket number. Users can connect frontend programs to specific servers by setting the II_DBMS_SERVER, to the desired server's address, in their environment. Incorrectly setting the II_DBMS_SERVER environment variable will generate the following errors:

```
E_LC0001 GCA protocol service (GCA_REQUEST)
     failure with status E_GCfe05
     Connect failed

E_LQ0001 Failed to connect to DBMS session
```

The II_DBMS_SERVER Environment Variable

If you start additional servers with **iistartup** or **iirundbms** without specifying the **-nopublic** option, the value in II_DBMS_SERVER will be replaced with the new server's communications address. Then, connection to the old server will only be possible by explicitly setting II_DBMS_SERVER to the former value. To see the value of II_DBMS_SERVER for the installation, type:

```
$ ingprenv | grep II_DBMS_SERVER
```

ingprenv prints the names and values of all the ODT-DATA environment variables for an installation, including a single value for II_DBMS_SERVER.

WARNING: No record is kept of the previous value of II_DBMS_SERVER. Consult the ODT-DATA error log for old values of II_DBMS_SERVER.

DBMS Server Startup Troubleshooting

Check these items if the server fails to start:

- Are system shared memory and semaphore resources installed?
- Does the log file ($II_LOG_FILE/ingres/log/ingres_log) exist?
- Is the recovery process "dmfrcp" running?
- Is the archiver process "dmfacp" running?
- Is II_DBMS_SERVER set to the current server's communications address?
- Is the local node in the **/etc/hosts** file?

Logging and Locking Facility Parameters—rcpconfig

If the logging and locking configuration option of **iistartup** is chosen, **rcpconfig** *-init* is invoked to configure and initialize the log file. The user is prompted to enter logging and locking parameter values that will be written to **$II_SYSTEM/ingres/files/rcp.par** and used for configuration.The following sections contain descriptions of the **rcpconfig** parameters whose values must be set.

The rcpconfig Logging Parameters

ODT-DATA Release 6 builds a single circular log file per installation. The log file contains records used in aborted or incomplete transactions. It also contains records of completed transactions. These are taken by the archiver and placed in the corresponding journal files. The name of this file is *ingres_log*. It is located in the directory defined during installation by the environment variable II_LOG_FILE. This section contains information about the parameters used to configure logging.

- Number of log buffers in the memory (Default is 4.)

  This is the number of outstanding I/Os waiting to be put to the log file. These buffers are sized by the block size prompt,which follows. Sites with small transaction volumes may increase the number of log buffers and decrease the transfer block size,increasing throughput to the disk. Large transaction volume sites using larger block sizes and fewer log buffers log data faster.

- The maximum number of databases in logging system (Default is 32.)

  This is the maximum number of open databases that the logging system can handle at one time. The high side is the safe side for this value so that an unexpected database access attempt is not halted by a lack of available slots. This can also be used as a lockout method to prevent users from accessing additional databases. Gauge your answer accordingly.

- Maximum number of transactions in logging system. (Default is 32.)

  This is the maximum number of current (pending) transactions that can be handled by the logging system. Figure this value based on the amount of concurrent ODT-DATA processes on the system, servers(to include the recovery and archiver), and unique server-database connections. Gauge this value on the high side so that your system is able to start new transactions without making users wait until a slot becomes available.

- Block size of the log file. The legal block size is 4, 8, 16, or 32kbytes. (Default is 4)

  The log file is broken down into blocks which are used to transfer logging data (transaction information) from the log buffers in memory to the logging file on disk. This value is the block size of that unit of transfer per I/O. Sites processing large transaction volumes should use larger values, accomplishing the most throughput with fewer system actions.

- Log-full-limit in percentage. (Default is 95%.)

  Once the logging file reaches a certain percent of usage, the logger halts and backs out the oldest one and any that it finds at that same time stamp. Once this is accomplished, transaction processing begins again, on the assumption that available space in the file has been reclaimed. This is hard stopping point that prevents further transactions from being processed until sufficient logging files are cleared. This contrasts with the force abort limit which usually prevents this point from being reached. Large logging files should use the default value or higher since small values increase the likelihood large transactions will not proceed to completion.

- Percentage of log file for each consistency point. (Default is 5.)

  How often consistency points are written to the log file is determined by this value. The larger the logging file, the smaller the percentage should be, as this insures that there start time is not prohibitively long during the recovery process. This percentage can be set from 1 to 75.

- Enter the maximum consistent point interval for invoking archiver (Default is 19%.)

  The logging system uses consistency points to keep track of all active databases, transactions, and lock lists at certain intervals in the log file. This decreases the time required to restart the system after a crash, by finding the latest consistency point and resuming processing from there. This value tells the archiver that a certain number of consistency points have been written and it is time to wake up and begin archiving applicable data involved in that range. For example, if each consistency point involves five percent of the log file, a value of four here would wake the archiver each time twenty percent of the log file is available to be archived.

- Force-abort-limit in percentage. (Default is 80%)

  This soft failure point causes the oldest pending transactions to be aborted, preventing the log file from reaching the log full limit and causing a halt of all transaction processing until available log file space has been freed. Do not make this value too close to the log full limit value, or else the value of this parameter will be severely reduced because of the high probability that the log full limit will be reached anyway.

The rcpconfig Locking Parameters

This section contains information about parameters used to configure the shared memory locking.

- The size of the locks hash table (Default is 63.)

  In the ODT-DATA lock manager, there are two types of lock lookups. The first type uses the lock hash table to locate information about a lock owned by a specific user. This value creates the size of this lock lookup table used to determine the state of a given lock on a given resource. From this table the lock block is located and the associated resource block can be examined. Make this value greater than or equal to the resource hash table, because many types of locks can be queued on the same resource, but the same lock cannot refer to more than one resource.

- Size of the resources hash table (Default is 63).

  The second type of lock lookup is performed directly on the resource being locked. Through this table, information about specific resources is located and the associated locks on these resources can be determined. This value sets the size of this hashed lookup table.

- Maximum number of locks in locking system. (Default is 2000.)

  For transactions to process to completion and database access to be granted, there must be locks available in the ODT-DATA lock manager. This number should be a sum of all resources and locks required on the ODT-DATA system. Currently, ODT-DATA uses approximately 100-120 locks per user. Judge your answer according to your concurrency requirements. Sites that use very large transactions, with high max locks values, should set this value on the high side to insure that the ODT-DATA lock manager does not run out of locks. This will force a reconfiguring of the lock manager and its associated data structures.

- The maximum number of lock lists in locking system (Default is 128.)

  Lock lists are maintained on current transactions to speed processing and assure that MSTs are handled correctly. Locks accumulated by a pending transaction are chained together to help the transaction manager locate locking information required for completion of the task. There should be two lock lists per active user and five per server (to include the recovery and archiver processes).

- Maximum number of locks per transaction (Default is 150.)

  So that one transaction does not use all system locks or create an environment in which lock management overhead reduces performance, this parameter is required to place a cap on the number of locks a particular single statement or multi-statement transaction can own. Once this value is reached, escalation to table-level, locking takes place, reducing the number of locks taken by the transaction and letting the transaction proceed more quickly.

The rcpconfig Archiver and Recovery Shutdown Parameters

The **rcpconfig** command will shut down the recovery and archiver processes and deinstall shared memory when used with the following two options.

-shutdown	This will refuse all further connections and transaction processing, but allow those currently executing to finish and then execute a clean shutdown of the recovery and archiver processes. This is the friendly way to close down the recovery process and allow pending transactions to complete.
-imm _shutdown	This executes an immediate stop on all pending transactions and shuts down the archival and recovery processes. This should be used only when there is a critical need to close down the ODT-DATA recovery system that cannot be delayed until pending transactions finish.

The **rcpconfig** command with the **-shutdown** option is used by **shutserver** during installation shutdown.

ODT-DATA Installation Shutdown

The following describes **shutserver**, an automated shutdown utility and an emergency manual shutdown procedure.

The Installation Shutdown—shutserver

The shutdown utility can only be invoked by the superuser. From the operating system prompt type:

```
$ shutserver
```

You are then prompted through the steps to shut down your installation or sections of it.

Emergency Manual Installation Shutdown Procedure

The following describes an emergency procedure to shut down an ODT-DATA installation. Normally the server (**iidbms**), archiver (**dmfacp**), and recovery (**dmprcp**) processes should only be terminated with the **shutserver, iimonitor, and rcpconfig** utilities. If these utilities fail or hang, you may need to stop these processes using the UNIX **kill** command.

1. Log in as the "ingres" user.

2. Identify the ODT-DATA installation code of the installation you want to shut down by typing:

```
$ ingprenv | grep II_INSTALLATION
```

   The two-letter installation code will be displayed:

```
II_INSTALLATION=r6
```

3. Identify the installation's server(s) and their UNIX process id(s):

```
$ ps -e | grep iidbms
```

   The UNIX process id number and the installation code of the server(s) will be among the information displayed.

4. If a server has the UNIX process id "1912" and the correct installation code, shut it down using **kill** with the SIGQUIT signal:

```
$ kill  -QUIT  1912
```

   The following list of UNIX signals and expected effects is for information only. **SIGQUIT** is the preferred signal to use.

SIGHUP	Terminates the server if there are no active sessions.
SIGTERM	Terminates the server when all sessions finish.
SIGQUIT	Terminates the server immediately.

SIGKILL Terminates server process abnormally. Run **cscleanup**.

5. Once the server processes associated with the installation are stopped, the recovery and archiver processes can be killed. To kill the processes manually, identify their UNIX process ids and verify that they are the correct processes by their installation code:

```
$  ps -e | grep  dmfrcp
$  ps -e | grep  dmfacp
```

   The UNIX process id number and the installation code of the recovery and archiver processes are among the information displayed.

6. If **dmfrcp** has process id "10469", **dmfacp** has process id "10471", and they have the correct installation code, then **kill** them using the SIGQUIT signal:

```
$  kill -QUIT  10469
$  kill -QUIT  10471
```

NOTE: If a server process was terminated abnormally with SIGKILL instead of SIGQUIT, **cscleanup** should be run. This program will attempt to release global system resources the server might have owned, such as shared memory and semaphores. The **csinstall** command should not be run again.

Once these steps are completed, the installation is completely shut down. Restart it using the command; **iistartup -n**.

WARNING
If the installation cannot be restarted from this point, you may have to shut down again and reinitialize the log file. This should not be done except as a last resort, as it will interfere with the recovery process.

Chapter 7
Troubleshooting with Log Files

This chapter explains how to use the ODT-DATA log files to troubleshoot ODT-DATA.

ODT-DATA Log Files

ODT-DATA creates log files where it writes information about your installation. The files described in this chapter are English text log files that you can use for troubleshooting.

NOTE: See Chapter 3 for information on the logging file associated with the logging and locking facility.

The Error Log

When you have an ODT-DATA problem, check the file *$II_SYSTEM/ingres/files/errlog.log*. Messages about your installation are appended to this log along with their dates and times. You find the following information in *errlog.log*:

- Archiver shutdown
- DBMS server startup and shutdown
- Error messages
- Warning messages

The *errlog.log* file is maintained by the system administrator. The *errlog.log* file will continue to grow until you shut down the installation and manually truncate the log.

The Archiver (dmfacp) Log

The file *$II_SYSTEM/ingres/files/II_ACP.LOG* is overwritten each time the archiver process is started up. ODT-DATA writes information about the current archiver process in this log, such as:

- Archiver startup
- Error messages
- Warning messages

The Recovery (dmfrcp) Log

The file *$II_SYSTEM/ingres/files/IIRCP.LOG* is overwritten each time the recovery process is started up. ODT-DATA writes information about the current recovery process in this log such as:

- Current logging and locking parameter values
- Error messages
- Recovery operations information
- Warning messages

The Installation (iibuild) Logs

During the ODT-DATA installation procedure, error messages and sometimes startup, and shutdown messages are saved in log files named for the processes whose output they store. The following log files are located in the directory *$II_SYSTEM/ingres/files*:

- **iigcn.log**
- **rcpconfig.log**
- **rundbms.log**

Appendix A
ODT-DATA Startup Files

When you invoke the **sql** command, the terminal monitor can read up to three different startup files. These can contain **sql** commands and macro definitions. Startup files can be installation-wide, database-specific or user-specific depending upon how they are invoked.

Installation-Wide Startup Files

The system startup file is automatically read when ODT-DATA is invoked with the terminal monitor. This file is included with your ODT-DATA installation and can be customized by the ODT-DATA system administrator to meet the specific requirements of your site.

The file *$II_SYSTEM/ingres/files/startsql* can be edited by the ODT-DATA system administrator to include SQL commands that are executed each time the terminal monitor is invoked.

For example, if the standard editor at your installation is not **/usr/bin/vi**, the **macro** {editor} can be redefined in the SQL **startup** file to invoke the proper editor. For more information about SQL terminal monitor macros, refer to the optional *ODT-DATA SQL Reference Manual.*

Database-Specific Startup File

Database-specific environment variables can be set to contain the full pathname of a system startup file. The file is read when the ODT-DATA terminal monitor is invoked for a particular database.

The environment variable DBNAME_SQL_INIT can be set to the full pathname of a file containing SQL commands. The filename is specified by the user. For DBNAME, substitute the name of the database for which the file is to be read. The database name must be specified in uppercase.

The ODT-DATA user can set these environment variables installation-wide by using the **ingsetenv** command as follows:

```
$  ingsetenv DBNAME_SQL_INIT path_name
```

The definitions for environment variables set with the **ingsetenv** command are stored in the file *$II_SYSTEM/files/symbol.tbl*. Type the command **ingprenv** to see the ODT-DATA environment variables and their values.

User-Specific Startup File

These definitions can also be set locally as user-specific environment variables. A good place to set them locally is in the user's ".login" or ".profile" file.

C shell example:

```
setenv DBNAME_SQL_INIT path_name
```

Bourne shell example:

```
DBNAME_SQL_INIT=path_name
export DBNAME_SQL_INIT
```

The environment variable II_SQL_INIT can be set to the full pathname of a file containing SQL commands. The file name is specified by the user.

A good place to set user-specific environment variables is in the user's .login or .profile file.

C shell example:

```
setenv II_SQL_INIT path_name
```

Bourne shell example:

```
II_SQL_INIT=path_name
export II_SQL_INIT
```

Appendix B

Authorizing User Access to ODT-DATA and Databases

ODT-DATA maintains a special "database" named **iidbdb**. It is used to store information about databases and authorized ODT-DATA users, and specifies the databases that can be accessed by specific users. The information in the **iidbdb** is updated by ODT-DATA when a database is created or destroyed. The **iidbdb** is consulted to validate a user's request to use ODT-DATA, to validate a user's request to use a particular database, and to determine where a particular database is stored. The **iidbdb** is itself an ODT-DATA database owned by the ODT-DATA system administrator.

The information in the **iidbdb** regarding the databases and where they are located is automatically maintained by ODT-DATA. This information is updated by the **createdb** and **destroydb** commands. This information is further affected by commands that relocate tables in a database into different directories or filesystems.

The information in the **iidbdb** about authorized ODT-DATA users and the databases they can access is maintained by the ODT-DATA system administrator with the **accessdb** program, described in this chapter.

ODT-DATA users other than the ODT-DATA system administrator can use the **catalogdb** command to view this information. This command is described in the optional *ODT-DATA SQL Reference Manual.*

Database Access

For user Bob to access database "xyz," at least one of the following conditions must be true:

- Bob is the data base administrator (DBA) for "xyz."
- "xyz" is globally accessible.

- Bob has been explicitly authorized by the xyz DBA to use database "xyz."
- Bob has the ODT-DATA superuser flag set in his user entry, and uses the **-u** (user) flag on the command line

By default, databases are globally accessible when they are created. The **-p** (private) flag of the **createdb** command must be used to create a private database. Operations such as changing the global access status of a database, extending a database to different locations, or authorizing a particular user to access a specific private database, may be performed by the ODT-DATA system administrator using the **accessdb** command. User authorization for database access is accomplished by adding the user to the table of users authorized to access the database, or by adding the database to the list of databases the user is authorized to access.

Database access does not automatically authorize access to tables in the database. Only the DBA for that database may authorize user access to shared tables with the SQL **create permit** command.

Defining the Terminal

If you are running in ODT-DATA/WINDOWVIEW the *termcap* variable is set.

The **accessdb** command must be run on a cursor-addressable terminal whose description is contained in the *$II_SYSTEM/ingres/files/termcap* file. The environment variable TERM (or TERM_INGRES) sets the terminal definition for use by the forms products. For example, if you want to abandon ODT-DATA/WINDOWVIEW system and use your terminal with function keys active in the C shell environment, you can identify your terminal, in your *.login* or *.cshrc* file, to **accessdb** by typing the following command:

```
setenv TERM_INGRES ansi
```

To identify your terminal in the Bourne shell environment, in your profile file, type the following commands:

```
TERM_INGRES=ansi
export TERM_INGRES
```

You can specify any of the valid terminal codes listed in the optional manual *Using ODT-DATA Through Forms and Menus.*

Invoking accessdb

After you define your terminal, start up **accessdb** by typing the following command at the operating system level:

```
$     accessdb
```

Remember that only the ODT-DATA System Administrator, "root" and other accounts you set up in the **iidbdb** with ODT-DATA superuser privilege are allowed to use **accessdb**.

Using accessdb

Because **accessdb** is a forms-based program, it is important to understand how to enter data into forms and select menu items before using it. If you are unfamiliar with the ODT-DATA forms products, start out using QBF™ or VIFRED™ to become familiar with forms applications. Refer to the book *Using ODT-DATA* in the User's Guide for more information.

When **accessdb** starts up, it displays a main menu of commands. The process of using **accessdb** consists of selecting a command from this main menu, moving on to another form or menu, optionally selecting another command, and exiting by selecting the **Quit** command. The execution of commands causes **accessdb** to display a new form with menu items appropriate to the function chosen. To return to the main menu from any form, enter the **End** menu item. Other menu items may also return you to the main menu after performing their sub-function.

All menus within **accessdb** contain an entry for **Help**. If you are in doubt about the meaning of a form that you have selected, select the **Help** menu item to get a quick reminder.

Functions in accessdb

The main menu in **accessdb** contains the following commands:

Command	Function
Catalog	Submenu of additional operations.
Database	Summarize information about a database
ExtendDB	Extend a database to a new volume.
LocationName	Create/Examine a *locationname*-area mapping.
User	Create/Modify/Delete an ODT-DATA user or specify the databases a user may access.
Help	Print help about the top level menu.
Quit	Exit from the **accessdb** program.

Database Function

The **Database** command is used to view and update information about a single database. When the **Database** command is selected from the main menu, you are prompted to enter an existing database. A form describing the database is displayed or an error message is generated if the database does not exist. To return to the main menu, select the **End** menu item. Help is available by selecting the **Help** menu item. The information displayed on the form includes the fields listed in the following table.

NOTE: Only the user table and the global access flag may be updated using the **Database** command.

Field Name	Mode	Description
Database name	read	Name of database.
Owner	read	Owner name.
Database location	read	*Locationname* upon which the ODT-DATA database resides (from **createdb**).

Field Name	Mode	Description
Checkpoint location	read	*Locationname* upon which ODT-DATA checkpoints reside (from **createdb**).
Journal location	read	*Locationname* upon which ODT-DATA journal files reside (from **createdb**).
Global access	read/write	Set "no" for private databases, and "yes" for globally accessible databases.
Database users	read/write	For private databases, a list of users explicitly authorized to access database. This is a table field, which can be edited. To delete a user, move to the row and press **Return**. To add another user, move to an empty row and enter the new name.

The **Database** form allows you to change both the global access status of a database and its list of explicitly authorized users. To do so, once the database form is on the screen, edit the form to reflect the changes, then select the **Save** menu item. To void the changes you made, select the **End** menu item instead.

ExtendDB Function

The ExtendDB function is used to extend a database to other locations. To locate tables outside the default area or the area corresponding to the *locationname* specified on the **createdb** command line, the database must be extended to a previously defined location.

ODT-DATA Locationnames

Locationnames are labels that are mapped to a UNIX directory. Each *locationname* maps to a single area. The area is described with the full path name of a directory. *Locationnames* are chosen by their creator and must follow the ODT-DATA naming convention. They must be alphanumeric and begin with a letter. The maximum length of a *locationname* is 32 characters. The area designation can be up to 240 characters including the "/" character and must follow the UNIX syntax for directory names.

Locationnames are used by the **createdb** and **finddbs** utilities. If a *locationname* is not specified the default *locationname* is used.

Each ODT-DATA installation has a set of default *locationnames*. These defaults are ii_database, ii_journal and ii_checkpoint. The *locationnames* map to the ODT-DATA environment variables II_DATABASE, II_JOURNAL, and II_CHECKPOINT, respectively. These defaults are mapped to the areas these environment variables are set to during the installation procedure. They *cannot* be changed.

To Extend a Database

Select the **ExtendDB** menu item from the main menu. Enter the login of the DBA of the database to be extended. A form then displays two tables: one listing existing *locationnames* and database names, and one in which you can add new database extensions. The names currently in the first table are those databases that have been extended to the specified *locationname*. One database may be extended to many locations.

To extend a database, both the database and *locationname* must exist. The database must be owned by the specified DBA, and the corresponding *locationname* must be available for databases as defined in the **LocationName** menu item in the main menu. To extend a database to a location, move the cursor to the table of new database locations, and enter the database name and the corresponding *locationname* in the proper columns. When you are done extending databases for a particular DBA, enter the **Save** menu item. The data in the table is then checked. If any of the data in the table are invalid, or if a database extension or restriction is not allowed, the cursor is positioned on the row containing the offending data. You can then correct the data and reenter the **Save** menu item. When the changes are accepted, you are returned to the main menu.

LocationName Function

The **LocationName** function is used to view or add the *locationnames*. Each execution of the **LocationName** function operates on a single *locationname*.

When the **LocationName** function is selected from the main menu, the system prompts you for a *locationname*. Enter a new (i.e., undefined on the system) or existing name. If you enter an existing name to view, a form appears displaying the current attributes of the *locationname*. If the *locationname* being entered is new, you are prompted to create a new location. If the answer is "yes" a blank form appears with the default values filled in. Fill in the appropriate data fields on the form.

Adding a New Locationname

The new *locationname* appears on the top of the form. A table-field displays the databases currently using that location. (This should be empty now.) Fill in the "area" field with the full pathname for the new area *locationname*. Before database, journal, or checkpoint usage permission can be granted for a **locationname**, the corresponding ODT-DATA directory and subdirectories must exist and be accessible by ODT-DATA. Fill in the usage permissions. The privileges are as follows:

- **Data Bases** allows database relations to reside on the specified *locationname*. The default is "yes."
- **Journals** allows journals to reside on the specified *locationname*. The default is "no."
- **Check Pts** allows checkpoints to reside on the specified *locationname*. The default is "no."

Select or create a directory to become the new ODT-DATA area. The directory must have the following permissions:

- read, write, and execute for "owner"
- read and execute for "group"
- read and execute for "world"

Create a subdirectory named "ingres." This directory must be owned by "ingres" and have the following permissions:

- read, write, and execute for "owner"
- read and execute for "group"
- read and execute for "world"

Create a subdirectory named "data," "jnl" or "ckp." This directory must be owned by "ingres" and have the following permissions:

- read, write, and execute for "owner"
- no permission for "group"
- no permission for "world"

Create a subdirectory named "default." This directory must be owned by "ingres" and have the following permissions:

- read, write, and execute for "owner"
- read, write, and execute for "group"
- read, write, and execute for "world"

To create the subdirectories needed for data in the area /install/new_area, follow this procedure:

```
$  mkdir  /install/new_area
$  mkdir  /install/new_area/ingres
$  mkdir  /install/new_area/ingres/data
$  mkdir  /install/new_area/ingres/data/default

$  chown  ingres  /install/new_area
$  chown  ingres  /install/new_area/ingres
$  chown  ingres  /install/new_area/ingres/data
$  chown  ingres  /install/new_area/ingres/data/default

$  chmod  755  /install/new_area
$  chmod  755  /install/new_area/ingres
$  chmod  700  /install/new_area/ingres/data
$  chmod  777  /install/new_area/ingres/data/default
```

To create directories and subdirectories for journals or checkpoints, follow the same procedure substituting "jnl" or "ckp" for "data" in the procedure.

When the form accurately describes the new *locationname*, select the **Save** menu item. If you decide not to create the locationname, select the **End** menu item.

Modifying a Locationname

When you exit the *locationname* form, only the usage permissions can be modified. The mapping between the *locationname* and area is permanent.

User Function

The **User** function can add, modify, or delete access to ODT-DATA by a user and can identify the databases that a user may access. The **User** function can be used to display an existing user's authorization information. Each invocation of the **User** function operates on a single user.

When the **User** function is selected from the main menu, you are prompted for a user name. Enter either the existing ODT-DATA user or the login of a user to be added. A form describing the user is displayed.

Adding a New User

When you enter a new user, you are prompted to verify your intent to create a new user. Enter yes and the form with the new user name, with default permissions is displayed.

Fill in the user information on the form that appears. The privileges are as follows:

- **Create database** permission allows a user to create new databases. The default is "yes."
- **Update system catalog** permission allows a user to directly update system catalogs (attribute, relation, etc.) using SQL. This is rarely needed. The default is "no."
- **Set trace flag** permission allows a user to set the debugging trace flags within ODT-DATA. The default is "no."
- **Superuser** permission allows a user to impersonate any other user in the system or to run the **accessdb** program.

Change the default privileges provided by entering a "y" or "n" next to the privilege shown. The first database owned by the user, is a read-only table field. The other table field, of the databases that the user can access, is filled in with the names of private databases you wish the user to access. Enter a database name, press **Return**, enter another database, and so on. When finished, press the **Menu** key to bring up the menu, and select the **Save** menu item to create the new user. If you decide not to create the user after filling in part of the form, use the **End** menu item.

Modifying an Existing User

To examine or modify an existing user's permissions, select the **User** command from the main menu and enter the user's login when prompted. Edit the form, including changes to the databases the user is explicitly authorized to access in the "May Access" table field. The "Owns" table field is read-only. Only the **createdb** and **destroydb** commands can be used to add and delete databases. Select the **Save** menu item to save the changes. To examine the user's permissions, or decide not to save your changes, specify the **End** menu item instead.

Deleting a User

To delete the authorization entry for an ODT-DATA user, select the **User** command from the main menu and enter the user's login. When the user's information form appears on the screen, issue the **Delete** menu item. If you decide not to delete the user, specify the **End** menu item instead.

Deleting a user is not permitted if that user owns any databases.

Catalog Function

In addition to establishing database extensions, locationname-area mappings and ODT-DATA users, the **accessdb** utility enables you to view current database access entries. The **Catalog** operation in the main **accessdb** menu calls a submenu of functions that display the data in separate frames.

The **Catalog** submenu contains the following commands:

Command	Function
Databases	Display a read-only table of databases.
LocationNames	Display the read-only table of *locationname*-area mappings.
Users	Display a read-only table of users.
Help	Get help about the operations in this menu.
End	Return to the **accessdb** main menu.

Databases Catalog Function

The **Databases** command displays a read-only table field containing a summary of each ODT-DATA database. For each database, the database name, owner, and global access flag are displayed. To change information about a database, return to the **accessdb** main menu and select the **Database** command. To scroll up and down in the table field, use the techniques described in the book *Using ODT-DATA* in the User's Guide.

To return to the **Catalog** menu, select the **End** menu item. To return to the main menu, select **End** again.

LocationNames Catalog Function

The **LocationNames** function displays a read-only table field containing each *locationname*-area mapping. Scroll through the table field to view its entire contents.

To return to the main menu, select the **End** menu item.

Users Catalog Function

The **Users** function displays a read-only table field containing a summary of each ODT-DATA user. For each user, the login and user access privileges are displayed. The display is a table that may be scrolled up and down to view its full contents. You cannot modify this information. To *change* information about a user, return to the **accessdb** main menu and select the **User** function.

To return to the main menu, select the **End** menu item.

Summary of Accessdb

The **accessdb** command can *only* be used by the ODT-DATA System Administrator or ODT-DATA superusers to modify, add, delete or list ODT-DATA users and database access privileges. It uses a forms-based interface, so **accessdb** must be run on a supported video terminal. Other ODT-DATA users cannot use **accessdb**, but they may use the **catalogdb** command, described in Chapter 4 of the optional *ODT-DATA SQL Reference Manual*. The **catalogdb** command displays *in read-only mode* information similar to **accessdb**.

Appendix C

ODT-DATA Environment Variables

Environment variables require definitions for your ODT-DATA installation. Certain installation-wide parameters should only be used by the "ingres" user in symbol tables. Other variables can be defined or redefined by individual users to customize their local ODT-DATA environment.

Setting Installation Wide Environment Variables

Environment variables are made system-wide by the "ingres" user, who can use the **ingsetenv** command to write them to the ODT-DATA symbol table.

For example, to set the environment variable ING_EDIT for an installation, you would log in as the "ingres" user and type:

```
$ ingsetenv ING_EDIT /usr/bin/vi
```

You can display all installation-wide variables by typing:

```
$ ingprenv
```

Setting User Defined Environment Variables

ODT-DATA environment variables can be set or reset by users in their local environment using UNIX commands. For example, one variable usually set in the user's environment is TERM_INGRES. It defines the ODT-DATA termcap definition to be used by the Forms System. To reset it in your local environment:

C shell:

```
% setenv TERM_INGRES vt100f
```

Bourne shell:

```
$ TERM_INGRES=vt100f
$ export TERM_INGRES
```

Users can display the values set in their environment:

```
$ env
```

Environment variables set in a user's local environment supersede the variables set in the symbol table. Set user-defined variables in the user's **.login** or **.profile** file.

Environment Variable List

Environment variables include the following:

- *DBNAME*_ING can be set installation wide or locally, to the full pathname of a file containing SQL commands. The commands are processed when a user connects to *DBNAME*, using the ODT-DATA SQL terminal monitor. Setting this variable is the equivalent of a user executing "\i file" in the terminal monitor each time they connect to *DBNAME*. Note that *DBNAME* is the name of the database and must be specified in uppercase.

- *DBNAME*_SQL_INIT may be set installation-wide or locally, to the full pathname of a file containing SQL commands. The commands are processed when a user connects to *DBNAME*, using the ODT-DATA SQL terminal monitor. Setting this variable is the equivalent of a user executing "\i file" in the terminal monitor each time they connect to *DBNAME*. Note that *DBNAME* is the name of the database and must be specified in uppercase.

- II_AUTHORIZATION is set during installation. The variable is set by the ODT-DATA System Administrator, during the **iibuild** procedure. It can only be updated by the "ingres" user using the **ingsetauth** command.

- II_CHECKPOINT is set to the full pathname for the default checkpoint location, ii_checkpoint. This variable is set by the ODT-DATA system administrator during the **iibuild** procedure, and it may not be changed, even during installation updates. Specific databases may designate alternate locations for checkpoints as a parameter to the **createdb** command.

- II_CONFIG is set to the full pathname of the ODT-DATA files directory during the ODT-DATA installation procedure.

- II_DATABASE is set to the full pathname for the default database location, ii_database. This variable is set by the ODT-DATA system administrator during the **iibuild** procedure and it may not be changed, even during installation updates. Specific databases may designate alternate locations as a parameter to the **createdb** command.

- II_DATE_FORMAT defines the format style for date *output*. Default is the US value with an output format of *dd-mmm-yyyy*. Default legal *input* formats for dates are

  dd-mmm-yyyy
  mm/dd/yy
  mmddyy

If the environment variable is set, it replaces one of these formats with an alternative format. The following are valid settings for II_DATE_FORMAT:

Value	Alternative Input Format	Replaces Input
US	*dd-mmm-yyyy*	
ISO	*yymmdd*	*mmddyy*
SWEDEN or **FINLAND**	*yyyy-mm-dd*	*dd-mmm-yyyy*
MULTINATIONAL	*dd/mm/yy*	*mm/dd/yy*
GERMAN	*dmmyy*	
	ddmmyy	
	dmmyyyy	
	ddmmyyyy	

In all cases, the alternative input format becomes the default output format. The three **US** default input formats listed above are unaffected by alternative settings and remain valid date input formats.

- II_DBMS_SERVER points to the DBMS server to which the user's process will connect to. The default value for any server connection request points to the primary DBMS server, which is created at startup.
- II_DECIMAL specifies the one character used to separate fractional and nonfractional parts of a number. Default value is the period (.), as in 12.34. Alternatively, the comma (,) can be used, as in 12,34. Only (.) and (,) are allowed.
- II_DML_DEF defines the default query language for an installation as **SQL**. RBF uses this variable.
- II_ERSEND is set during the ODT-DATA installation procedure to the full pathname of the *errlog.log* file.

- II_FILES is set to the full pathname of the ODT-DATA files directory during the ODT-DATA installation procedure.

- II_GCN*xx*_PORT contains the connect address of an installation's name server, where *xx* is its two-letter installation code (II_INSTALLATION).

- II_HELP_EDIT, if set to any value, adds an extra operation, **Edit**, to menus encountered in help text screens in the ODT-DATA forms systems. The **Edit** operation enables users to edit the help text in the screen with the text editor defined by ING_EDIT.

- II_INSTALLATION is a two-character code used to define a particular ODT-DATA installation. It should be defined at the installation level, using the **ingsetenv** command. It should never be defined at the user level.

- II_JOURNAL is set to the full pathname for the default journal location ii_journal. This variable is set by the ODT-DATA system administrator during the **iibuild** procedure and may not be changed, even during installation updates. Specific databases may designate alternate locations for journals as a parameter to the **createdb** command.

- II_LG_MEMSIZE is set to a value that will be used for the size, in bytes, of the locking and logging shared memory segment, created when **csinstall** is called by **iistartup**. The size of the segment is fixed, until the next time **csinstall** is executed. II_LG_MEMSIZE is calculated from the locking and logging parameters you selected for your ODT-DATA installation, when **iistartup** calls **iirun** to start the recovery process. The value of II_LG_ MEMSIZE should never be set below the default of approximately 200K.

  In the current release of the system, the size of the LG/LK shared memory segment defaults to approximately 200K. This segment's size is fixed once it is created by **csinstall**. The size may increase the next time **csinstall** is called. When the **dmfrcp** process starts up, it reads the locking and logging parameters that the user has specified during the **iibuild**, and calculates the maximum amount of memory that may be needed to support those parameters. The RCP sets a system-level environment variable II_LG_MEMSIZE to this value (represented in number of bytes). Until shutdown the rcp will continue to use the default size segment, possibly returning "out of (locking/logging) memory" error conditions on some queries. Subsequently, whenever the system is restarted (using **iistartup**, and thus calling **csinstall** to recreate the locking/logging segment), the LG/LK segment will be created with whatever size the variable II_LG_MEMSIZE is set to. II_LG_MEMSIZE should never be set below the default of approximately 200K.

- II_LOG_FILE is set to the parent directory of *ingres/log/ingres_log*. It determines the location of the installation-wide logging file. The **iibuild** procedure prompts you for this.

- II_MSG_TEST is set to false during the ODT-DATA installation procedure, and should not be changed.

- II_MONEY_FORMAT defines the format of money output. You can change output by setting the variable to a string with two symbols separated by a colon (:). The symbol to the left of the colon must be an "L" for a leading currency symbol, or a "T" for a trailing symbol. To the right of the colon, put the currency symbol you want prepended or appended to the amount. Examples follow:

Environment Variable Definition	Result
L:$	$100
T:DM	100DM
T: FF	100 FF

- II_MONEY_PREC shows the number of digits of precision to be used in the default representation of money data. The default is two digits (for decimal currency). Valid choices are 0, 1 and 2.

- II_NUM_SLAVES specifies the number of slave processes that a DBMS server will create to do disk operations. The default is two, but systems with many or faster disk drives have to increase the number of slave processes. For example:

```
$ ingsetenv II_NUM_SLAVES 4
```

  The above example allows each new server that is started to have four slaves. Stop and restart servers to reset the number of slaves. The maximum supported value is 10, and the minimum is 0. Setting this variable to 0 will degrade performance substantially.

- II_PATTERN_MATCH determines the set of pattern-matching characters for the QBF qualification function.

- II_PRINTSCREEN_FILE is provided for use with the **printscreen** function. It specifies a default file name for the output file of the **printscreen** function. If this variable is set and no file name is sent to the **printscreen** function, then no prompt is given and this file name is used. If not set, the user is prompted for a file name. If the file name *printer* is specified for II_PRINTSCREEN_FILE, the screen depiction is sent directly to the line printer. For more information, refer to the optional manual *Using ODT-DATA Through Forms and Menus*.

- II_SORT specifies the area that the location **ii_database** is mapped to. This is the default location, in which ODT-DATA creates temporary sort files. Sort files are created during the processing of ODT-DATA/SQL commands like **modify** and **create index**.

- II_SQL_INIT may be set locally, to the full pathname of a file containing SQL commands. When a user with this variable set connects to the ODT-DATA SQL terminal monitor, the commands in the named file are processed. Setting this variable is the equivalent of executing *\i file* in the terminal monitor each time a user connects to a database.

- II_SYSTEM specifies the parent directory of *ingres*. This environment variable should not be changed unless ODT-DATA is reinstalled.

- II_TEMPLATE is set during the ODT-DATA installation procedure to the full pathname of the database *template* directory.

- II_TERMCAP_FILE specifies an alternative termcap file to use. This file must be in termcap file format.

- II_THOUSANDS is set to a one-character symbol indicating the separator between thousands in numbers. Choices are the comma (,) and the period (.) The default is a comma.

- II_TMPDIR is used by ODT-DATA to locate temporary sort files. These are created during the processing of many ODT-DATA commands: queries (such as retrieval with joins and sort by clauses), the modify command and index, to name a few. II_TMPDIR is not for temporary files created by the ODT-DATA frontend and programs. Use II_TEMPORARY to locate these files elsewhere.

- ING_EDIT specifies the default editor spawned by various editor commands. The default for the entire installation is set during the ODT-DATA installation procedure. Users can also set this in their local environment.

- ING_PRINT specifies the default printer command issued by the **Print** function in ODT-DATA. The default is PRINT.

- ING_SET may be set installation-wide or locally, to a quoted string. The string must be 64 characters or less, or it will be invalid. It may contain either **set** commands separated by colons, or the word "include" followed by the full pathname for a file of **set** commands. If the variable is defined, the **set** commands are executed when any ODT-DATA frontend, including the terminal monitor, connects to a DBMS server. For example:

  C shell

  ```
  % setenv ING_SET "set autocommit on;
        set result_structure 'cbtree'"
  ```

  Bourne shell

  ```
  $ ING_SET="set autocommit on;
        set result_structure 'cbtree'"
  $ export ING_SET
  ```

- ING_SET_*DBNAME* may be set installation wide or locally, to a quoted string. The string must be 64 characters or less, or it will be invalid. It may contain either **set** commands separated by colons, or the word "include" followed by the full pathname for a file of **set** commands. If the variable is defined, the **set** commands are executed when any ODT-DATA frontend including the terminal monitor, connects to a DBMS server for *DBNAME*. Note that *DBNAME* is the name of the database and must be specified in uppercase.

- ING_SHELL, if defined, contains the pathname of the shell that ODT-DATA/MENU uses when the **shell** operation is invoked.

- ING_SYSTEM_SET may be set installation-wide, to a quoted string. The string must be 64 characters or less, or it will be invalid. It may contain either **set** commands separated by colons, or the word "include" followed by the full pathname for a file of **set** commands. If the variable is defined, the **set** commands are executed when any ODT-DATA frontend, including the terminal monitor connects to a DBMS server.

- INIT_INGRES may be set locally, to the full pathname of a file containing SQL commands. When a user with this variable set connects to the ODT-DATA SQL terminal monitor, the commands in the named file are processed. Setting this variable is the equivalent of executing "\i file" in the terminal monitor each time a user connects to a database.

- TERM is the terminal description for the terminal upon which you are executing one of the ODT-DATA forms-based products, such as QBF or VIFRED. See *Using ODT-DATA Through Forms and Menus* for a complete list of supported values. This environment variable is defined in the user's **.login** or **.profile** file.

- TERM_INGRES contains the terminal designation for the terminal upon which you are executing one of the ODT-DATA forms-based products, such as QBF or VIFRED. See the optional manual *Using ODT-DATA Through Forms and Menus* for a full list of supported values. This environment variable is most conveniently set in a **.login** or **.profile** file. TERM_INGRES takes precedence over TERM in defining terminal type to ODT-DATA and allows TERM to be defined differently for use by other UNIX programs such as *vi*.

Appendix D
ODT-DATA System Recovery

The recovery tools, **rolldb** and **finddbs**, are provided by ODT-DATA to recover from a database or installation corruption. This section briefly discusses when to use each of these tools, and describes the **finddbs** command. See the optional *ODT-DATA SQL Reference Manual* for additional information about these commands.

The **rolldb** command can regenerate a database from a static backup called a checkpoint. If your installation keeps a dynamic record of changes to the database called *journals*, they can be used to restore the database up to the time of the system failure.

If system failure causes corruption of the information in the *iidbdb*, the **finddbs** command can be used to recover the locations of databases. This information is stored in the *iidbdb* system catalog, *iidatabase*. The **finddbs** command allows users to search any directories they choose for databases and enter them into the *iidatabase* table.

Using finddbs

The syntax of the **finddbs** command is as follows:

finddbs [**-a** | **-r**] [**-p**]

The arguments are the following:

-a	This runs **finddbs** in "analyze" mode, but simply writes intended updates to the terminal. Use this option if you suspect *iidbdb* database catalog is out of order.
-r	This runs **finddbs** in "replace" mode, actually inserting databases located by the program in the *iidbdb*.
-p	makes all located databases private, except for the *iidbdb*.

The **finddbs** command helps recover ODT-DATA when *iidbdb* has been destroyed. Only the ODT-DATA system administrator can use it. The **finddbs** command runs in analyze mode, which is the default, or replace mode. Analyze mode is selected with the **-a** flag, while replace mode is selected with the **-r** flag. Analyze mode is a read only mode that informs you about possible errors in the *iidbdb iidatabase* table with a new table formed by scanning the ODT-DATA directories on a set of devices for databases. The use of replace mode is not recommended unless the *iidbdb* **iidatabase** table is in error.

When **finddbs** starts, it first builds a list of found databases. The list is formed by scanning the $II_SYSTEM/*ingres/data/default* directory. You are then prompted for any other directories to search. If you respond with a directory name not preceded by a slash, **finddbs** looks in $II_SYSTEM/*ingres/data/name*, where *name* is the name you specify. If you specify a directory name beginning with a slash, **finddbs** looks in */name/ingres/data/default*. Enter an empty line to exit this scan phase. It is important to search all directories that contain databases. If your installation creates all its databases in the default directory, then the default search will suffice. In non-default directory names are specified on the **creatdb** command line at your installation, or you have extended databases to alternate locations with **accessdb**, then you must specify all such directory names so that **finddbs** can find the databases they contain.

NOTE: Use **finddbs** in a peer ODT-DATA installation that is not located in the home directory of the ODT-DATA system administrator. Define the ODT-DATA system administrator's environment to run the peer installation and use **finddbs** as described.

Use the analyze mode of **finddbs** if you suspect that the *iidatabase* table of the *iidbdb* is in error. The output of analyze mode consists of two lists. The first gives names of databases present on disk but not contained in the *iidatabase* table. If analyze mode reports differences between the found database list (built during the scan phase) and the *iidbdb iidatabase* table, you may decide to run **finddbs** in replace mode.

The **finddbs** command replace mode replaces the contents of the *iidbdb iidatabase* table with a new table consisting of the databases found during the directory scan phase. Before running **finddbs** in replace mode you should first run it in analyze mode to see what changes would be performed by replace mode. By default, replace mode sets the access status of all databases to "global". The **-p** flag causes all databases to be made private, except for the *iidbdb*.

Examples of the **finddbs** command are:

```
#     Run finddbs in analyze mode to examine
#     directories and iidatabase table in the iidbdb.

$     finddbs -a

#     Replace the contents of the iidatabase system
#     catalog with databases found running this
#     command.

$     finddbs -r
```

In both cases, **finddbs** prompts you for the directories to check for ODT-DATA databases.

WARNING

When the **finddbs** command is run in the replace mode, the entire contents of the "iidatabase" table are destroyed and replaced. All directories containing databases must be scanned. Only the directories that are scanned have their databases included in their new table.

Appendix E
Running ODT-DATA under the Network File System

The following appendix assumes that you are familiar with the basics of mounting filesystems in an environment that provides Network File System (NFS) services.

As the ODT-DATA system administrator, you should understand the following issues and read the scenarios below, before installing ODT-DATA products in an NFS environment.

ODT-DATA provides its own concurrency control services to permit multi-user access to database tables. It is important to know that concurrency control is bound to the shared memory associated with a single processor. The ODT-DATA DBMS lock manager will provide concurrency control for multiple users running on a single processor, but there is no provision within the DBMS server process (iidbms) for synchronized locking of file resources between multiple network nodes.

If ODT-DATA data resides on NFS disk partitions that are exported to other network nodes capable of running the ODT-DATA DBMS server process, transaction concurrency can be compromised. In this environment, the potential exists for two or more ODT-DATA DBMS server processes, using different shared memory, to access the same database independently. Then, the corruption of the user tables and system catalogs is possible. To prevent this, use one of the following examples as a guideline for installing ODT-DATA on multiple nodes that share Network File Systems.

Configuration Scenarios

The following two scenarios have been described to assist the ODT-DATA system administrator at configuration startup time. Each is typical of an ODT-DATA installation in an environment that provides NFS services.

NOTE: The ODT-DATA environment variable $II_SYSTEM is used throughout these scenarios to describe a UNIX file system path specification. When specifying the disk partitions to be exported, mounted, and so on, you will need to replace $II_SYSTEM with the physical path.

Scenario 1

This is an example of a shared ODT-DATA installation, where the ODT-DATA processing is shared by the NFS server and clients using ODT-DATA/NET. The ODT-DATA frontend tools and applications may be executed on both the server node and the client nodes. The DBMS server and other installation related processes are executed only on the server node.

The ODT-DATA system software distribution is installed on the NFS server. The binaries, libraries, and auxiliary files are shared. The $II_DATABASE, $II_CHECKPOINT, and $II_JOURNAL locations are shared. All ODT-DATA DBMS server processes (**iidbms**), the archiver (**dmfacp**) and recovery (**dmfrcp**) processes are executed on the NFS server. Concurrency management is maintained only by the NFS server node. The clients' ODT-DATA frontend application processes are connected to a DBMS server process, on the NFS server, by using ODT-DATA/NET.

NOTE: It is important that the resolution of $II_SYSTEM is the same for the NFS server as for the clients. You can do this with symbolic links.

Server characteristics:

- $II_SYSTEM/ingres is exported to the frontend clients.
- The **iibuild** procedure is executed at ODT-DATA installation time. See details in Chapter 4 of this book.
- The **iidbms, dmfacp,** and **dmfrcp** processes are running.
- The **iistartup** command is in */etc/rc.local*, and it is executed at system startup time.
- ODT-DATA frontend tools and applications may be executed locally on the NFS server node. For example:

```
$  qbf  dbname
```

Client characteristics:

- Only ODT-DATA frontend tools and applications execute on this node. No DBMS server, archiver, or recovery process, or concurrency control is running on this node.
- **$II_SYSTEM/ingres** is imported from the NFS server, to a "mount" location.

- In the following example of a filesystem, configuration for clients, "/install/61" is $II_SYSTEM for both NFS server and clients.

```
$ df
Filesystem kbytes used avail capacity Mounted on
server:/export/root/client
server:/export/exec/odt/usr
server:/install/61/ingres
```

- ODT-DATA frontend tools and applications are connected to the remote DBMS server process (iidbms) on the NFS server node by using ODT-DATA/NET. For example:

```
$ qbf nodename::dbname
```

NOTE: Not all ODT-DATA commands and utilities are available with ODT-DATA frontends executing over ODT-DATA/NET.

Scenario 2

This is an example with separate ODT-DATA installations for the NFS server and clients. The NFS server and clients maintain logically separate installations with distinct $II_DATABASE, $II_CHECKPOINT, and $II_JOURNAL locations. The ODT-DATA system software distribution is installed on the NFS server and can be reinstated entirely on the clients, where symbolic links to the server installation are available. This minimizes disk space usage at the expense of communications traffic overhead.

Clients can share static ODT-DATA files with the NFS server, like the ODT-DATA binaries, libraries, utilities, and technical notes. ODT-DATA DBMS server processes (iidbms), archiver (dmfacp), and recovery (dmfrcp) processes and concurrency control, run on BOTH the NFS server and clients. ODT-DATA frontend tools and applications can be executed either locally or in connection with a remote DBMS server process by using ODT-DATA/NET.

NOTE: To improve performance in the development environment, all clients should have their own copies of the *INGRES/lib* directory.

Server characteristics:

- **$II_SYSTEM/ingres** is exported to NFS clients. For example, to see the list currently exported, type:

```
$ cat /etc/exports
```

- The **iibuild** procedure is executed at ODT-DATA installation time. See details in Chapter 4 of this book.
- The **iidbms, dmfacp,** and **dmfrcp** processes are running.
- The **iistartup** command in */etc/rc.local* is executed at system startup time.
- ODT-DATA frontend tools and applications may be developed locally on the NFS server. For example:

```
$ qbf dbname
```

Client characteristics:

NOTE: The clients will share the *bin, lib, utility,* and *notes* directories with the NFS server, using an NFS mount and symbolic link. The *data, jnl, ckp,* and *files* directories will be set locally using the **iibuild** utility. Create the *files* directory by copying the central files directory.

- **$II_SYSTEM/ingres** is imported from the NFS server, to some "mount" location. This "mount" location will not be used as $II_SYSTEM; a symbolic link will need to be set up for this purpose.
- The **iibuild** command is executed at ODT-DATA installation time. See the filesystem configuration example below, and details in Chapter 4 of this book.
- Concurrency control and **iidbms, dmfacp,** and **dmfrcp** processes are running on the clients.
- The **iistartup** command in */etc/rc.local* is executed at system startup time.
- ODT-DATA frontend applications and tools may be connected to a remote DBMS server process (**iidbms**) by using ODT-DATA/NET:

```
$ qbf nodename::dbname
```

They may also be connected to a local DBMS server process (iidbms):

```
$ qbf dbname
```

An example of how you configure a filesystem for a client follows:

```
$ df
Filesystem kbytes used avail capacity Mounted on
server:/dev/sd0a/db
server:/dev/sd0b/bck
server:/export/root/client/
server:/export/exec/odt/usr
server:/usr/m/server/usr
server:/install/ingres/m/server/ingres
```

- Identify a location for $II_SYSTEM on the client. Define a symbolic link for the following directories from the mounted location to the $II_SYSTEM location. For this example $II_SYSTEM is set to "/install/61".

```
$ ln -s /m/server/ingres/bin/install/61/ingres/bin
$ ln -s /m/server/ingres/notes/install/61/ingres/notes
$ ln -s /m/server/ingres/lib/install/61/ingres/lib
$ ln -s /m/server/ingres/utility/install/61/ingres/utility
$ ln -s /m/server/ingres/vec/install/61/ingres/vec
$ ln -s /m/server/ingres/dbtmplt/install/61/ingres/dbtmplt
$ ln -s /m/server/ingres/release.doc/install/61/ingres/release.doc
```

- Copy the $II_SYSTEM/files directory from the server node to $II_SYSTEM location of the client node.

```
$ rcp -r server:/m/server/ingres/files/install/61/ingres
```

- Before **iibuild** is executed, a *.version* file must be copied from the server node to the **$II_SYSTEM/ingres** directory of the client. This prevents the entire distribution from being read onto the client nodes during the installation procedure.

```
$ cp /m/server/ingres/.version/install/61/ingres
```

- Execute installation procedure in Chapter 4, to create directories and set up the installation.

Warning: A version file must exist before **iibuild** is executed.

The following environment variables must be set to directories which are physically resident on the local disk, and NOT set to directories which are resident on an NFS mounted disk.

Example:

```
$II_DATABASE = /db
$II_CHECKPOINT = /bck
$II_JOURNAL = /bck
$II_LOG_FILE = /bck
```

Likewise, alternative ODT-DATA locations must reside on local disks.

NOTE: When upgrading a Scenario 2 installation, copy new file versions to the non-linked directories. Backup copies of the installation-dependent files should be made, as with the *INGRES/files* directory.

Glossary

abstract datatype A datatype that is not native to the operating system, but is implemented with a data structure and a set of operators. Examples of data operators are the interval function for the date and the addition for money.

aggregate A Computation that operates on set of values(for example an average). See also **set function**.

attribute In VIFRED, a characteristic such as highlighting, or, a validation check that affects the display and behavior of a field.

backend The process responsible for interacting with the data. Also called data manager. The backend receives query language commands from a frontend and returns the appropriate data. See also **frontend**.

base table A physical table as opposed to a view. Also used in contrast to a secondary index. See also **table**.

break column A column in a report for which a special action, such as a subtotal, occurs when data values change.

Btree A storage structure characterized by a dynamic index tree.

catalog A table that keeps track of database objects. Catalogs are automatically supplied and maintained by ODT-DATA.

cell The intersection of a row and a column in a table field (or more rarely, in a table).

checkpoint A static backup ODT-DATA creates of a database.

column A vertical selection of data in a table or afield that represents one piece of information.

correlation name	An alternate name in SQL for a table, usually a shortened form of the name. See also **range variable.**
data manager	The process responsible for interacting with the data. Also called backend. See also **backend.**
data window	The area of a form where information can be entered.
database	In the relational database model, a collection of tables.
database administrator	The ODT-DATA user that owns the database (for example the user that created it).
data set	The set of records retrieved by the query statement. In particular, the set of records associated with a table field by a query.
DBA	Database administrator.
default	A selection provided automatically by ODT-DATA, such as a form in QBF or a default report.
dynamic SQL	A part of embedded SQL allowing the user to build queries at run-time.
Embedded SQL	An ODT-DATA application development tool in which SQL commands are placed in a third generation language program such as FORTRAN or COBOL.
field	An area of a form used for data entry and retrieval, composed of a title, data window and attributes.Also used as a synonym for column. See also **column, table field, simple field.**
form	The computerized equivalent of a paper form, where users can enter, store and retrieve data.
frame	A piece of an application composed of a form and a menu.
frontend	A user interface to ODT-DATA. It can be an ODT-DATA tool (for example QBF) or a custom application. See also **backend.**

FRS (Forms Run-Time System) The part of ODT-DATA that controls the display of forms and user's manipulation of forms and menus.

frskey A logical key used in ODT-DATA application code to refer to a keyboard key.

hash A storage structure characterized by a number of "buckets" (primary pages) where records are placed according to the value of a random function applied to their keys.

heap A default storage structure having no index and no ordering.

inconsistent database A database on which a transaction was not completed. A transaction underway when the system crashed.

index See also **secondary index.**

integrity A backend test to assure that data matches certain specifications.

ISAM (Indexed Sequential Access Method) A storage structure characterized by a static index tree.

JoinDef One or more tables joined together in QBF and functioning as a single object in QBF for data manipulation purposes.

journal A log of transactions since the last static backup. For each transaction, ODT-DATA journals show the change, the date and time of the change, and the user who made it.

key The part of a record that uniquely identifies it (logical key). The column(s) of a table on which a storage structure is built. A heap storage structure has no key.

locking The mechanism by which ODT-DATA makes a multiuser environment possible (for example it protects each user's work from corruption by other users).

Menu key The key that moves the cursor to the menu line.

MST	(Multi-statement Transaction) A transaction including several statements identified and executed as a block.
null value	A special value representing missing or unknown information.
object	Any database entity: table, form, QBF name, application, joindef, graph, report.
ODT-DATA/MENU	The tool allowing users to access all of ODT-DATA's menus.
optimizer	The part of ODT-DATA responsible for finding the fastest way to execute queries.
page	An ODT-DATA page is a 2,048-byte structure with 2,010 bytes available for storing user data.
PERMIT	A backend statement to allow users other than the DBA to access data.
QBF	(Query By Form) An ODT-DATA tool performing Query Execution; appending, retrieving or modifying data in tables, and, **Join Definition**; specifying a set of tables for a query.
QBF name	Mapping of a form to a table or a joindef.
query	A data statement for viewing, changing, adding or deleting data.
query target	An object used in QBF. Query targets include tables, joindefs and QBF names.
range variable	An alternate name for a table, usually a shortened form of the name.
RBF	(Report By Forms) The ODT-DATA menu based editor for customizing reports.
record	A set of related data in a table; a tuple.Also used to refer to a tuple in the dataset associated with a table field (tuples in the data field are called rows). See also **row**.

relation	The technical word for table. See also **table.**
report	Displaying information from the database in an easy to use format.
REPORT	The ODT-DATA tool for running default or user defined reports.
row	A set of related data in a table; a tuple.Also used to refer to a tuple in a table field (tuples in the dataset associated with the table field are called records). See also **record.**
secondary index	A table composed of a key and a pointer to the records of the base table. ODT-DATA automatically maintains the index as records are added or updated in the base table.
server	A process which provides particular services to a number of processes. The data manager is a server in ODT-DATA release 6.0.
set functions	In SQL, a computation that operates on a set of values (for example an average). See also **aggregate.**
simple field	A field containing a single piece of data. See also **table field.**
SQL	(Structured Query Language) A language used to define, manipulate and protect data.
storage structure	A method of arranging the pages of a table. ODT-DATA supports four storage structures: heap, hash, ISAM and Btree.
subquery	A SQL subselect nestled within another SQL statement.
subselect	A SQL select statement containing only one select keyword without a union or an order by. a subselect issued to build a search condition for the main query.
system catalog	See **catalog.**
table	A set of data arranged in rows and columns.
table field	A field containing several pieces of data (one or more columns).

TABLES	A forms based tool accessible from the ODT-DATA/MENU for creating, examining, and deleting tables.
Terminal Monitor	An interface where the user can enter query language commands. The two terminal monitors are the line terminal monitor and the full-screen terminal monitor.
title	A character string used to identify a field on a form.
transaction	A set of statements that function as a unit.Either all or none are executed. A transaction may consist of a single statement or several statements grouped in an MST (Multi-statement transaction).
trim	A string of characters on a form used to decorate or instruct.
tuple	A row or record in a table.
validation check	In VIFRED, a test to insure that data entered into a field meets certain specifications.
view	A logical definition of data taken from one or more tables.
VIFRED	(Visual Forms Editor) The ODT-DATA menu based editor for customizing forms.
VIGRAPH	The ODT-DATA tool for creating, modifying, and running graphs.

Index

This index locates entries by book name and page number. Each of the book names in this volume is indicated with an abbreviation, listed in the table below. A key with this information is at the bottom of each following index page.

Book Name	Abbreviation
Administering ODT-VIEW	*VIEW*
Administering ODT-OS	*OS*
Administering ODT-NET	*NET*
Administering ODT-DOS	*DOS*
Administering ODT-DATA	*DATA*

A

Accelerator keys, *VIEW* 7
accept command (lineprinter), *OS* 225
Access privileges, *NET* 12, 22
accessdb command
  functions, *DATA* 52, 59
  using, *DATA* 51
Account is disabled, error message, *OS* 102
Accountability, *OS* 47
Accounts
  activity reporting, *OS* 176
  adding user, *DOS* 4
  deleting user, *DOS* 4
  disabled, *OS* 102
  enabling, *OS* 102
  locking, *OS* 159
  unlocking, *OS* 159
Active connections display, *NET* 25
Adding
  a computer
    mkself, *NET* 80
Adding *(continued)*
  a computer *(continued)*
    to a network, *NET* 4, 80
    with custom utility, *NET* 80
  a user, *OS* 151
Address
  network, *NET* 8
  parsing, *OS* 301
  resource record, *NET* 43
Administrative commands, summary, *OS* 216
Administrative roles, *OS* 50
Alerting to mount a print wheel, *OS* 255
Alias, defined, *NET* 155
ALIAS entry, *OS* 300
Alias files
  alias.list, *OS* 304-305, 312
  alias.user, *OS* 305, 312
  examples of, *OS* 305
  MMDF
    converting, *OS* 312
    editing, *OS* 304
Alias tables, search order, *OS* 300

Analyze mode (finddbs command), *DATA* 71
Anonymous account, *NET* 12,23
ap parameter, *OS* 301
Archiver log, *DATA* 46
Archiving, defined, *DATA* 2
ARP, defined, *NET* 155
ARPA, defined, *NET* 155
Attributes, default printing, *OS* 260
Audit
  audit daemon, *OS* 59
  authorizations, *OS* 60
  collection, *OS* 70
  data reduction/analysis, *OS* 60
  database, *OS* 98
  described, *OS* 49
  device driver, *OS* 58
  directories, *OS* 71
  disk space, *OS* 87
  enabling, disabling, *OS* 75
  event types, *OS* 63,71
  files, *OS* 76-77
  mandatory auditing, *OS* 64
  procedures, *OS* 69
  records produced
    application audit, *OS* 84
    audit subsystem, *OS* 86
    login/logoff, *OS* 84
    protected database, *OS* 85
    protected subsystem, *OS* 86
    system call, *OS* 80
    terminal/user account, *OS* 87
    user password, *OS* 85
  reporting, *OS* 77
  selection files, *OS* 77
  subsystem, *OS* 56
  subsystem parameters, *OS* 72
  sysadmsh selection, *OS* 60
  system audit event mask, *OS* 65
  user-specific event mask, *OS* 65
audit authorization, *OS* 172
Audit: file system is getting full, error message, *OS* 104
auth authorization, *OS* 172
Authentication database ... inconsistency, error message, *OS* 105
AUTHLOG, *OS* 303
Authorization string, *DATA* 63
Authorizations
  assigning, *OS* 157
  changing default, *OS* 172
  default, *OS* 164
  described, *OS* 48
  types
    kernel, *OS* 53,175
    secondary, *OS* 173
    subsystem, *OS* 172
autoexec.bat
  specifying for use with DOS application, *DOS* 45-47
Automatic activation (LAN Manager Client), *NET* 81

B

Backspace key, *OS* 4
backup, authorization, *OS* 172
Backups
  See also Filesystem
  DOS partition, *DOS* 15
  ODT-DOS filesystem, *DOS* 15
badhosts channel, *OS* 301,303,308
Berkeley Internet Name Domain (BIND), *NET* 1,12,33
Bidirectional port, *NET* 136
/bin directory, *OS* 182
bind, defined, *NET* 155
BIND (Berkeley Internet Name Domain), *NET* 1,12,33
BITNET, *NET* 5,37
Block
  arrangement on disk, *OS* 190
  defined, *OS* 35
  device, *OS* 132,134
  number, *OS* 190
  ownership, *OS* 37
  size, *OS* 190
Boot files for name server, *NET* 48

Bootstrap program, *OS* 24
Break key, *OS* 4
Bridge, defined, *NET* 155
Broadcast address for internet, *NET* 19
Broadcast network, defined, *NET* 155
BSD, defined, *NET* 155
Buffer network
  auto configuration, *NET* 101
  maximum size, *NET* 102
  relation to streams buffer, *NET* 101
  size, *NET* 100-101
Building a remote network system, *NET* 103
Bus, defined, *NET* 155
Bus cards, *OS* 279
Button bindings
  alternate specifications, *VIEW* 5
  configuring, *VIEW* 49

C

C program
  compilation header files, *OS* 186
  library files, *OS* 186
Cable network, *NET* 10
Cache, defined, *NET* 155
Cache initialization, *NET* 40
Caching-only server
  defined, *NET* 156
  example of, *NET* 48
Cannot obtain database information, error message, *OS* 103
Can't rewrite terminal control entry, error message, *OS* 104
Cartridge tape
  configuring, *OS* 265
  drive, *OS* 265
  /etc/default files, *OS* 270
  formatting, *OS* 273
  maintaining, *OS* 272
Case, network name, *NET* 3
Catalog function
  databases, *DATA* 59
  described, *DATA* 56
Catalog function *(continued)*
  locationnames, *DATA* 59
  user, *DATA* 59
CCITT, defined, *NET* 156
Changing filesharing parameters (LAN Manager/X performance), *NET* 84
Changing passwords, *OS* 151
Changing screen colors
  RGB database, *VIEW* 134
  .Xdefaults, *VIEW* 133
Changing user IDs, *NET* 3
chan.log file, *OS* 303
Channel
  definition, *OS* 300
  directory, *OS* 302
  tables
    .chn file, *OS* 301
    search order, *OS* 301
Character
  device, *OS* 132, 134
  print wheels, *OS* 253
  sets, *OS* 253
Checkpoints, defined, *DATA* 11
checkque program, *OS* 310
chmodsugid authorization, *OS* 53, 175
.chn file, *OS* 307
chown authorization, *OS* 53, 175
chroot system call, *NET* 23
cleanque program, *OS* 310
Client, defined, *NET* 156
  setting up NFS, *NET* 12, 63
  starting automatically at login, *VIEW* 8
  X Window System, *VIEW* 4
clock, setting system time, *OS* 26
Clock synchronization service, *NET* 57
Cloning device, defined, *NET* 156
Cloning drivers, *NET* 16
CNCBS
  default value, *NET* 85
  defined, *NET* 86
Colon, *OS* 305-308
Colors
  customizing, *VIEW* 133-134
  defining in RGB database, *VIEW* 134

COM ports
administering, *DOS* 11
Command Line Options, *VIEW* 118
Commands
administrative, *OS* 216
location
/bin directory, *OS* 182
/usr/bin directory, *OS* 186
user, *OS* 215
Communications protocols, *NET* 5
Compatibility network, *NET* 3
Computer name restrictions, *NET* 80
Computer name (unique), *NET* 3
configaudit authorization, *OS* 53, 60, 175
config.sys
specifying for use with DOS application, *DOS* 47, 49
use with ODT-DOS, *DOS* 32-33
Configurable driver routines, *OS* 131
Configuration
data structures, *NET* 84
installation, *NET* 79
mkself utility, *NET* 80
network, *NET* 2
scripts for changing video systems, *VIEW* 137
Configuration file
format, *NET* 21
MMDF
converting, *OS* 311
editing, *OS* 297
configure command, *OS* 133
Configuring
a device driver, *OS* 129
consumer, *NET* 79
ODT-DATA, *DATA* 7, 11
screen colors, *VIEW* 133-134
STREAMS, *NET* 16
the interface, *NET* 18
UNIX computers (custom), *NET* 79
confstr parameter, *OS* 302
Connection, defined, *NET* 156
Connectionless, defined, *NET* 156
Connectionless packet delivery, *NET* 6
Consistency
system, *NET* 3
user ID numbers, *NET* 3
Console
display adapters supported, *DOS* 7
requirements, *DOS* 6
constable file, *NET* 81, 98-100
Consumer
computers, *NET* 10
configuration file, *NET* 81
defined, *NET* 156
network, *NET* 82
read window, *NET* 97
requests, *NET* 10
write window, *NET* 97
Content types, *OS* 251
Context Indicator (sysadmsh screen), *OS* 8
Continuing an alias line, *OS* 305
Converting MMDF configuration files, *OS* 311
Copy protection, and system backup procedures, *DOS* 15
Copy-protected DOS applications, installing, *DOS* 50-52
CORMAPNCB
default value, *NET* 85
defined, *NET* 87
cpio program, filesystem restoring, *OS* 124
Crash utility, *VIEW* 125
cron authorization, *OS* 172
cscleanup command, *DATA* 43
csinstall command, *DATA* 43
CSNET, *NET* 4, 36
csreport utility, *DATA* 22
Ctrl keys, *OS* 4
custom utility, *NET* 79-80

D

Daemon
defined, *NET* 156
in NFS, *NET* 63
DARPA
defined, *NET* 156
internet, *NET* 4, 36

Data files, methods of loading, *VIEW* 59,64
Data link level, defined, *NET* 157
Database administrator (DBA),
responsibilities, *DATA* 1
Database command, *DATA* 52,59
Databases
access, *DATA* 49
extending, *DATA* 53
MMDF, *OS* 310
network, *NET* 22
private, *DATA* 50
Datagram
defined, *NET* 157
described, *NET* 6
date, setting system clock, *OS* 26
dbmbuild program, *OS* 310
DBMS (Database Management System)
and the server, *DATA* 19-21
parameters, *DATA* 32
startup, *DATA* 7,31
troubleshooting of startup, *DATA* 37
DBNAME_ING, *DATA* 62
DBNAME_SQL_INIT, *DATA* 47,63
DCL, *VIEW* 93
DDN, defined, *NET* 157
Debugging NFS, *NET* 65
Default printing attributes, *OS* 260
Default tar settings, *OS* 270
Default value
/etc/conf/cf.d/mtune file, *NET* 84
Lan Manager/X parameters, *NET* 84,86
/usr/lib/xnet/xnetrc file, *NET* 94
Defaults
accounts, *OS* 164
/etc/default directory, *OS* 184
files, *VIEW* 51
security
authorization parameters, *OS* 164
login parameters, *OS* 164
password parameters, *OS* 164
Defining
network computers (mkself utility), *NET* 80
user IDs (l), *NET* 3
deliver program, *OS* 302-303,310
Delivery
mode, *OS* 302
tailoring, *OS* 299
Description Line (sysadmsh screen), *OS* 8
Desktop Command Language, *VIEW* 93
Desktop Commands, *VIEW* 94
Desktop Manager
Command Line Options, *VIEW* 118
configuration
appearance, *VIEW* 51
application examples, *VIEW* 55
behavior, *VIEW* 53
drag triggers, *VIEW* 64
drop rules, *VIEW* 65
icon triggers, *VIEW* 62
icons, *VIEW* 53,59
mouse triggers, *VIEW* 62
Defaults File,
$HOME/.Xdefaults, *VIEW* 101
defined, *VIEW* 1
loading data files, *VIEW* 64
locked files, *VIEW* 89
Message Files, *VIEW* 111
rule files
components, *VIEW* 74
defined, *VIEW* 69
format, *VIEW* 70
rule files, *VIEW* 59
Support Utilities, *VIEW* 119
Destination, defined, *NET* 157
Destination address, defined, *NET* 157
/dev directory, *OS* 182
Device
assigning, *DOS* 28
assignment database, *OS* 98
attachment
dangers of incorrect specifications, *DOS* 28
direct, *DOS* 26
example specification of, *DOS* 27-29
restrictions on direct, *DOS* 26
specification fields, *DOS* 26
syntax for, *DOS* 26
configuring, *DOS* 26

Device *(continued)*
drivers
adding, *OS* 129
preconfigured, *OS* 131
name mapping syntax, *DOS* 30
number
major, *OS* 134
minor, *OS* 134
special filenames, *OS* 189
special files, *OS* 131, 134
specifications
for direct attachment, *DOS* 26
specifications, *DOS* 30
df command (display free space), *OS* 35
Dialer programs
compiling, *OS* 196
using, *NET* 134; *OS* 196
Dial-in, special password protection, *OS* 106
Dial-in/Dial-out, *NET* 136
Directory
block usage, *OS* 36
GID bit, setting, *OS* 96
organization, *VIEW* 65
directory, sticky bit, *OS* 92
disable command, lineprinter, *OS* 219
Disabled accounts, *OS* 102
Disabled terminals, *OS* 103
Discretionary Access Control (DAC)
defined, *OS* 48
denial, *OS* 57
Disk
block number, *OS* 190
block size, *OS* 190
configuration, *OS* 315
damage, *OS* 38
drives
administering, *DOS* 15
mounting as UNIX device, *DOS* 15
sharing among DOS users, *DOS* 16
free space, *OS* 34
gap number, *OS* 190
partition (DOS), *DOS* 16
security, *OS* 45
usage, *OS* 36
Diskettes, virtual
assigning, *DOS* 24
creating, *DOS* 19, 23-24
defined, *DOS* 23
size, *DOS* 24
using, *DOS* 24
Display
active connections, *NET* 25
interfaces, *NET* 27
protocol statistics, *NET* 30
routing table, *NET* 28
Display adapters, types supported, *DOS* 7
Display Area (sysadmsh screen), *OS* 9
DL_ATTACH primitive, *NET* 17
dmfacp, *DATA* 46
dmfrcp, *DATA* 46
DNS, defined, *NET* 157
.dom file, *OS* 306
Domain
database files for name server, *NET* 48
defined, *NET* 157; *OS* 302
.dom files
LHS (left-hand side), *OS* 307
RHS (right-hand side), *OS* 307
management, *NET* 54
matching, *OS* 303, 307
name
defined, *OS* 297
fully qualified, *OS* 306
registered, *OS* 298
top-level, *OS* 307
name pointer resource record, *NET* 45
setting up your own, *NET* 4, 36
tables, search order, *OS* 303
DOS
administration menu
fields explained, *DOS* 45-47
application menu, *DOS* 43
application programs
adding to dosadmin database, *DOS* 50
configuring to run from UNIX shell, *DOS* 49-50
deleting UNIX links to, *DOS* 55
installing copy protected, *DOS* 50-52

DOS *(continued)*
application programs *(continued)*
installing on DOS partition, *DOS* 52
installing personal, with dosadmin command, *DOS* 48-49
installing public, with dosadmin, *DOS* 39-47
installing under DOS without ODT-DOS, *DOS* 53-54
installing with dosadmin command, *DOS* 37-46, 50
removing from fixed disk, *DOS* 55
removing from Open Desktop, *DOS* 54
files, use of, *OS* 137
filesystems
configuring, *OS* 147
mounting, *OS* 137
images
and on-card ROMS, *DOS* 32
and standard ROMS, *DOS* 32
creating custom, *DOS* 34
defined, *DOS* 31
location of default, *DOS* 31
using dosadmin to create, *DOS* 33-34
when to remake, *DOS* 31-32
partition
backing up files, *DOS* 15
changing permissions, *DOS* 18
installing DOS application programs on, *DOS* 52
limited protection of, *DOS* 17
physical, *DOS* 16-17
removing, *OS* 144
seen as UNIX file, *DOS* 17
virtual, *DOS* 19-22
printer
adding new, *DOS* 13
configuring default, *DOS* 12
specifying, *DOS* 13
STACKS command
interpretation of, *DOS* 33
SUBST command, *DOS* 54
UNIX installed on, *OS* 142
utilities, use of, *OS* 137
dosadmin command
using to create DOS images, *DOS* 33-34
using to create virtual diskettes, *DOS* 23-24
using to create virtual DOS partition, *DOS* 20-21
using to install DOS application programs, *DOS* 37-45, 48-50
dosadmin database
adding DOS applications to, *DOS* 50
removing DOS applications from, *DOS* 55-57
dosadmin menu, fields explained, *DOS* 45-47
dosdev file
setting up, *DOS* 28
syntax, *DOS* 29
Drive E:
See DOS partition, physical, *DOS* 52
Driver
cloning of, *NET* 16
device nodes, *NET* 13
in kernel, *NET* 13
module, *OS* 131
non-cloning, *NET* 17
prefix, *OS* 131-132
priority level, *OS* 131
routines, *OS* 131-132
suite, *OS* 131-132
Drop rules, *VIEW* 59, 65
Drop Rules, *VIEW* 87
du command, *OS* 36

E

E drive
See DOS partition, physical, *DOS* 52
Editing MMDF configuration files, *OS* 297
Effective buffer size, *NET* 100
EGP (Exterior Gateway Protocol), *NET* 21
enable command (lineprinter), *OS* 219
Enabling a disabled account, *OS* 102
Enabling a disabled terminal, *OS* 103
encryption in security, *OS* 95

Environment variables
described, *DATA* 59
installation-wide, *DATA* 61
setting, *DATA* 8
user defined, *DATA* 62
Equivalence, *NET* 12, 22
Error log, *DATA* 45
Error messages
Account is disabled, *OS* 102
Audit: file system is getting full, *OS* 104
Authentication database ...
inconsistency, *OS* 105
Cannot obtain database information, *OS* 103
Can't rewrite terminal control entry, *OS* 104
login, *OS* 101
status, *NET* 152
sysadmsh screen, *OS* 9
Terminal is disabled, *OS* 103
/usr/adm directory, *OS* 186
You do not have authorization to run, *OS* 105
Escape key, *OS* 4
/etc directory, *OS* 183
/etc/auth/system/gr_id_map file, *OS* 102
/etc/auth/system/pw_id_map file, *OS* 102
/etc/auth/system/ttys file, *OS* 103
/etc/conf/cf.d/mtune
default values, *NET* 84
filesharing parameters, *NET* 83
/etc/conf/pack.d directory, *OS* 132
/etc/default directory, *OS* 145, 184, 186, 270
/etc/ftpusers, *NET* 12, 23
/etc/group
adding new computers, *NET* 4
sample entries, *NET* 5
/etc/hosts, *NET* 11, 54
/etc/hosts.equiv, *NET* 12, 22
/etc/motd
free space reminder, *OS* 35
/etc/motd, *OS* 184
/etc/named.pid, *NET* 54
/etc/passwd
adding new computers, *NET* 4
identification and authentication, *OS* 49
passwd(C), *NET* 4
/etc/passwd *(continued)*
sample entries, *NET* 5
user IDs, *NET* 3
/etc/rc, *OS* 184
/etc/rc0, *OS* 184
/etc/rc2, *OS* 184
/etc/rc.d/6
xnet.6, *NET* 81
xnet.6, net start rdr, *NET* 80
/etc/resolv.conf, *NET* 49
/etc/systemid, *NET* 80
/etc/termcap, defined, *OS* 184
Ethernet, defined, *NET* 157
ETRUNC parameter, *OS* 96
Events, window manager, *VIEW* 44
execsuid authorization, *OS* 53, 175
Extenddb function, *DATA* 53
Exterior Gateway Protocol (EGP), *NET* 21

F

Fault
alerting, *OS* 257
recovery, *OS* 258
fdisk command
partition
hard disk, *OS* 138
table, *OS* 139
with UNIX and DOS, *OS* 137
fids, MFPVC, *NET* 95
File
backups. *See* Filesystem, backups
block size, *OS* 36
damage, *OS* 42
data loss, *OS* 38
defaults, *VIEW* 51
inaccessibility, *OS* 38
organization, *VIEW* 65
removing, *OS* 35
repairing, *OS* 42
restoring, *OS* 121
rule, *VIEW* 51
sharing, *NET* 10
system. *See* Filesystem

File Control database, *OS* 99
File GID creation, *OS* 96
File ownership information, *NET* 4
File permissions, changing with DOS ATTRIB command, *DOS* 18
Filename
device special files, *OS* 189
truncation, ETRUNC parameter, *OS* 96
Files in NFS, *NET* 62
Filesharing parameters
changing, *NET* 84
/etc/conf/cf.d/mtune file, *NET* 83
Filesystem
See also Backups
automatic check, *OS* 43
backups
defined, *OS* 107
floppy disk labeling, *OS* 118
frequency, *OS* 107
listing procedure, *OS* 120
media storage, *OS* 115
restoring, *OS* 121
scheduled, *OS* 114
unscheduled, *OS* 117
verification procedure, *OS* 119
cleaning, *OS* 24
copies, *OS* 107
damage, *OS* 38
data loss, *OS* 38
defined, *OS* 33
maintenance, *OS* 34
partitioning, *OS* 331
repair, *OS* 38
restoring, *OS* 124
root, defined, *OS* 33
scheduled backup, *OS* 114
space
displaying free space, *OS* 34-35
lack of, *OS* 35
maintaining, *OS* 35
unmounting (umount command), *OS* 34
finddbs command
analyze mode, *DATA* 71
replace mode, *DATA* 71
using, *DATA* 71
Fixed disk, removing DOS applications from, *DOS* 55
Floppy disk
block size, *OS* 190
bootable floppy disk, *OS* 275
damage, *OS* 38
Emergency Boot Floppy Set, *OS* 275
mounting as DOS, *DOS* 15
root filesystem disk, *OS* 275
security, *OS* 45
virtual, *DOS* 24
Flow control, defined, *NET* 157
Focus, keyboard input, *VIEW* 7
Free space, *OS* 35
fsck command, *OS* 38
ftp account, *NET* 12, 23
Functions, window manager, *VIEW* 33

G

Gap number, *OS* 190
Gateway
defined, *NET* 8, 157
machines, *NET* 21
smart, *NET* 20
gethostbyname call, *NET* 54
getty process, disabling, *DOS* 11
gr_id_map file, *OS* 102
Group ID
described *OS* 305
login, requirements for network, *NET* 4
Group name, unique, *NET* 3
Groups
adding, changing, *OS* 164
maximum number of, *OS* 164
Guest ftp account, *NET* 12

H

haltsys command, *OS* 30
Hard disk
  adding another, *OS* 315
  block size, *OS* 190
  damage, *OS* 38
  mounting, *OS* 335
  partitions
    assigning, *OS* 140
    installing UNIX and DOS, *OS* 142
    two disks, *OS* 143
  tracks, *OS* 138
Hashed MMDF database, *OS* 310
Hayes modem with UNIX, *OS* 207
Hiding machines, *OS* 298, 308
Highlighting, menu option, *OS* 9
Home directory
  user account, *OS* 151
Host
  defined, *NET* 5, 158
  information resource record, *NET* 44
  intelligent, *OS* 301-303, 307
  intermediate, *OS* 307
  MMDF routing files, *OS* 306
  transparent, *OS* 298
Hosts database, *NET* 11
hosts file, *NET* 50
hosts.equiv file, *NET* 12, 22
hosts.rev file, *NET* 51
hwconfig command, *OS* 31

I

IAB, defined, *NET* 158
IBM PC-Network software
  compatible, *NET* 10
ICMP, defined, *NET* 158
Icons
  configuration. *See* Desktop Manager configuration.
Icons *(continued)*
  creating, *VIEW* 59, 79
  drop rules. *See* Drop Rules.
Identification and authentication,
  /etc/passwd, *OS* 49
IEN, defined, *NET* 158
ifconfig
  commands, *NET* 18
  netmask option, *NET* 19
II_AUTHORIZATION, *DATA* 63
iibuild command, *DATA* 5
II_CHECKPOINT
  defined, *DATA* 11
  location, *DATA* 11, 63
II_CONFI, *DATA* 63
II_DATABASE, *DATA* 10
II_DATE_FORMAT, *DATA* 63
iidbdb, *DATA* 49
  database database, *DATA* 49
  location, *DATA* 10
  when destroyed, *DATA* 72
II_DBMS_SERVER, *DATA* 19, 36, 64
II_DECIMAL, *DATA* 64
II_DML_DEF, *DATA* 64
II_FILES, *DATA* 65
II_HELP_EDIT, *DATA* 65
II_JOURNAL *DATA* 65
  defined, *DATA* 11
  location, *DATA* 11
II_LG_MEMSIZE *DATA* 65
II_LOG_FILE *DATA* 66
  location, *DATA* 9
  size, *DATA* 9
II_MONEY_FORMAT, *DATA* 66
II_MONEY_PREC, *DATA* 66
iimonitor utility, *DATA* 19-21
II_MSG_TEST, *DATA* 66
II_PATTERN_MATCH, *DATA* 66
II_PRINTSCREEN_FILE, *DATA* 67
iirundbms, *DATA* 32
II_SORT, *DATA* 67
II_SQL_INIT, *DATA* 48, 67
iistartup command, *DATA* 5, 31
II_SYSTEM, *DATA* 8, 67

II_TEMPLATE, *DATA* 67
II_TERMCAP_FILE, *DATA* 67
II_THOUSANDS, *DATA* 67
II_TMPDIR, *DATA* 67
Images, DOS, *DOS* 31
-imm_shutdown, *DATA* 41
include alias syntax, *OS* 305
inetd command, *NET* 21
ING_EDIT, *DATA* 67
ingprenv command, *DATA* 19,48
ING_PRINT, *DATA* 68
ING_SET, *DATA* 68
ING_SET_DBNAME, *DATA* 68
ingsetenv command, *DATA* 48,61
ING_SHELL, *DATA* 68
ING_SYSTEM_SET, *DATA* 68
Initializing
  cache, *NET* 40
  software, *NET* 84
INIT_INGRES, *DATA* 69
inittab file, *NET* 121
Installation
  shutdown, *DATA* 41
  utility, *DATA* 5
  script, xnet.6 file, *NET* 81
Installing
  a device driver, *OS* 130
  computers, *NET* 79
  with the custom program, *NET* 79
Installing DOS application programs
  on DOS partition, *DOS* 52
  on fixed disk, *DOS* 38-45
  under DOS without ODT-DOS, *DOS* 53-54
  with dosadmin command, *DOS* 37-50
Intelligent host, *OS* 301-303, 307
Interface
  configuration, *NET* 18
  display, *NET* 27
  options, setting, *NET* 18
Intermediate host, *OS* 307
Internal memory, adding, *OS* 281
Internet
  broadcast addresses, *NET* 19
  daemon, *NET* 21
Internet *(continued)*
  defined, *NET* 5,158
  protocol, *NET* 6
Internet address, defined, *NET* 158
Internetworking, defined, *NET* 158
Interrupt
  key, *OS* 4
  priority level, *OS* 132
  vectors, *OS* 133
IP, defined, *NET* 6

J

Journaling, *DATA* 11
Jumpers, bus card, *OS* 279

K

Kernel
  authorizations, *OS* 53,175
  configuration, *NET* 13
  data structures memory allocation, *NET* 83
  ETRUNC parameter, *OS* 96
  linking, *OS* 129
Kernel Configuration Module, *VIEW* 125
Key bindings, *VIEW* 5,48
Key disk copy protection
  installing applications that use, *DOS* 51
Keyboard, *OS* 4
Kill key, *OS* 4

L

LAN Manager Client, changing
  parameters, *NET* 84
LAN Manager/X, *NET* 10
Languages, supported by the Desktop Manager, *VIEW* 112
Layer, defined, *NET* 158
Length of network names, *NET* 3
/lib directory, *OS* 186

Line continuation, *OS* 305
Lineprinter
  accept command, *OS* 225
  adding, sysadmsh, *OS* 211
  disable command, *OS* 219
  dumb model interface, *OS* 244
  enable command, *OS* 219
  interface programs, *OS* 244
  lpmove command, *OS* 225
  lpsched program, *OS* 217
  reject command, *OS* 225
Linking the kernel, *OS* 129, 135
Links, UNIX, deleting DOS application programs, *DOS* 55
List
  channel, *OS* 300, 302, 308
  domain, *OS* 303, 307
  processor program, *OS* 308
  type alias, *OS* 304
list.chn file, *OS* 308
list-request alias, *OS* 302
Local
  area network, *NET* 10
  channel, *OS* 300
  machine name, *OS* 298, 308
  subnetworks, *NET* 18
  user ID number, file ownership information, *NET* 4
local.chn file, *OS* 307
local.dom file, *OS* 306
Locationname, *DATA* 53
  adding, *DATA* 55
  function, *DATA* 54
  in catalog function, *DATA* 59
  modifying, *DATA* 57
Lock lookups, *DATA* 39
Locked Files, *VIEW* 89
Locking
  a terminal, *OS* 151
  an account, *OS* 151
  configuration, *DATA* 39
  parameters, *DATA* 39
Log files, *DATA* 45-46
  MMDF, *OS* 303, 305
  troubleshooting, *DATA* 45-46
Logging
  configuration, *DATA* 37, 39
  defined, *DATA* 2
  parameters, *DATA* 37
  size of file, *DATA* 9
  system status, *DATA* 26, 30
Login
  error messages, *OS* 101
  restrictions
    changing, *OS* 168
    default, *OS* 56, 164
Logstat command
  functions, *DATA* 26, 30
  using, *DATA* 26, 30
Loopback interface, defined, *NET* 158
lp authorization, *OS* 172
LP print service, *OS* 217
lpmove command, *OS* 225
lpsched program, *OS* 217
LUID, *OS* 57

M

Machine name
  /etc/systemid file, *NET* 80
  mkself, *NET* 80
  MMDF, *OS* 297
Mail
  exchanger resource record, *NET* 47
  group member resource record, *NET* 47
  list, *OS* 300-308
  rename resource record, *NET* 46
  router, *OS* 289
  /usr/spool directory, *OS* 186
Mailbox
  file, *OS* 299
  information resource record, *NET* 46
Maintenance, administrative roles, *OS* 1
Major device number, *OS* 133-134
Mandatory auditing, *OS* 64

Master
files, *NET* 14
servers, *NET* 2,34,159
time daemon, *NET* 57
Matching domains, *OS* 303,307
MAXVCS
default value, *NET* 85
defined, *NET* 87
MBPVC
default value, *NET* 94
defined, *NET* 95
MCHANLOG, *OS* 303
MCHN entry, *OS* 300
MCONVCS
default value, *NET* 94
defined, *NET* 95
MDLVRDIR, *OS* 299
MDMN entry, *OS* 302
mem authorization, *OS* 172
Memory
adding internal, *OS* 281
allocation, kernel data structures, *NET* 83
parity errors, *OS* 282
removing internal, *OS* 282
Menu
options, *OS* 9
panes, configuring, *VIEW* 47
Menu Line (sysadmsh screen), *OS* 8
Menu panes, *VIEW* 5
Message Files
described, *VIEW* 111
format, *VIEW* 114
MFPVC
default value, *NET* 94
defined, *NET* 95
Micnet, configuring with MMDF, *OS* 312
micnet.chn file, *OS* 308,312
micnet.dom file, *OS* 312
MICOM-Interlan driver, *NET* 16
Microsoft LAN Manager software
compatible, *NET* 10
Minor device number, *OS* 134
mknod command, *OS* 131,134
mkself utility, *NET* 80
MLDOMAIN, *OS* 297,308
MLNAME, *OS* 297,308
MLOCMACHINE, *OS* 298,308
MMBXNAME, *OS* 299
MMBXPROT, *OS* 299
MMDF
address parsing, *OS* 301
ALIAS entry, *OS* 300
alias
examples, *OS* 305
line continuation, *OS* 305
converting files, *OS* 312
editing files, *OS* 304
ap parameter, *OS* 301
badhosts channel, *OS* 301,303,308
channel
defined, *OS* 300
directory, *OS* 302
.chn file, *OS* 307
configuration files
converting, *OS* 311
editing, *OS* 297
confstr parameter, *OS* 302
database, *OS* 310
deliver program, *OS* 302-303,310
directories, *OS* 299-303
.dom file, *OS* 306
domain
defined, *OS* 302
name, *OS* 297
hiding machines, *OS* 298,308
host. *See* Host
local
machine name, *OS* 298,308
routing, *OS* 300,306-307,312
log files, *OS* 303,305
mailbox file, *OS* 299
mailing list, *OS* 300-308
MCHN entry, *OS* 300
MDLVRDIR, *OS* 299
MDMN entry, *OS* 302
Micnet configuration, *OS* 308,312
mmdftailor file, *OS* 297
MTBL entry, *OS* 300

MMDF *(continued)*
partial domain matching, *OS* 303
pgm parameter, *OS* 302
pipe (|) redirection, *OS* 304-305
postmaster, *OS* 299,305
queue, *OS* 301
redirection alias, *OS* 304-305
relaying mail, *OS* 307
root domain, *OS* 302,303,307
routing example, *OS* 309
routing files
converting, *OS* 312,314
editing, *OS* 306
show parameter, *OS* 300-302
slash (/) redirection, *OS* 305
submit program, *OS* 301,303
support address, *OS* 299
system maintenance, *OS* 310
table
defined, *OS* 299
directory, *OS* 299
transport address, *OS* 306,308
troubleshooting, *OS* 310
undeliverable mail, *OS* 299
user-to-machine mapping, *OS* 305
UUCP configuration, *OS* 306,308,314
mmdfalias conversion utility, *OS* 312
mmdftailor file, *OS* 297
MMSGLOG, *OS* 303
mnlist conversion utility, *OS* 312
mnt directory, mounted filesystems, *OS* 186
Mode, channel delivery, *OS* 302
Modem
Automatic Dial Modem, *NET* 133
configuring the Hayes Smartmodem 2400, *NET* 115
control, *NET* 120
Devices file, *NET* 132
Dialers file, *NET* 134
files, *NET* 108
installation, *NET* 111
local network switch, *NET* 135
login sequence, *NET* 128
send and receive calls, *NET* 112
Modem *(continued)*
serial lines, *NET* 112
telephone line, *NET* 103
testing, *NET* 116,140
troubleshooting, *NET* 140
usage
available serial lines, *OS* 195
Devices file, *OS* 196
dialer programs, *OS* 196
dialing in, *OS* 202
dialing out with cu, *OS* 196
Hayes settings, *OS* 207
UUCP, use of modem under, *NET* 111
variable rate, *NET* 116
Modes of operation, defined, *OS* 25
Modifying stune, *NET* 84
Monitors, changing, *VIEW* 137
Motherboard cards, *OS* 279
Motif window manager, *VIEW* 4
Mounting a new filesystem, *OS* 333
Mounting a remote filesystem, *NET* 12,68
Mouse, *OS* 283
MSERVCS
default value, *NET* 94
defined, *NET* 96
msg.log file, *OS* 303
MS-NET software compatible, *NET* 10
MSUPPORT, *OS* 299
MTBL entry, *OS* 300
MTPVC
default value, *NET* 94
defined, *NET* 96
mtune file, default values, *NET* 84
Multiple drivers, *OS* 131-132
MultiScreen functionality, UNIX
ODT-DOS compatibility with, *DOS* 9
Multiuser mode, automatic activation, *NET* 81
.mwmrc
contexts, *VIEW* 41
defined, *VIEW* 5,33
sample file, *VIEW* 34
syntax
button bindings, *VIEW* 49
events, *VIEW* 44

.mwmrc *(continued)*
syntax *(continued)*
functions, *VIEW* 41
key bindings, *VIEW* 48
menu panes, *VIEW* 47
overall, *VIEW* 34
window manager
events, *VIEW* 44
functions, *VIEW* 36

N

N128
default value, *NET* 86
defined, *NET* 93
N16
default value, *NET* 86
defined, *NET* 92
N1K
default value, *NET* 86
defined, *NET* 93
N256
default value, *NET* 86
defined, *NET* 93
N2K
default value, *NET* 86
defined, *NET* 93
N4
default value, *NET* 86
defined, *NET* 92
N4K
default value, *NET* 86
defined, *NET* 94
N512
default value, *NET* 86
defined, *NET* 93
N64
default value, *NET* 86
defined, *NET* 93
NALIAS
default value, *NET* 85
defined, *NET* 87
Name resource record, canonical, *NET* 45
Name server
address resource record, *NET* 43
cache initialization, *NET* 40
caching-only server example, *NET* 48
canonical name resource record, *NET* 45
changing origin, *NET* 42
data files, *NET* 40
defined, *NET* 1,33
domain name pointer record, *NET* 45
host information resource record, *NET* 44
mail box resource record, *NET* 46
mail exchanger resource record, *NET* 47
mail group member resource record, *NET* 47
mail rename resource record, *NET* 46
mailbox information resource record, *NET* 46
master servers, *NET* 2,34
multiple files, *NET* 42
record, *NET* 43
remote, *NET* 39
resource record, *NET* 43
sample files, *NET* 48,52
SOA record, *NET* 42
starting, *NET* 54
types of, *NET* 2,34
using, *NET* 12
well known services record, *NET* 44
named program
debugging, *NET* 55
defined, *NET* 54
signals to reload, *NET* 55
named.local file, *NET* 50
NASEVENT
default value, *NET* 85
defined, *NET* 88
NBIDS
default value, *NET* 85
defined, *NET* 88
NBINDX
default value, *NET* 85
defined, *NET* 88

NBINDX *(continued)*
NBIOSIZ
  configuring for performance, *NET* 101
  default value, *NET* 94
  defined, *NET* 96
NBUFALLOC
  configuring for performance, *NET* 101
  default value, *NET* 94
  defined, *NET* 97
NCALLRETRY
  default value, *NET* 85
  defined, *NET* 89
NCONSTABLE
  default value, *NET* 85
  defined, *NET* 89
net start rdr
  custom utility, *NET* 80
  defined, *NET* 81
  /etc/rc.d/xnet.6, *NET* 80
  xfc on, *NET* 81
  xnet.6, *NET* 81
net stop rdr, defined, *NET* 82
NETDEV
  default value, *NET* 94
  defined, *NET* 96
netstat program, *NET* 20, 25
Network
  adding a computer, *NET* 80
  addresses, *NET* 8
  cable, *NET* 10
  compatibility, *NET* 3
  concepts, *NET* 2
  consumer, *NET* 79, 82
  databases, *NET* 22
  gateways, *NET* 8
  hardware configuration, individual, *NET* 10
  ID numbers, requirements, *NET* 4
  name, *NET* 3
  problems, *NET* 82
  processes, halt, *NET* 82
  protocols, *NET* 5
  resources, *NET* 10
  servers, *NET* 21
  starting, *NET* 81
Network *(continued)*
  system administrator responsibilities, *NET* 3
  transport hardware, *NET* 10
  troubleshooting, *NET* 25
Network buffer
  adjusting size, *NET* 100
  auto configuration, *NET* 101
  checking size, *NET* 100
  effective size, *NET* 100
  maximum size, *NET* 101, 102
  relation to streams buffer, *NET* 101
Network computers, defined, *NET* 80
Network interface, defined, *NET* 159
Network mask, defined, *NET* 159
Network parameters
  changing, *NET* 84
  defined, *NET* 83
  /etc/conf/cf.d/mtune, *NET* 84
  /usr/lib/xnet/xnetrc, *NET* 81, 94
NEXS
  default value, *NET* 85
  defined, *NET* 89
NEXTMAJOR, *OS* 133
NFS
  adding users, *NET* 74
  daemons, *NET* 63
  debugging, *NET* 65
  defined, *NET* 61
  described, *DATA* 75
  files, *NET* 62
  incompatibilities with remote filesystems, *NET* 74
  roles of UNIX systems, *NET* 61
  setting up clients, *NET* 12, 63
NFSBUFS
  default value, *NET* 85
  defined, *NET* 89
NFSTREAM
  default value, *NET* 85
  defined, *NET* 90
NGROUPS parameter, *OS* 164
NNCB_DEV
  default value, *NET* 85
  defined, *NET* 90

NNCB_NAMES
default value, *NET* 85
defined, *NET* 90
NNCBS
default value, *NET* 85
defined, *NET* 90
Nobypass, alias table characteristic, *OS* 300
Node, defined, *NET* 5
Non-cloning drivers, *NET* 17
NORDONLY
default value, *NET* 94
defined, *NET* 97
Normal operation
mode, *OS* 25
stopping, *OS* 29
NPTE
default value, *NET* 85
defined, *NET* 90
NQUEUE
default value, *NET* 86
defined, *NET* 91
NRECYCLE
default value, *NET* 85
defined, *NET* 91
NSTREVENT
default value, *NET* 86
defined, *NET* 91
NTIDS
default value, *NET* 85
defined, *NET* 91
NVCSUSED
default value, *NET* 94
defined, *NET* 95
NWB
default value, *NET* 85
defined, *NET* 92

O

ODT-DATA
configuring, *DATA* 7,11
introduction, *DATA* 2
system administrator, *DATA* 1
ODT-DOS
changes that affect users, *DOS* 35-36
list of files, *DOS* 35
system backup, *DOS* 15
On-card ROMS
and DOS images, *DOS* 32,35
Operating system, loading, *OS* 24
Outdated mail files, *OS* 310
override login, *OS* 56,101

P

Packet trace, *NET* 25
Parameters
DBMS server, *DATA* 32
default values, *NET* 94
Parity errors, memory, *OS* 282
Parsing MMDF addresses, *OS* 301
Partial domain matching, *OS* 303
Partition
DOS, *DOS* 16
hard disk
assigning, *OS* 140
deleting, *OS* 144
DOS, *OS* 144
fdisk command, *OS* 139
installing UNIX and DOS, *OS* 142
two hard disks, *OS* 143
table, *OS* 139
passwd(C), /etc/passwd, *NET* 4
Password
activity reports, *OS* 177
aging, *OS* 55
changing, *OS* 161
dial-in lines, *OS* 106
expiration, *OS* 170
restrictions
changing, *OS* 169
default, *OS* 164
super user, *OS* 3
Perform actions, protocols, *NET* 10
Performance, changing parameters, *NET* 84
Permission, mailbox file, *OS* 299

Permission modes, UNIX
on DOS partition, *DOS* 17
pgm parameter, *OS* 302
Picture file
background patterns, *VIEW* 97
control patterns, *VIEW* 97
editing, *VIEW* 59
example, *VIEW* 101
overview, *VIEW* 97
syntax, *VIEW* 99
Pipe (|) redirection, *OS* 304-305
Port, defined, *NET* 159
Postmaster, *OS* 299,305
Primary master server, defined, *NET* 159
Primary master server, example file, *NET* 48
Print service
fault alerting, *OS* 257
fault recovery, *OS* 258
LP, starting and stopping, *OS* 217
starting manually, *OS* 218
stopping manually, *OS* 218
print streams, *DOS* 13
Print wheel
alerting to mount, *OS* 255
defined, *OS* 253
Printer. *See* Lineprinter
Printer types, *OS* 251
printerstat authorization, *OS* 173
Printing
adding DOS printers, *DOS* 13
configuring default DOS printer, *DOS* 12
default attributes, *OS* 260
default spooler, *DOS* 12
determining configuration, *DOS* 12
disabling UNIX, *DOS* 14
DOS printer output, *DOS* 13
enabling UNIX, *DOS* 14
scheduler, *OS* 217
selecting print streams, *DOS* 13
sending output to printer, DOS, *DOS* 13
without using the UNIX spooler, *DOS* 14
printqueue authorization, *OS* 173
Problems, network, *NET* 82
Process
defined, *NET* 159
inheriting parent GID, *OS* 96
Protected databases, *OS* 97
Protected Password database, *OS* 98
Protected subsystems, *OS* 49,52
Protection, mailbox file, *OS* 299
Protocol
communications, *NET* 5
defined, *NET* 159
LAN Manager/X, *NET* 10
layering, *NET* 7
requests, *NET* 10
statistics display, *NET* 30
Public ftp account, *NET* 12
pw_id_map file, *OS* 102

Q

quel command, *DATA* 47
queryspace authorization, *OS* 173
Queues, MMDF
cleaning, *OS* 310
directory, *OS* 301
status, *OS* 310
quot command, block ownership display, *OS* 37
Quotation marks, *OS* 305

R

Raw
device, *OS* 135,10
log file, *DATA* 10
rcpconfig command, *DATA* 37,41
Read ahead parameter (constable file), *NET* 98-99
Read window parameter (constable file), *NET* 98-99
Reada timeout parameter (constable file), *NET* 98
Reado parameter (constable file), *NET* 98

Rebuilding the MMDF database, *OS* 310
Recovery
defined, *DATA* 2
finddbs command, *DATA* 71
log, *DATA* 46
rolldb command, *DATA* 71
system, *DATA* 71, 73
Redirection alias, *OS* 304, 305
Registered domain name, *OS* 298
reject command (lineprinter), *OS* 225
Relaying mail.
See Intelligent host;
See Intermediate host;
See Top-level domain name.
Relinking the kernel, *OS* 129, 135
Remote name servers, *NET* 39
Remote network system, *NET* 103
Removing
a user, *OS* 151
DOS application programs from fixed disk, *DOS* 55
Removing DOS application programs, *DOS* 54
Replace mode (finddbs command), *DATA* 71
Requests
sent by consumer, *NET* 10
sent to server, *NET* 10
Resource
classes, *VIEW* 10
description file.
See .mwmrc
See .Xdefaults
instances, *VIEW* 10
records, *NET* 40
specifications, window, *VIEW* 5
Responsibilities, database
administrator, *DATA* 1
Restricting
computer names, *NET* 80
file access, *NET* 12
RFC, defined, *NET* 159
RFC822-style addresses, *OS* 301
RFC919, *NET* 19
RGB database, *VIEW* 134
rgb.txt file, defining screen colors in, *VIEW* 134
.rhosts file, *NET* 22
rolldb command, *DATA* 71
ROMS, *DOS* 32, 35
root
defined, *NET* 159
directory, *OS* 181
filesystem backup, *OS* 107
override login, *OS* 56, 101
super user login name, *OS* 28
symbol (/), *OS* 33
Root domain, *OS* 302-303
root.cache file, *NET* 49
root.dom file, *OS* 307
routed(ADMN) program, *NET* 20
Routing
default, *NET* 20
example, *OS* 309
files
converting MMDF, *OS* 312-314
editing MMDF, *OS* 306
table
defined, *NET* 160
display, *NET* 28
management daemon, *NET* 20
table, *NET* 12
wildcard, *NET* 20
rs-232, *NET* 112, 114
Rule files, *VIEW* 51

S

Scan Windows, *OS* 18
Screen colors, customizing, *VIEW* 133
sdfile, location of, *DOS* 50
Search path, defining, *DOS* 47, 49
Secondary master server
defined, *NET* 160
example file, *NET* 49
Security
authck program, *OS* 99
defaults, *OS* 164
Defaults, *OS* 167
error messages, *OS* 101

Security *(continued)*
/etc/auth/system/ttys file, *OS* 103
integrity program, *OS* 100
kernel authorizations, *OS* 53, 172
override login, *OS* 56, 101
parameters, *OS* 164
sticky bit on directories, *OS* 92
subsystem authorizations, *OS* 50, 172
system integrity, *OS* 96
Semaphores, reporting, *DATA* 22
Serial
line, device special filenames, *OS* 191
ports, *OS* 193
Server
and DBMS, *DATA* 19-21
computers, *NET* 10
creating, *DATA* 31, 37
defined, *NET* 160; *DATA* 2
requests, *NET* 10
X Window System, *VIEW* 4
Session interface, *NET* 10
Sets, character, *OS* 253
Setting interface options, *NET* 18
Shared memory, limits, *DATA* 7
Sharing network resources, *NET* 2
show parameter, *OS* 300-302
Shut down consumer, xfc off, *NET* 82
Shutdown, *DATA* 41
command, *OS* 29
emergency, *DATA* 42
improper shutdown, *OS* 43
procedure, *DATA* 29, 41, 43
utility, *DATA* 6
shutserver
command, *DATA* 6
utility, *DATA* 41
Slash (/)
MMDF alias redirection, *OS* 305
root symbol, *OS* 33
Slave server, defined, *NET* 160
slink
functions
cenet, *NET* 16
denet, *NET* 17
uenet, *NET* 17
program, *NET* 16
Smart gateway, *NET* 20
Snapshots, *DOS* 31
SOA (Start of Authority) record, *NET* 42
Socket, defined, *NET* 160
SO_DEBUG option, *NET* 25
sql command, *DATA* 47
STACKS command, *DOS* 33
Standard resource record format, *NET* 40
Standard ROMS, and DOS images, *DOS* 32, 35
Start computer as consumer, *NET* 81
Start up commands, *NET* 80
Starting the LAN Manager Client network,
UNIX-based computer, *NET* 81
Starting the print service, manually, *OS* 218
Starting the system, *OS* 23
Startup files *DATA* 47
.startxrc, *VIEW* 8
Status Line (sysadmsh screen), *OS* 8
Stopping
computer as consumer, net stop rdr, *NET* 82
consumer, xfc abort, *NET* 82
network functions, *NET* 82
the print service, manually, *OS* 218
the system, *OS* 29
Streams
buffers, *VIEW* 129
changing parameters, *VIEW* 127
displaying current settings, *VIEW* 125
overview, *VIEW* 125
pipes, *VIEW* 130
queues, *VIEW* 128
total number of, *VIEW* 128
STREAMS
buffer, relation to network buffer, *NET* 101
configuring, *NET* 16
data blocks, *NET* 92
module, *OS* 134
tuning, *NET* 25

String delimiting, *OS* 305
une file, modification, *NET* 84
su authorization, *OS* 172
submit program, *OS* 301,303
Subnetworks, *NET* 18
SUBST command, DOS, *DOS* 54
Subsystem
  authorizations, *OS* 172-173
  database, *OS* 99
  defined, *OS* 52
  sysadmsh selections, *OS* 52
SUID/SGID/sticky bit clearing on files, *OS* 91
ummary of commands
  administrative, *OS* 216
  user, *OS* 215
Super user
  account, *OS* 3,28
  authorizations, *OS* 174
  login name (root), *OS* 28
  password, restricted use, *OS* 3
  precautions, *OS* 28
  prompt (#), *OS* 28
Superuser, *DATA* 57
Support address, *OS* 299
Support Utilities
  for the Desktop Manager, *VIEW* 119
suspendaudit authorization, *OS* 60,175
Swap space, requirements, *DATA* 8
witch settings, *OS* 321
Switching operating systems, *OS* 140
switchkey command, ODT-DOS, *DOS* 10
switch-screen sequence, *DOS* 10
symbol.tbl (file), *DATA* 48
Synchronization, *NET* 57
sysadmin authorization, *OS* 172
sysadmsh program
  backups, *OS* 119-120
  Context Indicator, *OS* 8
  creating backups, *OS* 114
  Display Area, *OS* 9
  Error Messages, *OS* 9
  files, restoring, *OS* 121
  Menu Line, *OS* 8
  menu, options, *OS* 8
sysadmsh program *(continued)*
  performing unscheduled backup, *OS* 117
  printer selection, *OS* 211
  Status Line, *OS* 8
sysadmsh, user IDs, *NET* 3
System
  administration directory, *OS* 186
  administrator. *See* System administrator
  backup, *DOS* 15
  boot messages, *OS* 31
  cleaning the filesystem, *OS* 24
  consistency, *NET* 3
  disk storage, *OS* 33
  equivalence, *NET* 12,22
  maintenance
    account, *OS* 3
    defined, *OS* 1
    mode, *OS* 25,30
  recovery, *DATA* 71,73
  security, *OS* 45
  starting, *OS* 23
  stopping, *OS* 29
System administrator
  backups, *OS* 107
  duties, *OS* 1
  file access, *OS* 33
  filesystem, *OS* 34
  free space, *OS* 35
  super user account, *OS* 3
  system maintenance mode, *OS* 25
  user account creation, *OS* 151

T

Table, *OS* 299
Tailoring MMDF configuration files, *OS* 297
Tape drive
  configuring, *OS* 265
  /etc/default files, *OS* 270
  formatting, *OS* 273
  maintaining, *OS* 272
  using, *OS* 265
tar, default settings, *OS* 270

/tcb directory, *OS* 187
TCP
  defined, *NET* 6, 160
  reliable transmission, *NET* 6
TCP/IP, defined, *NET* 5
Temporary files, *OS* 35
TERM, *DATA* 50, 69
terminal authorization, *OS* 172
Terminal Control database, *OS* 99
Terminal is disabled, error message, *OS* 103
Terminals
  defined, *DATA* 50
  disabled, *OS* 103
  enabling, *OS* 103
  locking, *OS* 176
  unlocking, *OS* 176
TERM_INGRES, *DATA* 50, 69
tids, tree connect table entries, *NET* 96
Time daemon
  constraints, *NET* 58
  master, *NET* 57
  options, *NET* 59
time, setting system clock, *OS* 26
timed program, administration, *NET* 57
timedc command, *NET* 60
/tmp
  cleanup, *OS* 35
  directory, *OS* 186
Top-level domain name, *OS* 307
Transparent host, *OS* 298
Transport layer interface, *NET* 10
Tree connect table entries, tids, *NET* 96
Triggers
  drag, *VIEW* 64
  icon, *VIEW* 62, 84
  ids, *VIEW* 91
  mouse, *VIEW* 62, 84
  overview, *VIEW* 74, 89
Troubleshooting
  MMDF, *OS* 310
  network, *NET* 25
  server startup, *DATA* 37
  UUCP, *NET* 140
  with log files, *DATA* 45-46
trpt program, *NET* 25
Trusted
  alias table characteristic, *OS* 300
  applications, *OS* 62
  system, defined, *OS* 46
Trusted Computing Base, *OS* 47
ttys, ttys-t, ttys-o file, *OS* 103
Tuning STREAMS, *NET* 25
Types, printer, *OS* 251

U

UDP, defined, *NET* 160
umount command, *OS* 34
Undeliverable mail, *OS* 299
Unique network name, *NET* 3
unit select, *NET* 17
UNIX
  devices, *DOS* 15
  keyboard, *OS* 4
  print spooler, *DOS* 12-13
  removing partition, *OS* 144
  shell
    configuring DOS applications to run, *DOS* 49, 50
  UNIX-based computer, starting the network, *NET* 81
Unlocking
  a terminal, *OS* 151
  an account, *OS* 151
Updating the MMDF database, *OS* 310
User
  accounts
    adding, *DOS* 4
    deleting, *DOS* 4
  activity reporting, *OS* 176
  adding, *NET* 74; *OS* 152; *DATA* 49, 57, 59
  audit parameters, *OS* 163
  block ownership display, *OS* 37
  changing
    group, *OS* 160
    password, *OS* 161
  commands, *OS* 215

User *(continued)*
deleting, *DATA* 49, 58, 59
equivalence, *NET* 12, 22
function, *DATA* 57
ID, *NET* 3; *OS* 305
identification number, *NET* 3
in catalog function, *DATA* 59
locking an account, *OS* 159
modifying, *DATA* 58
name, unique, *NET* 3
password expiration, *OS* 162
removing, *OS* 158
unlocking an account, *OS* 159
User ID
changing, *NET* 3
requirements for networks, *NET* 4
sysadmsh Accounts selection, *NET* 3
User-to-machine mapping, *OS* 305
/usr directory, *OS* 186
/usr/lib/xnet/constable, *NET* 81
/usr/lib/xnet/rc.d, xnet.6, *NET* 81
/usr/lib/xnet/xnetrc
filesharing parameters, *NET* 94
network parameter file, *NET* 81
/usr/mmdf/chans directory, *OS* 302
/usr/mmdf/log directory, *OS* 303
/usr/mmdf/table directory, *OS* 299
/usr/spool/lp/model file, *OS* 244
/usr/spool/mail directory, *OS* 299
/usr/spool/mmdf/lock/home directory, *OS* 301
/usr/sys/conf/master, *NET* 14
/usr/sys/conf/unixconf, *NET* 14
uucico program
connecting, *NET* 107
debugging with, *NET* 141
file transfer, *NET* 107
link to remote computer, *NET* 107
master and slave modes, *NET* 109
protocol negotiation, *NET* 107
Systems file, *NET* 108
work files, *NET* 103
uucico program *(continued)*
UUCP
ACU (Automatic Calling Unit), *NET* 125, 140
administration, *NET* 137
cabling, *NET* 104, 110, 114
chat script
correcting, *NET* 128
defined, *NET* 124, 128
escape sequences in, *NET* 129
expect/send pairs, *NET* 127
subexpect/subsend pairs, *NET* 127
terminator on send string, *NET* 127
configuration files, *NET* 106
configuring
described, *NET* 118
with MMDF, *OS* 314
connecting a serial wire, *NET* 110
control files, *NET* 118
creating passwords, *NET* 122
cu program, *NET* 115, 128
daemons, *NET* 106
data files, *NET* 137
debugging, *NET* 140, 141
defined, *NET* 103
Devices file
adding dial-out entries, *NET* 132
and uucico, *NET* 107, 108
baud rates, *NET* 116
defined, *NET* 106
format, *NET* 132
LAN, *NET* 126
shared line, *NET* 121
tty entry, *NET* 117
Dialers file, *NET* 108
dial-in/dial-out, *NET* 118
dialing prefixes, *NET* 124
directories, *NET* 106
disable command, *NET* 110, 121
enable command, *NET* 121
error messages, *NET* 149
/etc/inittab, *NET* 121
execute files, *NET* 107, 137
filesystem, access to, *NET* 106, 130

UUCP *(continued)*
getty program, *NET* 121, 136
links
ACU, *NET* 133
described, *NET* 123
direct, *NET* 132
Local Area Networks, *NET* 132, 133
lock files, *NET* 109, 137
log files, *NET* 106, 137
login
entries, *NET* 122
IDs, *NET* 130
prompt, *NET* 128
script, *NET* 124, 128
LOGNAME entry, Permissions
file, *NET* 130
MACHINE entry, Permissions file, *NET* 130
modem
configuring the Hayes Smartmodem
2400, *NET* 115
connecting, *NET* 114
control, *NET* 120
described, *NET* 104, 127
dialing configuration, *NET* 112
installing, *NET* 111
serial lines, *NET* 112
testing, *NET* 116
variable, *NET* 116
node name, *NET* 123, 141
ownership of files, *NET* 120
passwd file, *NET* 122
passwords
creating, *NET* 122
Systems file, *NET* 106
Permissions file, *NET* 106, 107, 119, 130
protocols, *NET* 133
public directory, *NET* 106, 130
remote commands, *NET* 106, 131, 137
retry period, *NET* 125
rmail program, *NET* 130

UUCP *(continued)*
rs-232
connecting two computers, *NET* 110
installation, *NET* 104
null modem cable, *NET* 110
pin connectors, *NET* 110, 114
sample transaction, *NET* 108
security
access defined by login ID, *NET* 130
described, *NET* 119
login
IDs, *NET* 106, 130
serial lines
disabling, *NET* 121
installing, *NET* 112
shared dial-in/dial-out, *NET* 136
spool directory, *NET* 137
Systems file, *NET* 123-125
TM. (temporary data file), *NET* 137
troubleshooting, *NET* 104, 140
uucico program
connecting, *NET* 107
debugging with, *NET* 141
file transfer, *NET* 107
master and slave modes, *NET* 109
protocol negotiation, *NET* 107
uucp program, *NET* 103, 106
uudemon.clean, *NET* 137
uudemon.hour, *NET* 108
uuinstall program, *NET* 118
uulog command, *NET* 137, 142
uuname command, *NET* 142
uusched program, *NET* 107
uutry program, *NET* 117, 141
uux program, *NET* 103, 106, 137
uuxqt program, *NET* 107
work files, *NET* 137
uucp program, *NET* 103, 106
uucp.chn file, *OS* 308, 314
uucp.dom file, *OS* 306, 314
uudemon.clean, *NET* 137
uudemon.hour, *NET* 108
uulist conversion utility, *OS* 314
uulog command, *NET* 137, 142

uuname command, *NET* 142
uusched program, *NET* 107
uutry program, *NET* 117, 141
uux program, *NET* 103, 106, 137
uuxqt program, *NET* 107

V

vectorsinuse program, *OS* 133
Verify requests, protocols, *NET* 10
Video cards, changing, *VIEW* 137
Virtual diskette, *DOS* 19
Virtual DOS floppies, *DOS* 23
Virtual DOS partition, *DOS* 19

W

WAN, *NET* 11
Well known services resource record, *NET* 44
Wheel, print
  alerting to mount, *OS* 255
  defined, *OS* 253
Whitespace, *OS* 305-308
Wide-area network, *NET* 11
Window
  See also .Xdefaults and .mwmrc
  appearance and behavior, *VIEW* 1
  configuration files, *VIEW* 1,5
  frames
    components, *VIEW* 5
    configuration. *See* .Xdefaults and .mwmrc
  manager
    configuration. *See* .Xdefaults and .mwmrc
    defined, *VIEW* 4
    functions. *See* .mwmrc
  opening automatically at login, *VIEW* 8
  read, *NET* 97
  Scan Window, *OS* 18
  write, *NET* 97
Window manager
  colors, *VIEW* 133
  events, *VIEW* 44
Window manager *(continued)*
  functions, *VIEW* 33
Windows, configuration files, *VIEW* 1
Write window parameter (constable file), *NET* 98, 100
writeaudit authorization, *OS* 53, 60, 175

X

X Window System
  See also STREAMS.
  client, *VIEW* 4
  described, *DOS* 10
  server, *VIEW* 4
  STREAMS, *VIEW* 4
X.25, defined, *NET* 160
.Xdefaults
  classes, *VIEW* 10
  defined, *VIEW* 5
  resource groups, *VIEW* 10
  resources
    client specific, *VIEW* 27
    color values, *VIEW* 133
    component, *VIEW* 23
    specific, *VIEW* 13
  sample file
    color, *VIEW* 133
    monochrome, *VIEW* 131
  sample file, *VIEW* 11
  screen colors, *VIEW* 133
  syntax, *VIEW* 13, 23, 27
xenix file, *OS* 182
xfc abort, stops consumer, *NET* 82
xfc off, shuts down consumer, *NET* 82
xfc on
  net start rdr, *NET* 81
  starts computer as consumer only, *NET* 81
XITONCLOSE
  default value, *NET* 85
  defined, *NET* 92
xnet.6
  /etc/rc.d/6, *NET* 81
  net start rdr, *NET* 81

xnetrc file, configuring for
performance, *NET* 101

Y

You do not have authorization to run, error message, *OS* 105

Suggestions – Criticisms – Corrections

Are you happy with this manual? If so, let us know.
If not, help us improve it by informing us

- where you have noticed mistakes
- where the content is unclear.

From:

Name

Company/department

Address

Postal Code

Telephone ( )

Local Siemens office

Contact person

Siemens AG
K D ST QM2
Manualredaktion
Otto-Hahn-Ring 6
Postfach 83 09 51

D-8000 München 83

From:

Name

Company/department

Address

Postal Code

Telephone ( )

Local Siemens office

Siemens AG
K D ST QM2
Manualredaktion
Otto-Hahn-Ring 6
Postfach 83 09 51

D-8000 München 83

Hit a "stumbling block"? Tell us where.

Manual title: SINIX Open Desktop V1.0, U5746-J-Z95-1-7600

Page	Problem:

I am
- ☐ a programmer
- ☐ a system administrator
- ☐ an ordinary user
- ☐ _____

I use the manual
- ☐ frequently
- ☐ occasionally for reference
- ☐ _____

Manual title: SINIX Open Desktop V1.0, U5746-J-Z95-1-7600

Page	Problem:

I am
- ☐ a programmer

I use the manual
- ☐ frequently