HP 9000 Computers

DTC Device File Access Utilities and Telnet Port Identification

HEWLETT
PACKARD

HP Part No. B1030-90002
Printed in U.S.A. 01/95

Edition 2
E0195

Legal Notices

HEWLETT-PACKARD COMPANY
3000 Hanover Street
Palo Alto, California 94304 U.S.A.

Printing History

The manual printing date and part number indicate its current edition. The printing date will change when a new edition is printed. Minor changes may be made at reprint without changing the printing date. The manual part number will change when extensive changes are made.

Manual updates may be issued between editions to correct errors or document product changes. To ensure that you receive the updated or new editions, you should subscribe to the appropriate product support service. See your HP sales respresentative for details.

Edition 2 January 1995 (HP-UX Release 10.0)

Contents

Figures

1

Introduction

This manual explains how to use the DTC Device File Access Utilities (DDFA) and the Telnet Port Identification feature to communicate with HP 9000 serial devices connected to network terminal servers such as the HP Datacommunications and Terminal Controller (DTC). DDFA facilitates the use of pseudoterminal (pty) device files to open outgoing Telnet connections to the DTC or other terminal server, while Telnet Port Identification uses them to establish incoming Telnet connections from known devices on the server. Telnet is one of the HP 9000 Internet Services (HP B1030B), formerly known as the HP 9000 ARPA Services.

DDFA was originally designed for use with DTCs, but as of HP-UX 10.0, it can now be used with other terminal servers which use addressing schemes similar to the DTC. There will be some guidelines on how to configure DDFA for use with non-DTC terminal servers, but the DTC will be used as the primary example throughout this manual.

In addition, there is general discussion on the topic of pty use with HP DTC terminal servers versus the HP MUX, and how to troubleshoot incoming and outgoing DDFA connections.

Intended Audience

This manual is intended for at least three types of users:

- The HP 9000 system administrator or network administrator who defines and configures the device files associated with DTC and other terminal server devices on the system.
- The HP 9000 system operator or network operator who may implement the actual tasks set up by the system or network administrator.
- The applications programmer who needs programmatic control of devices on the DTC or other terminal server using standard HP-UX input/output calls such as **open**, **close**, **ioctl**, **read**, and **write**. This programmer may also have applications using devices connected to MUX ports and wishes to extend the application to use DTC devices.

Prerequisites

Before reading this manual and using this software, you should be familiar with how to:

- Handle HP-UX operating system and system administration, especially with devices and device files.
- Configure a DTC with either the DTC Manager/UX product (HP J2120A) or the OpenView DTC Manager product on the PC (HP D2355A).
- Configure and administer other terminal servers.
- Program devices and use device files for pseudoterminals (ptys) or MUXes.
- Access on-line DDFA man pages on your system like ddfa(7), dp(4), dpp(1M), ocd(1M), ocdebug(1M), as well as ioctl(2), ioctl(5), pcf(4), and termio(7).

Supported Configurations

There are several types of software and hardware products and services used in conjunction with DTC Device File Access Utilities. Refer to the “Related HP Documentation” section for more information.

- **HP-UX operating system version**. The DTC Device File Access Utilities are included with HP-UX version 9.0 and later. The DDFA Utilities consists of executable files, default configuration files and manual reference pages.
- **Internet Services.** DDFA is an extension of the Telnet service. Because Telnet is one of the Internet Services (HP B1030B), DDFA is automatically installed and requires Internet Services and the LAN Link to be configured and operating properly.
- **HP 9000 Series 700 and 800 systems.** The DDFA Utilities are supported only on HP 9000 Series 700 and 800 systems. Because these systems have Internet Services as part of their HP-UX system, these systems also have DDFA installed.
- **DTC Manager/UX or OpenView DTC Manager.** The HP DTCs which access an HP 9000 Series 800 are configured and managed by either the host-based *HP DTC Manager/UX* (HP J2120A) or the PC-based *OpenView DTC Manager* (HP D2355A). The PC-based OpenView DTC Manager also configures and manages DTCs which access HP 9000 Series 700 systems and HP 3000 systems. The version for DTC Manager/UX should be A.04.00 or later. The version of the OpenView DTC Manager should be A.14.10 or later. The version for DTC 16RX Manager/UX should be A.04.00 or later.
- **Printers and plotters.** DDFA Utilities can be used with the HP-UX spooler for printers and plotters for which a model script exists., and are supported for use with HP DTCs.

- **Terminals.** Any terminal supported for use with HP DTCs is also supported.
- **Modems.** Any modem supported for use with HP DTCs is also supported.
- **HP DTCs.** The following DTCs are supported on HP 9000 systems:
  - □ HP 2340A - DTC 16 with 16 ports.
  - □ HP 2360A - DTC 16TN with 16 ports.
  - □ HP 2363A - DTC 16MX with 16 ports.
  - □ HP 2364A - DTC 16RX with 16 ports.
  - □ HP 2345A - DTC 48 with 48 ports.
  - □ HP 2370A - DTC 72MX with 72 ports.
- **Other Terminal Servers.** DDFA Utilities can be used with non-HP terminal servers which use addressing schemes similar to the DTC. However, in order to work, the individual user must perform three tasks: (a) Configure the non-DTC terminal server ports for use with the DDFA product. (b) Configure DDFA for use with non-DTC ports, and (c) Test whether this particular terminal server configuration works correctly with DDFA before calling HP.

The user must follow the configuration guidelines discussed in this manual, as well as those given by the specific terminal server vendor in order for HP to support the DDFA interface.

Related HP Documentation

HP-UX System manuals:

- *HP-UX System Administration Tasks* (the HP part number is different for each of the HP 9000 systems)
- *Installing HP-UX 10.0* (the HP part number is different for each of the HP 9000 systems)

LAN and Internet Services manuals:

- *Installing and Administering Internet Services* (B1030-90000)
- *Installing and Administering LAN/9000* (98194-90050)
- *Using Serial Line IP Protocols* (98194-90051)

DTC Manager manuals:

- *Using HP OpenView DTC Manager* (D2355-90001) for the PC-based DTC manager.
- *Using HP OpenView DTC Manager/UX* (J2120-62000) for the host-based DTC manager for HP 9000s.
- *Using DTC 16RX Manager* (J2496-90000) for the DTC 16RX manager.

2

Overview of DDFA And Telnet Port Identification

This chapter gives a brief overview of DTC Device File Access (DDFA) Utilities and Telnet Port Identification. Read this chapter to learn how DDFA and Telnet Port Identification work, and how device files are used to communicate with devices attached to DTCs and other terminal servers. Chapters 3, 4, and 6 provide information on configuring, executing and troubleshooting DDFA connections to DTCs and other terminal servers.

This product enhances and supplements the Telnet protocol by providing the following benefits:

- **DDFA makes accessing devices attached to DTCs or other terminal servers like accessing MUX devices.** DDFA Utilities allow the system administrator to set up a correspondence between these DTC ports and HP-UX device files. With this correspondence defined, the system spooler or a user application can manipulate well-known device files to read and write to specific server ports.
- **DDFA allows the HP-UX Spooler for printers attached to DTCs or other servers to be configured in SAM.** After the correspondence between printers on a DTC and HP-UX device files has been set-up, SAM (System Administration Management tool) can be used to configure the spooler for DTC-connected printers as well as for MUX-connected printers. The only difference is that the pty device file name of the DTC printer must be used instead of a tty name for

a MUX printer. In fact, the standard spooler model scripts can be used with server printer(s).

- **DDFA allows user applications to access devices attached to DTC and other terminal servers using standard HP-UX system calls.** After the correspondence between DTC devices and HP-UX device files has been set up, user applications can use the standard HP-UX **read**, **write**, **open**, **close**, and **ioctl** calls. These calls access the devices by manipulating their corresponding device files.
- **Telnet Port Identification lets the system administrator ensure that incoming Telnet connections from specific DTC ports will be assigned to specific, rather than random pty device files.**

DDFA Utilities and Telnet Port Identification *cannot* be used simultaneously on the same device file, since they provide separate functionality. However, they may be used on the system at the same time.

How DDFA Utilities Work

The DTC Device File Access Utilities are a group of configuration files, executable files and one or more daemons. Together these utilities allow the HP 9000 system administrator to set up logical pairs of pty device files and physical ports on the DTC or other server. Once this is done, the server's pty devices can be accessed in the same way as MUX-connected tty devices can.

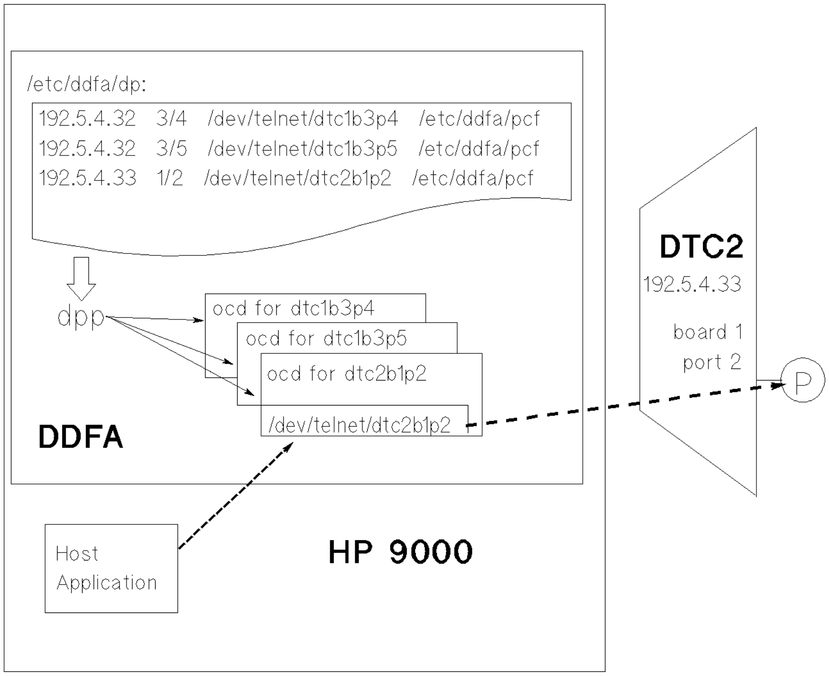
These ptys can be assigned to "incoming" connections or to "outgoing" connections. Incoming connections to the system are initiated by input devices attached to the server (such as terminals), while outgoing connections are initiated by the system to output devices attached to the server (usually printers). When an application on the system opens one of these device files, the DDFA Utilities transparently manage the creation of a Telnet connection to the associated server port and its device.

Configure and execute the DDFA Utilities in two simple steps:

1. Edit the Dedicated Port configuration file (**dp**) to add an entry for each physical terminal server port which describes the association to a unique pty device filename.
2. Run the Dedicated Port Parser program (**dpp**) to parse the dp file and to execute an **ocd** process for each outgoing connection defined in the dp file.

Each time the HP-UX system initiates an outgoing connection to a pre-defined terminal server port, its unique Outgoing Connection Daemon (**ocd**) becomes active. It then establishes the Telnet connection and manages it until the connection is closed.

Figure 2-1 shows how the system, the DTC terminal server, and the DDFA Utilities interact with one another.


/etc/ddfa/dp:
192.5.4.32 3/4 /dev/telnet/dtc1b3p4 /etc/ddfa/pcf
192.5.4.32 3/5 /dev/telnet/dtc1b3p5 /etc/ddfa/pcf
192.5.4.33 1/2 /dev/telnet/dtc2b1p2 /etc/ddfa/pcf
dpp
ocd for dtc1b3p4
ocd for dtc1b3p5
ocd for dtc2b1p2
/dev/telnet/dtc2b1p2
DDFA
Host
Application
HP 9000
DTC2
192.5.4.33
board 1
port 2
P

An outgoing dedicated port is defined in the dp file where a pty device file name is created for each dedicated port (in column 3).

The dedicated port parser (dpp) spawns an outgoing connection daemon (ocd) for each outgoing dedicated port.

When a host application accesses the pty device file specified in the dp file, a connection to the DTC device is initiated and communication can begin.

Figure 2-1. HP 9000 and DTC Interaction With DDFA

How Telnet Port Identification Works

Telnet Port Identification is made possible by a set of enhancements to the Telnet daemon (**telnetd**), which is part of Internet Services. In earlier versions of Telnet, incoming connections, including those coming from a DTC server, were always assigned pty device files on a random basis.

Internet Services Telnet allows the system administrator to set up pre-defined pty names defined in the DDFA dedicated port file, **dp**. In addition, the DTC download code was enhanced so that it will deliver board and port information to the host (via Telnet) at connection establishment time. The host will map the incoming connection to its pre-defined pty device file, thereby providing a "dedicated port" by which the identity of the caller can be determined. The Telnet Port Identification feature is available with HP-UX and DTCs, but may not be compatible on other servers.

How Device Files Are Handled By MUXes and DTC Terminal Servers

Recall that a device file is an HP-UX file that "points" to a system device. The system administrator often uses the name of the device file when configuring software to access that device. When devices are connected to an HP-UX MUX (multiplexer), they are assigned to tty device file names. When devices are connected to a DTC server, they are assigned to *random* pty device file names. To the user logging on at his terminal from either a MUX or a DTC, the terminal functions the same way. The user does not see how the device file is assigned to the connection, and whether the MUX driver or the terminal server driver is used.

The HP-UX System Administrator creates a device file for each MUX port, using the HP-UX **insf** or **mknod** command. Each device file maps to a specific physical

MUX port, and device files are named by convention, so that each identifies a unique MUX port. For example, **/dev/tty2p3** means port 3 on MUX card 2.

A device connected to a DTC or other terminal server communicates with the system via Telnet. Therefore, it is considered to be a logical device, and it is serviced by a **pseudoterminal device driver** (pty). It is referenced using its pty device file name.

Usually, this pty is assigned to the Telnet connection randomly from a pool of free ptys in the **/dev** directory or subdirectories at connection setup time. In many cases, the randomness of pty assignments for Telnet users is acceptable, because the physical location of the Telnet user is unimportant. In fact, users of system-to-system Telnet have always been subject to this situation. However, when a specific DTC device must always be associated with the same pty, then the randomness of pty assignments must be removed through a utility such as DDFA.

Starting with HP-UX 10.0, pty device files for incoming connections should be assigned to the directory **/dev/telnet** so that they can be more easily tracked and be correctly displayed by commands such as **ps -ef**.

Setting Up Outgoing and Incoming Connections

A print job sent from the system to a printer creates an outgoing connection. When a user logs in at a terminal on a DTC and receives a system prompt from the host to complete the login, an incoming connection is created. Both operations require the use of a device file name.

Whenever an application on the host needs to access a MUX device, the application can read and write to the tty device file that belongs to the MUX device. However, if an application wants to open a DTC device, difficulties arise, since no pty device file is assigned before the connection to the device is established. DDFA makes this connection setup process transparent to the application. All the application needs to do is to issue standard HP-UX **open()**, **read()**, **write()**, **close()** and **ioctl()** calls to the known pty.

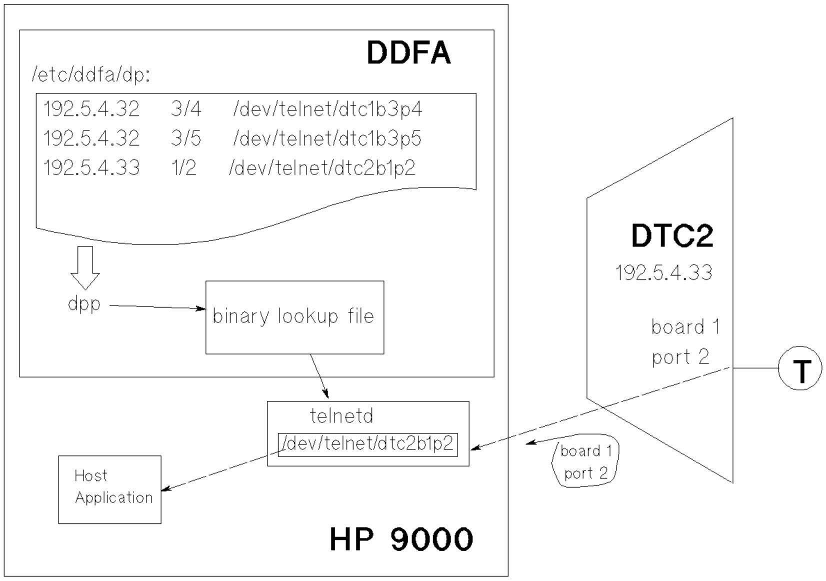
Whenever a MUX-connected terminal logs onto a system, the device file associated with the session is always the same. The user can find out what MUX port the terminal is connected to by typing the **tty** command. In the example below, the device file name is shown to be **/dev/tty2p3** for MUX card 2, port 3.

```
tty

/dev/tty2p3
```

The Telnet daemon (**telnetd**) assigns a pty to the connection when a user logs into the system from a terminal on a DTC. The pty device files refer to logical devices, and the Telnet daemon selects them randomly from the pool of free ptys in the **/dev** directory and its subdirectories. Even though you can use the HP-UX **who** or **tty** command to find the name of the device file associated with your Telnet session, the result does not show which DTC port is yours. The assigned pty is different each time you login, even from the same terminal.

Figure 2-2 illustrates the system and DTC interaction on an incoming connection with Telnet port identification. When the system accepts an incoming Telnet connection, it asks the calling DTC to give it the board and port numbers of the DTC port. If a mapping between the DTC port and a pty device file was defined in the **/etc/ddfa/dp** file, then the defined pty device file is used to service the incoming connection.



```
192.5.4.32    3/4    /dev/telnet/dtc1b3p4
192.5.4.32    3/5    /dev/telnet/dtc1b3p5
192.5.4.33    1/2    /dev/telnet/dtc2b1p2
```

An incoming dedicated port is defined in the dp file where a pty device file name is assigned for each dedicated port.

The dedicated port parser (dpp) scans the file and creates a binary lookup file for incoming dedicated ports.

When a DTC terminal logs into the system via Telnet, the Telnet daemon gets DTC board and port info from the DTC, and uses the binary lookup file to assign the defined pty to the connection.

Figure 2-2. Incoming Connection Using Telnet Port Identification

DDFA File Directories and Setup

DDFA is automatically installed along with Internet Services when your system receives HP-UX 10.0. There are four basic steps to setting up DDFA on your system:

1. **Use the installation/update utility on your system to install HP-UX and Internet Services.** Refer to the *Installing and Updating HP-UX, HP-UX System Administration Tasks*, and*Installing and Administering Internet Services* manuals. After installing HP-UX, the following DDFA filesets must be on your system:

   INETSVCS-DTC for DDFA

   /usr/sbin/dpp
   /usr/sbin/ocd
   /usr/sbin/ocdebug
   /usr/examples/ddfa/dp
   /usr/examples/ddfa/pcf

   INETSVCS-RUN for Telnet Port Identification

   /usr/lbin/telnetd
   Only **telnetd** is modified for Telnet Port Identification.

   INETSVCS-MAN for DDFA References

   /usr/share/man/man1m/dpp.1m
   /usr/share/man/man1m/ocd.1m
   /usr/share/man/man1m/ocdebug.1m
   /usr/share/man/man4/dp.4
   /usr/share/man/man4/pcf.4
   /usr/share/man/man7/ddfa.7

   Refer to the above manuals for information on installation and migration from releases prior to 10.0.

2. **If updating to HP-UX 10.0, update the location of older DDFA device files**. DDFA now supports the HP-UX 10.0 file layout conventions, as well as having pseudonym (pty) device files in a special

directory called **/dev/telnet**. A migration script (**/usr/contrib/bin/ddfa_device_mig**) has been provided to facilitate the movement of device files to this special directory.

3. **Configure the dedicated port (dp) files of DDFA.** See Chapter 3, Configuring DDFA Connections and Chapter 4, Defining and Executing DDFA Parameters.
4. **Configure the board and ports of the DTC(s).** The DTC must be configured and downloaded before actually making a connection. Refer to your DTC manual. If another terminal server is used, then configure it according to manufacturer's instructions, in conjunction with step 3 above.

The important point is that the DTC or other terminal server has certain parameters which must be configured to match with the corresponding ones in the system DDFA configuration. These must be downloaded before a connection is actually made.

Note that in releases prior to HP-UX 10.0, the Internet Services were called ARPA Services. The DDFA filesets used at that time were called ARPA-AUX, ARPA-RUN, and ARPA-AUX-MAN. Later they were relocated to reflect changes to Internet Services as well as the operating system's ATT System V.4 compliance.

3

Configuring DDFA Connections

This chapter explains the tasks necessary to configure DDFA for incoming and outgoing connections. Both primary and related configuration tasks involving DTCs and other terminal servers are described. The terms terminal server or server will be used to refer to either.

More details on individual parameters and explanations for DDFA configuration and execution are given in Chapter 4.

DDFA Master Files

DDFA includes the following configuration files, executable files, and daemon:

/etc/ddfa/dp The **Dedicated Port (dp)** configuration file is an ASCII file which contains the mapping information for each physical terminal server port and its associated pty device file.

/etc/ddfa/pcf The **Port Configuration File (pcf)** is an ASCII file which contains default port configuration parameters used by ocd processes. The **pcf** is referenced inside the **dp** file.

/usr/sbin/dpp The **Dedicated Port Parser (dpp)** is an executable file which parses the **/etc/ddfa/dp** file and spawns an Outbound Connection Daemon (**ocd**) for each outgoing and incoming connection specified in the **dp** file The **dpp** can be run manually from the Shell, or automatically each time the system is booted.

/usr/sbin/ocd The **Outbound Connection Daemon (ocd)** manages the connection and data transfer to each server port. Normally, **ocds** are spawned by the **dpp**. However, an **ocd** can also be run from the Shell with all the parameters from the **dp** file specified on the command line.

/usr/sbin/ocdebug The **Outbound Connection Daemon in debug mode** (**ocdebug**) is a special debugging version of **ocd** and performs the same tasks as **ocd**. In addition, the **ocdebug** daemon logs debug messages to the log file **/var/adm/ocd<pid>** for troubleshooting purposes. (The term '<pid>' refers to ocd's process identification number.)

After the system administrator runs the **dpp** program on the **dp** configuration file, an **ocd** daemon is created for each configured port for which an outgoing connection is desired. When the daemon is spawned, it takes a pty from the pty pool in the **/dev** directory (or its subdirectories). The daemon then creates a device file with the same major and minor number as the pty slave, and gives it the name listed in the **dp** file. The new device file is known as the "pseudonym".

User applications should use this pseudonym to access the server port when calling standard HP-UX intrinsics (such as **open**, **close**, **ioctl**, **read** and **write**). The daemon listens on the pty until an application does an **open** call using the pseudonym. The daemon manages the connection to the server port until it is closed. All of this activity is transparent when you use a terminal or printer on the DTC or other terminal server.

Related Configuration Tasks

This section assumes that the DTC or other terminal server is configured and can be accessed over the LAN before attempting to configure DDFA on the system. If this is not the case, then refer to the manuals for the DTC Manager/UX product (HP J2120A) or the OpenView DTC Manager product (HP D2355A) for instructions on how to configure the DTC. For non-DTC servers, refer to the manufacturer's documentation for information on configuration and addressing formats used. Once all of the server device ports are configured and downloaded, you will be ready to configure DDFA.

First, you must record some network information about the DTC or other terminal server, which will be required for the next section, DDFA Configuration Tasks. It includes:

- Node name used by the DTC or other terminal server, in the form **name.domain.organization.company**, or an alias name as configured in the /etc/hosts file or domain name server of the system.
- IP address used by the DTC or other terminal server.
- Port addressing information for each physical port on the server to be configured by DDFA. For a DTC port, this is its board number and port number, or its IP address. For a port on a non-DTC server, it may be its IP address or its TCP address.
- Port type (direct connection or modem).

- Connection type (outgoing or incoming connection).

DDFA Configuration Tasks

HP recommends keeping a backup of your HP-UX software in order to recover from potential problems in the future. In addition, be sure to read any READ ME FIRST files supplied with your system.

1. Log on as root or superuser to perform these DDFA configuration steps.
2. Check to see if a working directory for DDFA already exists.

   ```
   ls /etc/ddfa
   ```

3. If it does not, create a directory for the DDFA files. HP recommends **/etc/ddfa**.

   ```
   mkdir /etc/ddfa
   ```

4. Check to see if a dedicated port configuration file already exists.

   ```
   ls /etc/ddfa/dp
   ```

5. If it does not, copy the master template dedicated port file, **dp**, to the DDFA directory.

   ```
   cp /usr/examples/ddfa/dp /etc/ddfa/dp
   ```

   Do not alter **/usr/examples/ddfa/dp**, the master template **dp** file. Instead, modify the file **/etc/ddfa/dp** as explained below.

6. Copy the master template port configuration file, **pcf**, to the DDFA directory. Use this copy as the generic pcf file which works for most DTC and server devices.

   ```
   cp /usr/examples/ddfa/pcf /etc/ddfa/pcf
   ```

   Do not alter **/usr/examples/ddfa/pcf**, since that is the master template **pcf** file. Instead, you should go to the **/etc/ddfa** directory, and make one copy of the pcf template file for each type of pty device needed (printer, plotter, etc.). You should give the new **pcf** file in **/etc/ddfa** a name which describes the type of device (for example, **/etc/ddfa/laserjet**), and modify the contents as needed.

Configuring Outgoing Connections (Printers)

For each terminal server output device (such as a printer) , perform the following steps to assign a pty device file:

1. For a DTC, determine its Node Name, its IP address, the board number, and the port number on the DTC to which the device is connected.

   For a non-DTC server, determine its Node Name, its IP address and the TCP port number on the server to which the device is connected.

2. Define a device file name that you will use for this output device. It is helpful to select a name which describes the specific device port. For example, use the pty file name **/dev/telnet/dtc72b3p2** for a device on a DTC 72MX terminal server, board 3, port 2.

Note

Note the following physical port configurations and numbering conventions often recommended for DTCs: DTC 48s may have up to 6 asynchronous boards (numbered slots 0-5), and 6 modem ports (numbered 0-5) or 8 direct connect ports (numbered 0-7).

DTC 16s may have up to 2 asynchronous boards (numbered slots 0-1), and 6 modem ports (numbered 0-5) or 8 direct connect ports (numbered 0-7).

DTC 16TNs, DTC 16MXs and DTC 16RXs have a single asynchronous board (called slot 1 or 01); it supports 16 modem or direct connect ports (numbered 0-15).

DTC 72MXs may have up to 3 asynchronous boards, each of which will support 24 modem or direct connect ports (numbered 0-23).

It is important to check and note the DTC's board and port configuration, since the LAN board may not always be found in the default slot 0, and the asynchronous cards may be in any of slots 0-3.

---

3. Run the text editor of your choice, such as **vi**, to edit the **/etc/ddfa/dp** file.
4. Create an entry (one for each output device) in the **/etc/ddfa/dp** file using one of the following addressing formats:

   a. Formats 1 and 2 are used **only for DTC servers**. The slash (/) must separate the board and port parameters, which are unique to the DTC.

   ```
   <DTC IP addr>   <board>/<port> <pseudonym> <pcf>

   <DTC Nodename>  <board>/<port> <pseudonym> <pcf>
   ```

   b. Format 3 is accepted **by most terminal servers, including DTCs, for outgoing connections only**. The XX (**XX**) indicates a null value.

   ```
   <Server IP addr> XX/<port TCP addr> <pseudonym> <pcf>

   <Server Name>    XX/<port TCP addr> <pseudonym> <pcf>
   ```

   c. Formats 4 and 5 are used for addressing a server where the connection goes to the default TCP port address of 23; this feature is supported on DTCs and some other terminal servers.

   ```
   <Server IP addr>   XX/XX <pseudonym> <pcf>

   <Server Name>      XX/XX <pseudonym> <pcf>
   ```

5. Repeat steps 1-4 for each outgoing connection device until all entries are configured.

6. Save the modified file **/etc/ddfa/dp** file. It is now ready for parsing by **dpp**.

Configuration Example 1 - Let's configure a printer port which is on board 3 port 2 of a DTC 72MX whose IP address is 192.101.23.72. You wish to refer to this printer as **/dev/dtc72b3p2**. The entry in the **dp** file would be:

```
192.101.23.72  03/02 /dev/telnet/dtc72b3p2 /etc/ddfa/pcf
```

Note: For a DTC 16MX,TN or RX, the board number will always be '1' or '01'.

Configuration Example 2 - Let's configure a printer port on a non-DTC terminal server, whose node name is server.d.o.com and whose default port TCP address is 23. You wish to refer to this printer as **/dev/telnet/tserver1p2**. The entry in the **dp** file would be:

```
server.d.o.com  XX/XX  /dev/telnet/tserver1p2  /etc/ddfa/pcf
```

To configure a DTC printer with the HP-UX spooler, follow the steps listed above, then execute the **dpp** command. Proceed with the configuration steps used for a normal printer (MUX-connected printer), but instead of using a standard device file, you substitute the name of the device file that you defined in the **/etc/ddfa/dp** file (e.g., /dev/telnet/dtc72b3p2).

You may also use SAM to add the printer, using the standard system model scripts. Refer to Chapter 5, in the section "Setting Up Printers with the HP-UX Spooler," for a specific example. Also refer to the *HP-UX System Administration Tasks Manual* for additional information on the HP-UX printer spooler.

Configuring Incoming Connections (Terminals)

For each DTC input device (such as a terminal), perform the following steps to assign a dedicated pty file. These steps enable the Telnet Port Identification feature explained in Chapter 2. This is supported for DTCs only.

1. Determine the DTC's Node Name, its IP address, the board number and the port number on the DTC to which the terminal is connected.
2. Define a device file name that you will use for this input device. It is helpful to select a name which describes the specific device port. For example, use the pty file name**/dev/telnet/dtc16b1p2** for a device on a DTC 16TN terminal server, board 01, port 2.
3. Run the text editor of your choice (for example, **vi**) to edit the **/etc/ddfa/dp** file.
4. Add an entry (for each input device) to the **dp** file. For a DTC, use the following format:

   ```
   <DTC IP addr> <board>/<port> <pty>
   ```

   The slash (/) must separate the board and port parameters. For example, a terminal is on board 3 port 4 of a DTC 72MX whose IP address is 192.101.23.72. You wish to refer to this terminal as **/dev/telnet/dtc72b3p4**. The entry in the **dp** file would be:

   ```
   192.101.23.72  03/04  /dev/telnet/dtc72b3p4
   ```

   Note that no pcf reference is necessary for input devices such as a terminal.
5. Repeat steps 1-4 for each incoming connection device until all entries are configured.
6. Save the modified file **/etc/ddfa/dp** file. It is now ready for parsing by **dpp**.

Starting Up the ocd Daemons

1. After you have finished editing the **/etc/ddfa/dp** file, execute the Dedicated Port Parser (**/usr/sbin/dpp**) to scan the **/etc/ddfa/dp** file. This starts up the **ocd** daemon and assigns the dedicated ports for Telnet Port Identification.

   ```
   /usr/sbin/dpp  /etc/ddfa/dp  -k
   ```

2. Check to see that the ocd processes are running by using the **ps** command as follows:

   ```
   ps -ef | grep ocd
   ```

There should be one **ocd** process running for each outgoing dedicated port configured. Incoming dedicated ports do not use **ocd** processes.

Note

You should always run **dpp** every time you have added or removed any entries in the **dp** file.

For more DDFA configuration information, you may read on to Chapter 4, Defining and Executing DDFA Parameters. Refer also to the DDFA on-line manual reference pages (man pages), which may be printed from your system.

4

Defining and Executing DDFA Parameters

This chapter explains the configuration and execution parameters for the **dp** (dedicated port file), **pcf** (port configuration file), and **dpp** (dedicated port parser) in more detail.

The **dp** file defines the association of physical ports on a terminal server to logical HP-UX pty device files. A server port associated in this manner is referred to as a **dedicated port**, because of its fixed correspondence to a specific device file.

Using Dedicated Ports

The HP-UX system administrator defines the one-to-one mapping of DTC or non-DTC server devices with pseudonyms in this dedicated port file. Later, application configurators or system programmers will use these pty names to refer to these server devices.

The Dedicated Port File, dp

Dedicated ports are configured in the dedicated port file, **/etc/ddfa/dp**. The **dp** file is an ASCII file which consists of entries, one for each dedicated port, using one of the formats below.

For outgoing connections (ports for output devices such as printers and plotters):

```
<DTC IP Addr>       <board>/<port>       <pseudonym> <pcf>
<DTC Nodename>      <board>/<port>       <pseudonym> <pcf>
<Server IP Addr>    XX/XX                <pseudonym> <pcf>
<Server Name>       XX/XX                <pseudonym> <pcf>
<Server IP Addr>    XX/<port TCP addr>   <pseudonym> <pcf>
```

For incoming connections (ports for input devices such as terminals):

```
<DTC IP Addr>   <board>/<port>  <pseudonym>
<DTC Nodename>  <board>/<port>  <pseudonym>
```

The fields are defined as:

`DTC IP Addr` The IP address assigned to the DTC, or to one of its ports. The IP address is configured on the server and its ports by using the DTC Manager.

`DTC Node Name` The Node Name assigned to the DTC or one of its ports. The Node Name is configured on the server and its ports by using the DTC Manager.

`board` The physical DTC board of the port to be dedicated. Refer to the DTC hardware documentation for number of boards per each DTC type.

If `XX/XX` is specified for `<board>/<port>`, then the specified IP address or nodename references a port on the server and not the server itself.

`/` A slash must separate the board and port numbers.

`port` The port number of the DTC port to be dedicated. Refer to the DTC hardware documentation for number of ports per board for each DTC type.

`port TCP addr` The port TCP address of the server port to be dedicated. Refer to the server manufacturer's documentation for the type of addressing format used.

`pseudonym` The name of a pseudonym that will be used to access the dedicated server port. The device file name can be from 1 to 14 characters. The name of this file is created by the system administrator, and must not be the name of a file which already exists, or is already in use by the system. This file is created and managed solely by the DDFA Utilities and/or the Telnet daemon.

HP recommends using a name which helps identify the DTC port being referenced, such as **/dev/telnet/dtc1b2p4** for DTC1, board 2, port 4.

`pcf` Used for outgoing connections only. The **/etc/ddfa/pcf** file contains the port configuration which is used for the Telnet connection to the DTC or non-DTC port.

Refer to Chapter 5 for examples of how dedicated port (**dp**) files are used by DDFA.

The Port Configuration File, pcf

The port configuration file (**/etc/ddfa/pcf**) contains timer, connection, and data information for the related output device. The outgoing connection daemon, **ocd**, uses this information to manage outgoing Telnet connections from the HP-UX device file to the server port. The **pcf** file is required only for outgoing connections.

Each entry in the **dp** file which defines an outgoing dedicated port must refer to a port configuration file. In most cases, it is appropriate to use the default port configuration file, **/etc/ddfa/pcf**. You can use the same port configuration file for more than one dedicated port. Therefore, most entries in the dedicated port file will contain **/etc/ddfa/pcf** as the last field.

Whenever a **pcf** parameter or entry is changed, the **ocd** process created by that entry must be restarted in order to activate those changed parameters.

Each entry in the **pcf** file is a field-value pair on a separate line. Each field-value pair is separated by a colon (:) or one or more spaces.

The master **pcf** file is in **/usr/examples/ddfa/pcf**. The master file should always be copied to another directory, such as **/etc/ddfa/pcf**. The copy should be modified and referenced in the **dp** file. HP recommends that you create the **/etc/ddfa** directory to contain the **pcf** and **dp** files.

The **pcf** file contains the following entries and defaults:

`telnet_mode: enable`
Performs data transfer using the Telnet protocol. This option *must* be enabled when used with the DTC.

`timing_mark: enable`
Uses the Telnet timing mark negotiation at the end of the data transfer. If this is enabled, then all data is output from the server buffers to the device before the buffers are flushed.

`telnet_timer: 120`
Sets the timeout for the timing mark and binary negotiation to 120 seconds. If the negotiation does not complete within 120 seconds, an error message is logged to **/var/adm/syslog/syslog.log**,

and the error is sent to the user application program. Its range is 1 to MAXINT.

`binary_mode: disable` Transfers data in ASCII mode when this is disabled. Do not ignore processing of special characters such as XON/XOFF. If set to "enable," the data transfer over the network will be in binary mode and treatment of special characters (for example XON and XOFF) will not occur.

Because there is no flow control, data integrity cannot be guaranteed when binary mode is enabled.

If binary mode is disabled, it may still be negotiated by the application program setting IXON to 0 (zero) in the TERMIO data structure.

`open_tries: 1500` Sets the number of connection retries to 1500. If the retry process fails to make a connection, an error message is logged to **/var/adm/syslog/syslog.log**. The error message is also transmitted to the user application program.
The retry process can be interrupted by sending the SIGUSR2 signal to the **ocd** process using the **kill -17**

command. Its range is 1 to MAXINT. Zero causes exponential increments for times between retries, such as 1, 2, 4, 8, 16 and so on.

`open_timer: 30`

Sets the time between retries for making a connection to 30 seconds. Its range is 1 to MAXINT. Zero causes infinite retries.

`close_timer: 5`

Sets the time between an application program's **close** call and the actual close of the connection to zero (0) seconds. The connection will be closed and opened after every file is sent to the device. Setting the timer to a value other than zero avoids the overhead of opening and closing the connection when a spooler spools several files at a time. The connection closes after the specified length of time. Setting the timer to a high value effectively leaves the connection permanently open. Its range is 1 to MAXINT.

`status_request: disable`

Disables the sending of a status request to a device. If enabled, a status request is sent to the device. The device replies with a status such as busy or ready.

`status_timer: 30`

Sets the timeout for receipt of a status reply to 30 seconds.

If no status reply is received in that time period, an error message is logged to **/var/adm/syslog.syslog.log**. An error is also sent to the user application program. Its range is 1 to MAXINT. Zero causes MAXINT retries.

`eightbit: disable` Causes the eighth data bit (bit 7) to be stripped by the pty. If enabled, the eighth data bit is not stripped.

`tcp_nodelay: enable` Causes data to be sent to the LAN as soon as it is received by TCP. If disabled, data will be sent using normal TCP timing.

Managing Outgoing Dedicated Ports With dpp and ocd

Each outgoing dedicated port is managed by an outgoing connection daemon, **ocd**.

Once all outgoing dedicated ports have been defined in the dedicated port file, **/etc/ddfa/dp**, the DDFA Utilities can be started with the **dpp** command (see next paragraph). The result of this step will be that one outgoing connection daemon (**ocd**) process will be running for each valid outgoing entry defined in the dedicated port file. The dedicated port parser, **dpp**, scans the dedicated port file and spawns an **ocd** process for each outgoing entry in the file.

The **dpp** program can be run manually, or to start it up automatically at system bootup, set the system variable **DDFA** (to 1) in the **/etc/rc.config.d/netdaemons** file.

The syntax for **dpp** is:

```
dpp <dp_file> [-l <logfile>] [-c] [-k] [-p<ocd>]
```

`dp_file` Name of the dedicated port configuration file to use. Typically, the **dp** file is **/etc/ddfa/dp**.

`-l logfile` Name of the log file to which **dpp** should log errors. If not specified, the error messages are logged to standard error (stderr) screen.

`-c` Parses the dp file and logs all bad entries without executing **dpp** fully. It is useful to debug the **dp** file before starting the **ocds**. The **-p** option is ignored if **-c** is used.

`-k` Removes each device file from the **/dev/telnet** directory which is also in the **dp** file. Then, **dpp** spawns an **ocd** daemon for each valid entry in the **dp** file. Refer also to "Using dpp" later in this chapter.

The **ocd** daemon normally creates and removes the pty device files associated with server ports. However, if the **ocd** process is killed improperly, a device file may remain. If the system is rebooted, the **-k** option may be specified to delete any remaining device files and to restart all the **dp** file entries correctly.

`-p <ocd>` The default path for **ocd** is **/usr/sbin/ocd**. If the path is different, it must be specified with the **-p** option. The **ocd** must have execute permission set.

Full path names must be specified for the **dp** file and log files.

The **dpp** command performs the following actions for *each* outgoing dedicated port defined in the **dp** file:

- Checks the validity of the device file name.
- Runs an **ocd** process on the pty which was created. The **ocd** process manages the outgoing Telnet connection to the server port, establishing it when the pty file is opened.

  In addition, **ocd** maps the predefined pseudonyms to the ptys in the pty pool (in **/dev/pty** and **/dev/ptym**). Therefore, it is important to ensure that the number of ptys defined in the system pool in the **uxgen** file is sufficient. Consult the **insf** command man page for more information on creating ptys for the system pool.

After running **dpp**, you can check to see that the **ocd** processes are running by using the HP-UX **ps -ef** command from the Shell prompt as follows:

```
ps -ef | grep ocd
```

After the **ocd** processes have been started, accessing the pseudonym (opening the slave side of the pty) results in access to the associated device.

Preparing to Use dpp

Dpp cannot execute an **ocd** process for an entry in the **dp** file if the pty device file specified in that entry already exists. The existence of such a file could indicate that an **ocd** process is already running for that entry, or that another application is already using the device file. Therefore, it is important to kill all running **ocd** daemons properly and remove the device files they use before using **dpp** to start or restart the **ocd** daemons.

Stopping the DDFA Utilities involves two steps.

- First, the **ocd** processes must be killed.
- Second, the device files that the **ocd** processes use must be deleted from the file system.

Sending a signal 15 to an **ocd** process causes both of these things to happen automatically. For example,

```
kill -15 <ocd pid>
```

The pid is the process identification number of the **ocd** daemon. It can be found by executing **ps -ef**.

The **kill -9** command also kills the **ocd** process, but does not remove the device file. If the **kill -9** command is used, you will have to remove the device file manually from the **/dev/telnet** directory using the HP-UX **rm** command. Therefore, the **kill -15** command is more complete.

Using dpp to Kill and Restart ocds

Running the **dpp** program with the **-k** option causes **dpp** to kill all **ocd** processes for ports defined in the **dp** file and remove their associated pty device files. In addition, it restarts **ocd** processes.

Using **dpp -k** at system startup time is an excellent way to start up the **ocd** processes. Then all **ocd** processes defined in the **dp** file are started properly and old files removed even if the **ocd** processes were previously aborted. The syntax is:

```
   /usr/sbin/dpp   /etc/ddfa/dp -k
```

Using dpp to Start Up New ocds

After adding entries for outgoing dedicated ports to the **dp** file, run **dpp** without the **-k** option. The **-k** option causes **ocd** processes for existing entries to be killed. Aborting the existing **ocd** processes disrupts service to any current active sessions. To start **ocd** processes for the newly added entries to the **dp** file, execute **dpp** *without* the **-k** option. The syntax is:

```
   /usr/sbin/dpp   /etc/ddfa/dp
```

Ocd processes will be started only for the newly added entries. **Dpp** will ignore entries that already exist.

Using dpp to Remove Existing ocds

If you remove outgoing entries from the **dp** file by running **dpp**, be sure to kill their associated ocd processes using the **kill -15** command. This command will ensure that the **ocd** processes and their associated device files are removed properly. The syntax is:

```
kill -15 <ocd pid>
```

The `<ocd pid>` is the process identification number of the **ocd** daemon whose entry was removed from the **dp** file.

Managing Incoming Connections by telnetd

Incoming connections are initiated by the DTC (and not by the system) and are handled by the Telnet daemon (**telnetd**). The Telnet daemon uses the Port Identification feature to assign a pseudonym for incoming DTC connections based on entries in the **dp** file.

If an incoming dedicated port is defined in the **dp** file, **telnetd** always uses the pseudonym specified there. If the pseudonym defined there is already in use, **telnetd** refuses the connection.

If an incoming Telnet connection is not from a DTC, or if the DTC Port is not defined in the **dp** file, then **telnetd** assigns a pty to the connection in the traditional manner, which is randomly from a pool of ptys.

Using dpp to Update Telnet Port Identification Info

The **dpp** process creates a binary file which encodes the information for the incoming dedicated port mappings defined in the **dp** file. Therefore, it is important to run **dpp** after making changes to the **dp** file. The syntax is:

```
/usr/sbin/dpp   /etc/ddfa/dp
```

Incoming connections on dedicated ports do not rely on the use of **ocd**. However, **telnetd** *does* require **dpp**, because **dpp** creates the binary lookup file that **telnetd** uses.

5

Application Examples

This section provides the following configuration examples for common uses of the DDFA Utilities and Telnet Port Identification.

- Setting up printers with the HP-UX Spooler
- Accessing DTC or non-DTC ports programmatically
- Using DTC ports with CSLIP or SLIP incoming modem connections

Setting Up Printers with the HP-UX Spooler

One very common use of DDFA Utilities is sending HP-UX spooler output to printers attached to DTCs or other terminal servers. Refer to the *HP-UX System Administration Tasks Manual* for printer spooler information. Recall that when the HP-UX spooler is used with MUX-based printers, the printer is identified by its device file, such as **/dev/lp**. Each printer also requires the configuration of a model script, which defines the manner in which the spooler communicates with a specific printer type. The system administrator configures this information using SAM, or using the following HP-UX commands from the Shell prompt: **lpadmin**, **lpsched**, **lpstat**, **accept**, **enable**, and **disable**.

The HP-UX spooler can be configured to use DTC or other server-based printers in exactly the same way as MUX-based printers are. The following tasks must be completed *before* configuring the HP-UX spooler:

1. The dedicated port file (**/etc/ddfa/dp**) must contain a dedicated port entry for the server printer.
2. An **ocd** process must be running for that port.

When configuring the HP-UX spooler, the pseudonym defined in the dedicated port file (**/etc/ddfa/dp**) identifies the printer. As usual, the system administrator selects a standard system model script that matches the printer connected to the DTC or other terminal server.

HP-UX Spooler Example

You wish to configure the HP-UX spooler so that the default system printer is a DTC printer. Afterwards, test the configuration by printing a file to the DTC printer using the HP-UX **lp** command. Use either SAM (**System Administration Management**) tool *or* the HP-UX **lpadmin** command to configure the HP-UX spooler. Use the *HP-UX System Administration Tasks Manual* to find the specific menu paths.

- Steps to configure the HP-UX spooler from SAM:
  1. After entering SAM, go to the screen for local printer/plotter.
  2. Select the option for adding a printer/plotter requiring a nonstandard device file.
  3. SAM scans the system hardware and lists all the MUX cards on the system. Select any MUX card and any port on that MUX card. You will later change the MUX device file name to another device file name associated with the DTC or non-DTC server port.
  4. Complete the screen for adding a printer as follows:

     Printer name: Assign a name to the printer.

     Printer model / interface: With this field selected, press the Enter key. Select the appropriate printer from the list displayed.

Printer device file name: Enter the printer device file name that you entered in the file **/etc/ddfa/dp** for the DTC or other terminal server port. This replaces the MUX device file name that SAM supplied.

5. Complete the other fields as desired.
6. Exit from SAM.

- Another way to configure the printer is to use the command line method with the following commands:

```
lpshut
lpadmin  -pdtcprinter  -v/dev/dtc1b1p2   -mhp2235a
accept dtcprinter
enable dtcprinter
lpsched
lpstat -t
```

If desired, you can make **dtcprinter** be the system default printer:

```
lpadmin  -ddtcprinter
```

Now use the **lp** command to verify that your printer configuration worked:

```
lp /etc/ddfa/dp
```

Accessing DTC or non-DTC Ports Programmatically

Programs that communicate with devices on a MUX use the tty names defined in the **/dev** directory. Programmatic access to DTC or non-DTC server ports relies upon using outgoing dedicated ports. which are defined in the **dp** file, and have an **ocd** process running for that port. Then the standard HP-UX calls **read()**, **write()**, **open()**, **close()**, and **ioctl()** can be used.

When using the **open()** call, the name of the device file for the path parameter should be a pseudonym defined in the **dp** file. When the **open()** executes, the outbound connection daemon, **ocd**, initiates a connection to the server port specified in the **dp** file. Then, read and write calls to the device can begin.

Here is a sample application for outgoing connections:

```
#include <errno.h>
#include <stdio.h>
#include <sys/fcntl.h>
#include <sys/ptyio.h>
#include <sys/ioctl.h>
#include <sys/termio.h>

main()
{
char msg [ 80] ;
char i;
int fd;
if ( ( fd = open (  "/dev/telnet/dtcb1p4", O_RDWR  ) ) == -1 ) {
   printf (  "Unable to open device file: error %d:  %s\n",
                    errno, strerror ( errno ) );
   exit ( 1 );
}
for ( i = 0; i <= ’z’ - ’a’; i++ ) {
    sprintf ( msg, "Character %1d is %c\n", i + 1, ’a’ + i );
    write ( fd, msg, strlen ( msg ) );
}
if ( ioctl ( fd, TCSBRK, 0 ) == -1 ) {
    printf ( "Unable to send break to device: error %d: %s\n",
                    errno, strerror ( errno ) );
    exit ( 1 );
}
if ( close ( fd ) == -1 ) {
    printf ( "Unable to close device file\n" );
    exit ( 1 );
}
exit ( 0 );
}
```

Use of the **ioctl()** call and the TERMIO structure is limited to actions which are supported by the terminal server hardware and which are allowed over a networked connection.

Note ☝

The main TERMIO restrictions include modem signal control and parity checking.

Refer to the **ddfa** manual reference page for more details on **ioctl** and TERMIO limitations. Refer to the **termio** manual reference page for details on terminal input/output, ioctl, and TERMIO structures.

Configuring Incoming Connections for CSLIP/SLIP

Compressed Serial Line Internet Protocol (CSLIP) and Serial Line Internet Protocol (SLIP) are serial protocols which are used to facilitate TCP/IP networking over an asynchronous line. A common use is for a local system to dial in over modems to a DTC, which completes the connection to a remote HP 9000 system on the network.

Here are the steps required to set up an HP 9000 dial-in connection using the DTC, CSLIP or SLIP and Telnet Port Identification. Refer to the appropriate product documentation for how to configure specific parts of the connection.

1. Set up the DTC port using the OpenView DTC Manager. (The connection must have binary mode enabled on the DTC modem port. This can only be done on a DTC that is managed by a PC-based DTC Manager.)
2. Assign a dedicated pty for the incoming connection by configuring the **dp** entry in the **/etc/ddfa/dp** file. Run **dpp** on the **dp** file to activate the changes. This step must be done on the remote host.
3. Fill in the entries in the CSLIP or SLIP configuration files (**ppl.remotes**) on both the local and remote hosts. In the **ppl.remotes** file on the remote host, for the serial line parameter, enter the name of the dedicated

device file, as defined in the previous step. Remember that this device entry need not contain a **pcf**.

Use the *Using Serial Line IP Protocols* manual for examples of configuring local and remote **ppl.remotes** files.

4. Configure the HAYES-compatible modem attached to the DTC according to manufacturer's instructions. The maximum dial-in speed supported by HP for CSLIP or SLIP over DTC connections is 9600 baud. The modem should also be configured for flow control off.
5. Run **ppl** on the local host on the IP address of the remote host, as described in the **SLIP** manual.

   ```
   ppl  -o <SLIP IP addr>
   ```

CSLIP/SLIP dial-in connections over DTCs are supported in binary mode, at speeds of up to 9600 baud for HP 9000 Series 700 and 800 hosts running either HP-UX 9.x or 10.0. They are only supported in DIAL-IN mode, since DTC ports cannot be used to dial out. See the **SLIP** manual for other applications, such as dialing in and out over HP-MUXes.

6

Simple DDFA Troubleshooting

DDFA connection problems are often due to improper use and configuration of DDFA, application software, or faults in the device or terminal server connection. This manual focuses on simple tasks the user can perform to check for simple configuration faults and to test the DDFA to DTC terminal server connection.

A rule of thumb in simple troubleshooting is to divide and conquer. Identify the problem and break it into a set of smaller, easier to test components. For each symptom, try to isolate possible causes and find ways to eliminate them. For example, if an error message is displayed in the log file, find its description in Appendix B and perform the suggested action.

Troubleshooting Checklist

- Is the server device operational? For a printer, check to be sure that it is powered on, is not out of paper, and is on-line.
- Can the DTC or other terminal server port be accessed through the LAN? Ping the IP address of the server. If successful, it means that the network data path is working correctly.
- Are there any configuration mismatches between the **dp** file on the host and the DTC or other server? If so, correct them and try again.

  In the **dp** file, the following parameters should match the DTC or non-DTC server port settings:

  DTC or server IP address, DTC or server node name, board/port numbers (for DTCs only), and the port's TCP address.
- Are the device port characteristics like direct connect or modem, flow control, baud rate and parity correct? If not, then correct them, since they do impact the connection, especially one with modems.
- If the problem continues, go to the following sections, based on whether it is an incoming or an outgoing connection.
- If the problem still cannot be resolved, collect the diagnostic information listed at the end of this chapter, and call your Hewlett Packard support engineer.

Troubleshooting Outgoing Connections

Outgoing connections use **ocds** to send output to serial devices such as printers and plotters. Use this list of tasks to isolate problems due to syntax errors, incorrect port addresses, telnet operation, ocd processes, pty device files, or the system data path.

Check for Syntax Errors

Because **dpp** parses the **dp** file and tries to execute an **ocd** process for each valid entry, problems associated with **dpp** usually involve incorrect command line syntax, or references to illegal or non-existent files on the **dpp** command line. This class of problem will usually affect *all* ports configured in the **dp** file. The **dpp** error messages are listed in Appendix A of this manual.

Check for Incorrect Addresses

DDFA supports several different port addressing formats (See Chapter 3). If a ping to the IP address of the DTC or other terminal server fails, then you may have an IP configuration problem.

Verify the correct IP address of the DTC or other terminal server by cross-checking the **dp** file, and the server configuration.

Verify the TCP port associated with a DTC port to which you are trying to connect using the formula.

```
TCP_port = ( 32 * board + port + 1 ) * 256 +23
```

Check Telnet Operation

Verify that you can use telnet to access the port:

```
telnet <DTC or Server IP Addr> <TCP port>
```

If this command fails, then the port is probably in use, is not set up correctly, or may be broken. If the telnet connection succeeds, then **ocd** should also succeed.

Verify ocd

Check to see if the ocd process is running.

```
ps -ef | grep ocd
```

Look at the ocd command line to make sure you have the correct syntax and values for IP address of the DTC or non-DTC server. (Note: Board and port apply to DTCs only.)

Check the pty

Make sure the pty has a character special device file with access rights of rw-rw-rw and a major number of 17. The minor numbers will always be different.

```
ll   /dev/telnet/<devfilename>
```

The resulting display should look something like this:

```
crw-rw-rw-  1  root  sys  17  0x000030 Jan  20  10:15  /dev/telnet/dtc1b0p0
```

Check System Level Data Path

Verify that system level processes are working properly.

```
cat /etc/services  >  /dev/telnet/<devfilename>
```

At the system level, **ocd**, **cat** and all other applications use the same outbound data path. If this data is sent to the device correctly, it proves that it is possible to detect the opened device file, open a connection to the DTC or non-DTC server, and send the data to the desired port. If this fails, then it indicates the need to troubleshoot **ocd**. Refer to the **ocdebug** man page for details.

Troubleshooting Incoming Connections

Incoming dedicated connections consist of terminals or PCs directly connected or dialed into the DTC server. They use the Telnet Port Identification feature, and are supported only for connections originating from DTC-attached devices to HP 9000 systems.

One common problem occurs whenever the user attempts to log into a system and is assigned an unexpected device file. Use the following list of tasks to verify that the user's port is correctly configured in the **dp** file.

- Check that the entry in the **dp** file contains the proper configuration. Remember that entries for incoming connections do not require the specification of a port configuration file (**pcf**).
- Make sure to execute the **dpp** utility after every update of the **dp** file. **Dpp** creates a binary file which is used by **telnetd** to assign the proper device files to incoming connections.
- Check the version of the DTC Manager which manages DTC server ports. If the DTC Manager is PC-based, you should have version A.14.00 or later. If it is the host-based DTC Manager, you should have version A.03.00 or later.
- Check the version of **telnetd** on the HP 9000 using the HP-UX **what** command.

  ```
  what /usr/lbin/telnetd
  ```

  On the $Header line display by the **what** command, the telnetd version should be 1.22.110.10 or later.

  ```
  $Header: telnetd.c, v 1.22.110.10
  ```

- Make sure that a Telnet connection can succeed.

  ```
  telnet <host IP addr>
  ```

Remember that the assignment of dedicated ptys on the HP 9000 depends upon the successful exchange of DTC port information by the DTC Telnet subsystem and the Telnet daemon on the HP 9000 at connection establishment time. Only the correct versions of software are capable of exchanging this information.

If You Have to Call Hewlett-Packard

If your troubleshooting indicates that a serious problem has arisen and you need to call your Hewlett-Packard support engineer, please ask your system administrator to help you collect as many of the following items as possible. This list of items will greatly assist the support engineer in isolating and resolving DDFA problems.

- A description of the problem and the conditions under which it occurred. For example, did it occur on a previous release of DDFA or HP-UX, can it be replicated, does the same problem occur with a MUX connection?
- The following data taken while the problem is occurring:
  1. a log file of an ocdebug run (**/var/adm/ocd<pid>**)
  2. the syslog.log file from **/var/adm/syslog**
  3. a nettl trace (see man pages for how to use nettl)
  4. printout of spooler information by typing **lpstat -t**
  5. printout of process information by typing **ps -ef**
  6. printout of network status information from **netstat -n**
- The complete DDFA data path configuration:
  1. printout of the **/etc/lp/interface** file associated with the offending printer
  2. listing of the **dp** files and **pcf** files used
  3. version number of HP-UX by typing **uname -a**

4. hardware platform, such as HP 9000 Series 800 or 700, and the model number
5. version of DDFA used by typing **what**
6. list of installed DDFA patches from typing **what**
7. DTC Manager version. Refer to the appropriate DTC Manager manual for details.
8. map of your DTC or non-DTC configuration down to the port level
9. type of device to which you wish to send outgoing or transmit incoming data
10. map of your network between the system and the DTC or non-DTC terminal server
11. Non-DTC server information, including manufacturer, hardware and software versions and server configuration settings. Refer to the manufacturer's product documentation.

A

Appendix A: dpp Error Messages

Error Number	Message	Reason
0	dp_file is mandatory	No **dp** file was found when **dpp** was started.
1	dp file must be the first argument	The **dp** file must be specified as the first argument when **dpp** is started.
2	Cannot read dp file <dp_file>	The **dp** file mentioned does not exist or cannot be accessed with current access rights.
3	No log file defined (-l option)	The **-l** option has been detected with no argument.
4	Cannot create log file (-l <log_file>)	The log file cannot be created because of invalid path or insufficient access rights.
5	Cannot access log file (-l <log_file>)	The log file cannot be accessed because of invalid path or insufficient access rights.
6	No **ocd** file defined in program option	The option **-p** has been detected with no arguments.

Error Number	Message	Reason
7	Cannot execute **ocd** program (-p <program>	The **ocd** program does not exist or is not an executable file with correct access rights.
8	Cannot purge device file (/dev/*xxx*)	The option **-k** has been selected and the device file exists but cannot be purged because of insufficient access rights.
9	Cannot execute default program (/usr/sbin/ocd)	The **ocd** program cannot be executed because of insufficient access rights or because it has not been properly installed.
10	Entry ignored (Bad IP address)	The mentioned entry of the **dp** file does not have a valid IP address.
11	Entry ignored (No port/board info)	The entry is ignored because **board/port** information was not mentioned in the **dp** file.
12	Entry ignored (Bad port number)	The port mentioned in the displayed entry is not either a decimal value, or a string composed of **x** or **X** characters.
13	Entry ignored (Bad board number)	The board mentioned in the displayed entry is not a decimal value, or it is not a string composed of **x** or **X** characters.
14	No more processes available on system	The **ocd** program cannot be started because no more processes can be started on the system.

Error Number	Message	Reason
15	Entry ignored (No device name)	The device file name has not been configured in the **dp** file.
16	Entry ignored (Bad device name)	The displayed device file cannot be created because of invalid path or insufficient access rights.
17	Entry ignored (Bad config name)	The displayed configuration file cannot be read because of invalid path or insufficient access rights.
18	Entry ignored (Invalid log level)	The logging level was not set because of invalid values (Must be 0-3).
19	Entry ignored (Bad node name)	The displayed configuration file cannot be read because of invalid node name.
30	Too many login entries (Limited to 2000)	More than 2000 dedicated ports for Telnet Port Identification were specified.
31	Cannot access dpp login file (/var/adm/dpp_login.bin)	Port Identification feature is disabled due to a dpp login file with insufficient access rights.
50	Device file not specified in /dev/telnet	Many commands (such as ps) may not display the correct device file in their output unless device file is located in /dev/telnet.
99	Unknown error	Error of unknown origin.

B

Appendix B: ocd Logging Messages

The outbound connection daemon (**ocd** or **ocdebug**) sends its system logging messages to the file **syslog.log** in the directory **/var/adm/syslog**. Refer to the manual pages for how to use **ocd** and **ocdebug** logging for basic troubleshooting.

The **ocd** logging messages are classified into log levels 0 through 3 (critical, serious, warning or informational). Each error message is described by error number and lists a cause and suggested action.

Critical Messages

Critical logging messages are issued when quitting ocd or ocdebug is caused by the use of invalid options, argument values, or pseudonyms, or the reception of SIGINT (signal interrupt).

Usage logging messages are caused by syntax errors.

```
usage: ocd -n<node_name> -f<pseudonym> [-b<board_no>] [-p<port_no>] [-c<pc_file_path>]

                      [-l<log_level>]

usage: ocdebug -n<node_name> -f<pseudonym> [-b<board_no>] [-p<port_no>] [-c<pc_file_path>]

                      [-l<log_level>]  [-d<level>]

CAUSE: Ocd or ocdebug was executed with an illegal or missing option.

ACTION: Restart ocd or ocdebug using the correct syntax and note that:

1. Legal options for both ocd and ocdebug are -n, -f, -b, p, -c, and -l.

2. Option -d is a legal option only for ocdebug.

3. Options -n and -f are mandatory for both ocd and ocdebug.

4. Each option requires an argument.


(100) ERROR: Invalid node name <node_name>: system error <system_error>: <system_message>

CAUSE: The node name argument specified with the -n option could not be
       resolved into an IP address due to the system error given.

ACTION: Restart ocd or ocdebug using a valid node name and note that:

1. The node name specified with the -n option must be defined in a name database.

2. The node name specified with the -n option must have an IP address assigned to it in that
     name database.

3. The access to that name database must not be impeded in any way.

One way of validating the node name is to execute the nslookup command.
```

```
(101) ERROR: Invalid board number <board_number>

CAUSE: The board number argument specified with the -b option was not in
        the range 0 to 7 for a DTC, or it was not a string consisting of "X" or "x".

ACTION: Restart ocd or ocdebug using a DTC board number in the range 0 to 7, or a string
       consisting of "X" or "x".

(102) ERROR: Invalid port number <port_number>

CAUSE: The port number argument specified with the -p option was not correct as follows:

If the board number argument contained a value 0 to 7 for a DTC,
         then the port number argument was not in the range 0 to 31.

If the board number argument was a string consisting of "X" or "x",
       then the port number argument was not an integer value representing
       a TCP service port, or a string consisting of "X" or "x" (which
       refers to the default TCP port address of 23 for the node name specified).

ACTION: Restart ocd or ocdebug using a port number that is an integer value
       of a valid DTC port number in the range 0 to 31 or a TCP service port
       number, or a port number that is a string consisting of "X" or "x".

(103) ERROR: Pseudonym has become invalid

CAUSE: The pseudonym in use ceased to exist (most likely because another
       process deleted it) or was not in use by ocd or ocdebug and possibly
       was in use by another process.

NOTE: This message is followed by message (403).

ACTION: Make sure that no other process is using the pseudonym and restart ocd or ocdebug.

(104) ERROR: Pseudonym <pseudonym_name> no longer exists

CAUSE: The pseudonym specified ceased to exist (most likely because another process deleted it).

ACTION: Restart ocd or ocdebug.
```

```
(105) ERROR: Pseudonym <pseudonym_name> unavailable

CAUSE: The pseudonym specified was in use by another process and was not
       available for use by ocd or ocdebug.

ACTION: Determine if the process using the pseudonym is necessary and
       restart ocd or ocdebug if it is not (after stopping that process and
       removing the pseudonym).

(106) ERROR: Pseudonym <pseudonym_name> in use but not by DDFA

CAUSE: The pseudonym specified was in use by or created by a process other
       than ocd or ocdebug.

ACTION: Determine if the process using the pseudonym is
        necessary and restart ocd or ocdebug if it is not
        (after stopping that process and removing the pseudonym).

(107) ERROR: Pseudonym <pseudonym_name> in use by DDFA with
          process identifier <process_identifier>

CAUSE: The pseudonym specified was in use by another ocd or ocdebug.

ACTION: Determine if the process using the pseudonym is necessary and restart ocd
       or ocdebug if it is not (after stopping that process  and removing the pseudonym).

(108) ERROR: Received SIGINT signal

(109) ERROR: Cleaning up daemon for pseudonym <pseudonym_name> and device
             at node <node_name> board <board_number> port <port_number>

CAUSE: Ocd or ocdebug received a SIGINT signal and closed its connection
        and deleted its pseudonym.

NOTE: These messages are followed by messages (403) and (432).

ACTION: Restart ocd or ocdebug.
```

```
(150) ERROR: Unable to allocate pty because no pty available for use

CAUSE: There were no ptys available for use on the system.

ACTION: Allocate more ptys on the system or wait to restart ocd or ocdebug
       until there are fewer processes running.

(151) ERROR: Unable to obtain file information for pty <slave_pty_name>:
                          system error <system_error>: <system_message>

CAUSE: There was no information available for the pty specified due to the
       system error given.

ACTION: Look up what the stat(2) man page states about the system error
        given and make the correction as appropriate.

(152) ERROR: Unable to create pseudonym <pseudonym_name> with device
       identifier <device_identifier>: system error <system_error>: <system_message>

CAUSE: The pseudonym specified was not created due to the system error given.

ACTION: Look up what the mknod(2) man page states about the system error
        given and make the correction as appropriate.

(153) ERROR: Pseudonym <pseudonym_name> does not belong to this daemon

CAUSE: The pseudonym specified was not in use by ocd or ocdebug and possibly
       was in use by another process.

ACTION: Determine if the process using the pseudonym is necessary and restart ocd or
         ocdebug if it is not (after stopping that process and removing the pseudonym).

(154) ERROR: Unable to obtain information concerning availability of
            pseudonym <pseudonym_name>: system error <system_error>: <system_message>

CAUSE: There was no information available for the pseudonym specified due to
       the system error given.

ACTION: Look up what the stat(2) man page states about the system error given and
        make the correction as appropriate.
```

```
(155) ERROR: Unable to free pseudonym <pseudonym_name>:
             system error <system_error>: <system_message>
```

CAUSE: The pseudonym specified was not freed from use by another process due to the system error given.

ACTION: Look up what the kill(2) man page states about the system error given and make the correction as appropriate.

```
(156) ERROR: Pseudonym <pseudonym_name> is not pty slave
```

CAUSE: The pseudonym specified was not a device special file.

ACTION: Delete the pseudonym and restart the ocd or ocdebug.

```
(157) ERROR: Unable to delete pseudonym <pseudonym_name> left by previously
             run process: system error <system_error>: <system_message>
```

CAUSE: The pseudonym specified was not deleted for use by a fresh ocd or ocdebug due to the system error given.

ACTION: Delete the pseudonym and restart ocd or ocdebug.

MESSAGE(S) THAT ACCOMPANY ALL CRITICAL ERROR MESSAGES:

```
(199) ERROR: Terminating daemon
```

CAUSE: Ocd or ocdebug was shutdown due to a critical error.

ACTION: Take action based on cause indicated by preceding error messages.

Serious Messages

Serious logging messages are issued after:

1. **ocd** closes a connection or pseudonym that has broken the communication path between the application and the device.
2. Unexpected conditions impact **ocd** or other processes.

```
(200) ERROR: Unable to obtain data from network: system error <system_error>: <system_message>

CAUSE: There was no data to receive from the network due to the system error given.

NOTE: This message is followed by messages (296), (297), (298), and (299).

NOTE: System error 232 - A transport reset was received from the remote terminal server.
      System error 238 - A transport connection timeout was received from the remote terminal server.

ACTION: Determine if the terminal server and device are operational and enable
        the printer or restart the application if they are.

(201) ERROR: Binary negotiation failed on opening connection

CAUSE: The binary negotiation was not completed successfully on opening a
       connection to the device on the remote terminal server.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server and device can accept binary data
        and restart the application if they can.

(202) ERROR: Invalid initial status request received

CAUSE: The initial status request to the device on the remote terminal server failed.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server and device are operational and restart
        the application if they are.
```

```
(203) ERROR: Connection failed

CAUSE: A connection to the remote terminal server was not made.

NOTE: This message is followed by messages (296), (297), and (298).

ACTION: Determine if the terminal server and device are operational and enable
        the printer or restart the application if they are.

(204) ERROR: Invalid terminating status request received

CAUSE: The final status request to the device on the remote terminal server failed.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server and device are operational and restart
        the application if they are.

(205) ERROR: Give up trying to obtain connection as number of open connection tries
             has been exhausted

CAUSE: The number of tries to open a connection to the device on the remote
       terminal server was exhausted.

NOTE: This message is followed by messages (296), (297), and (298).

ACTION: Determine if the terminal server and device are operational and enable
        the printer or restart the application if they are.

(206) ERROR: Unable to obtain timing mark negotiation from network:
             system error <system_error>: <system_message>

CAUSE: There was no timing mark negotiation to receive from the network due to
       the system error given.

NOTE: This message is followed by message (298).

ACTION: Determine if the terminal server and device are operational and enable the printer if they are.
```

```
(207) ERROR: No timing mark negotiation to obtain from network within time interval allowed

CAUSE: There was no data and thus no timing mark negotiation to receive from
       the network during the time to wait for such negotiations (most likely
       because of the load on the terminal server, network, or system).

NOTE: This message is followed by message (298).

ACTION: Determine if the terminal server and device are operational and enable
        the printer if they are.

(208) ERROR: No timing mark negotiation received in data during time interval allowed

CAUSE: There was no timing mark negotiation in the data received from the remote terminal server.

NOTE: This message is followed by message (298).

ACTION: Enable the printer.

(209) ERROR: Unable to obtain status reply from device:
             system error <system_error>: <system_message>

CAUSE: There was no status reply to receive from the network due to the system error given.

NOTE: This message is followed by messages (296), (297), (298), and (299).

ACTION: Determine if the terminal server and device are operational and enable
        the printer or restart the application if they are.

(210) ERROR: No status reply to obtain from device within time interval allowed

CAUSE: There was no data and thus no status reply to receive from the network
       during the time to wait for that status reply (most likely because of
       the load on the terminal server, network, or system).

NOTE: This message is followed by messages (296), (297), (298), and (299).

ACTION: Determine if the terminal server and device are operational and enable
       the printer or restart the application if they are.

(211) ERROR: Printer busy

CAUSE: The printer on the remote terminal server was busy with another request.

ACTION: Wait for the printer to free up before sending the next printer request.

(212) ERROR: Printer out of paper

CAUSE: The printer on the remote terminal server was out of paper.
```

ACTION: Add paper to the printer, put it on-line, and send the printer request.

(213) ERROR: Printer off-line

CAUSE: The printer on the remote terminal server was off-line.

ACTION: Put printer on-line and send the printer request.

(214) ERROR: Printer data error

CAUSE: The printer on the remote terminal server had a data error.

NOTE: This message is followed by messages (296), (297), (298), and (299).

ACTION: Determine if the printer is operational and enable printer and send the printer request if it is.

(215) ERROR: Unknown status reply <status_reply>

CAUSE: No known status reply was received in the data from the device on the remote terminal server.

ACTION: Refer to the manual for the device for an explanation of the status reply.

(216) ERROR: Printer problem not fixed within time interval allowed

CAUSE: The printer on the remote terminal server was not made ready during the time to wait for a status reply.

NOTE: This message is followed by messages (296), (297), (298), and (299).

ACTION: Put printer on-line, enable printer, and send the printer request.

```
(217) ERROR: Unable to obtain binary negotiation from network:
             system error <system_error>: <system_message>

CAUSE: There was no binary negotiation to receive from the network due to the
       system error given.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server and device are operational and restart the
        application if they are.

(218) ERROR: No binary negotiation to obtain from network within time interval allowed

CAUSE: There was no data and thus no binary negotiation to receive from the
       network during the time to wait for such negotiations (most likely
       because of the load on the terminal server, network, or system).

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server and device are operational and restart
        the application if they are.

(219) ERROR: Binary negotiation not completed on time

CAUSE: There was no binary negotiation in the data received from the remote
       terminal server.

NOTE: This message is followed by message (296), (297), and (299).

(220) ERROR: Received SIGUSR2 signal
(221) ERROR: Give up trying to obtain connection at request of user

CAUSE: Ocd or ocdebug received a SIGUSR2 signal and any more tries to open
       a connection to the device on the remote terminal server were stopped.

ACTION: Determine if the terminal server and device are operational and
        restart the application if they are.
```

```
MESSAGES COMMONLY SEEN WITH NON-STANDARD (USER-WRITTEN) APPLICATIONS:

(222) ERROR: Control device request rejected as connection not open

CAUSE: A control device request was received before an open request was
       received and after a close request was received.

NOTE: This message is followed by messages (296) and (297).

ACTION: Have all control device requests for a pseudonym sandwiched between an
        open request for that pseudonym and a close request for that pseudonym.

(223) ERROR: Unable to obtain control device request argument:
             system error <system_error>: <system_message>

CAUSE: The parameters to set for a set parameters control device request
       could not be obtained due to the system error given.

ACTION: Determine if the correct structure (termio) is used and the correct
        parameters are being set and make any changes necessary.

(224) ERROR: Unable to toggle binary mode off

CAUSE: The binary mode was not turned off successfully.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server and device are operational and whether
        binary mode can be toggled off for the terminal server and
        restart the application if all conditions are met.

(225) ERROR: Unable to toggle binary mode on

CAUSE: The binary mode was not turned on successfully.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server and device are operational and whether
         binary mode can be toggled off for the terminal server and
         restart the application if all conditions are met.
```

```
(226) ERROR: Unable to obtain send break request argument:
             system error <system_error>: <system_message>

CAUSE: The parameter for a send break control device request could not be
       obtained due to the system error given.

ACTION: Determine if the correct structure (termio) is used and the correct
        parameters are being set and make any changes necessary.

(227) ERROR: Unable to obtain flow control request argument:
             system error <system_error>: <system_message>

CAUSE: The parameter for a flow control device request could not be
       obtained due to the system error given.

ACTION: Determine if the correct structure (termio) is used and the correct
        parameters are being set and make any changes necessary.

(228) ERROR: Give up trying to obtain connection as connection already established

CAUSE: A connection to the device on the remote terminal server was already established.

ACTION: Establish only one connection for each device on a remote terminal server.

(229) ERROR: DO binary negotiation rejected by remote side

CAUSE: The idea of binary mode data transfers was rejected by the remote terminal server.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if binary mode can be toggled on for the terminal server
        and restart the application if it can.

(230) ERROR: DONT binary negotiation rejected by remote side

CAUSE: The idea of non-binary mode data transfers was rejected by the remote
       terminal server.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if binary mode can be toggled off for the terminal server
        and restart the application if it can.
```

```
(250) ERROR: Unable to obtain data from network or pty

CAUSE: There were no data or events to be passed on the network or through the
       pseudonym due to a system error.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Restart ocd or ocdebug and restart the application.

(251) ERROR: Unable to obtain data:      system error <system_error>: <system_message>

CAUSE: There were no data or events to be passed on the network or through the
       pseudonym due to the system error given.

ACTION: Look up what the select(2) man page states about the system error
        given and make the correction as appropriate.

(252) ERROR: Unable to obtain data from pty: system error <system_error>: <system_message>

CAUSE: There were no data to receive from the pty due to the system error given.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Look up what the read(2) man page states about the system error
       given and make the correction as appropriate.

(253) ERROR: Unable to obtain request information from pty:
             system error <system_error>: <system_message>

CAUSE: There was no control request data to receive from the pty due to the
       system error given.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.

(254) ERROR: Unknown Telnet state <telnet_state>

CAUSE: A Telnet state discovered while processing a Telnet command was not a
       known Telnet state.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server is using the Telnet protocol correctly
        and restart the application if it is.
```

(255) ERROR: Unknown internal parameter in port configuration file

CAUSE: The internal port configuration parameter discovered while processing the port configuration file is not a known parameter.

ACTION: No action.

(256) ERROR: Give up trying to obtain connection: system error <system_error>: <system_message>

CAUSE: Any more tries to open a connection to the device on the remote terminal server were stopped due to the system error given.

NOTE: This message is followed by messages (296), (297), and (298).

ACTION: Look up what the connect(2) man page states about the system error given and make the correction as appropriate.

(257) ERROR: Unable to obtain out-of-band data: system error <system_error>: <system_message>

CAUSE: There was no out-of-band data to receive from the network due to the system error given.

ACTION: Look up what the select(2) man page states about the system error given and make the correction as appropriate.

(258) ERROR: No timing mark negotiation to obtain from network

CAUSE: Unexpected data was received from a place other than the network before there was a timing mark negotiation to receive from the network.

NOTE: This message is followed by message (298).

ACTION: Enable the printer and send the printer request.

(259) ERROR: Unable to read timing mark negotiation from network: system error <system_error>: <system_message>

CAUSE: There was no timing mark negotiation to read from the network due to the system error given.

NOTE: This message is followed by message (298).

ACTION: Look up what the read(2) man page states about the system error given and make the correction as appropriate.

```
(260) ERROR: Unknown Telnet state <telnet_state> during timing mark negotiation

CAUSE: A Telnet state discovered while processing a Telnet timing mark
       negotiation was not a known Telnet state.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Determine if the terminal server is using the Telnet protocol correctly
        and restart the application if it is.

(261) ERROR: Unable to read status reply from device:
             system error <system_error>: <system_message>

CAUSE: There was no status reply to read from the network due to the system error given.

NOTE: This message is followed by messages (296), (297), (298), and (299).

ACTION: Look up what the read(2) man page states about the system error
        given and make the correction as appropriate.

(262) ERROR: No status reply to obtain from device

CAUSE: Unexpected data was received from a place other than the network
       before there was a status reply to receive from the network.

NOTE: This message is followed by messages (296), (297), (298), and (299).

ACTION: Restart the application, enable the printer, and send the printer request.

(263) ERROR: No binary negotiation to obtain from network

CAUSE: Unexpected data was received from a place other than the network
       before there was a binary negotiation to receive from the network.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Restart the application.

(264) ERROR: Unable to read binary negotiation from network:
             system error <system_error>: <system_message>

CAUSE: There was no binary negotiation to read from the network due to the system error given.

NOTE: This message is followed by message (296), (297), and (299).

ACTION: Look up what the read(2) man page states about the system error
        given and make the correction as appropriate.
```

```
(265) ERROR: Unable to get terminal parameters: system error <system_error>: <system_message>

CAUSE: There were no device file parameters to obtain from the pty due to the system error given.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.

(266) ERROR: Unable to set terminal parameters: system error <system_error>: <system_message>

CAUSE: The device file parameters were not set on the pty due to the system error given.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.

(267) ERROR: Unable to write data to pty: system error <system_error>: <system_message>

CAUSE: There were no data to send to the pty due to the system error given.

ACTION: Look up what the write(2) man page states about the system error
        given and make the correction as appropriate.

(268) ERROR: Unable to write data to network: system error <system_error>: <system_message>

CAUSE: There were no data to send to the network due to the system error given.

ACTION: Look up what the send(2) man page states about the system error
       given and make the correction as appropriate.

(269) ERROR: Unable to delete pseudonym <pseudonym_name> for recreation and use:
             system error <system_error>: <system_message>

CAUSE: The pseudonym was not deleted due to the system error given.

ACTION: Delete the pseudonym and restart ocd or ocdebug.

(270) ERROR: System returned error <return_error> while disabling printer:
             system error <system_error>: <system_message>

CAUSE: The disable command had an error as given.  This does not affect
       the disabling of the printer queue.

ACTION: No immediate action is necessary although enabling the Unix spooler printer queue
         in the future is required to restart the printing process for that printer queue.
```

MESSAGE(S) THAT ACCOMPANY SOME SERIOUS ERROR MESSAGES:

(296) ERROR: Recreating pseudonym <pseudonym_name> with new pty

CAUSE: The pseudonym was recreated and linked to a new pty opened for it.

NOTE: This message accompanies messages (200), (201), (202), (203), (204), (205), (209), (210), (214), (216), (217), (218), (219), (222), (224), (225), (229), (230), (250), (252), (253), (254), (256), (260), (261), (262), (263), and (264).

ACTION: No action is necessary.

(297) ERROR: Removing pseudonym <pseudonym_name> and closing pty <master_pty_name>

CAUSE: The pseudonym was deleted and its associated pty was closed due to a serious error with the pseudonym.

NOTE: This message accompanies messages (200), (201), (202), (203), (204), (205), (209), (210), (214), (216), (217), (218), (219), (222), (224),(225), (229), (230), (250), (252), (253), (254), (256), (260), (261), (262), (263), and (264).

NOTE: This message is followed by message (432).

ACTION: No action is necessary.

(298) ERROR: Printer disabled

CAUSE: The Unix spooler printer queue associated with the outbound connection daemon was disabled.

NOTE: This message accompanies messages (200), (203), (205), (206), (207), (208), (209), (210), (214), (216), (256), (258), (259), (261), and (262).

ACTION: No immediate action is necessary although enabling the Unix spooler printer queue in the future is required to restart the printing process for that printer queue.

(299) ERROR: Closing connection due to error with data path

CAUSE: The connection to the device on the remote terminal server was closed due to a serious error with the connection or with the pseudonym.

NOTE: This message accompanies messages (200), (201), (202), (204), (209),(210), (214), (216), (217), (218), (219), (224), (225), (229), (230), (250), (252), (253), (254), (260), (261), (262), (263), and (264).

ACTION: No action is necessary.

Warning Messages

Warning logging messages are issued when an unexpected condition occurs that does not affect normal operation of **ocd** or other processes.

Usage messages are caused by syntax errors.

```
USAGE MESSAGE: ocd: unknown option: <option> (ignored)

CAUSE: The option given was not a valid option.

ACTION: Remove the option from the command line.


(300) WARNING: Invalid logging level <log_level> (valid values are 0, 1, 2, or 3)

CAUSE: The logging level given was not 0, 1, 2, or 3.

ACTION: Use a logging level of 0, 1, 2, or 3.

(301) WARNING: Port configuration file not specified or not specified correctly

CAUSE: The -c option was not specified.

NOTE: This message is followed by message (350).

ACTION: Use a correct configuration file name.

(302) WARNING: Timing mark negotiation disabled because Telnet mode disabled

CAUSE: Timing mark negotiation was enabled but the Telnet mode was disabled.

ACTION: Either enable the Telnet mode or disable the timing mark negotiation.
```

```
(303) WARNING: Pseudonym <pseudonym_name> not specified in /dev/telnet

CAUSE: The pseudonym specified was not to be placed into the directory
       /dev/telnet set aside for all pseudonyms.

ACTION: Place all pseudonym names in /dev/telnet.

(304) WARNING: Unable to change permissions of pseudonym <pseudonym_name> to <permission>:
               system error <system_error>: <system_message>

CAUSE: The permissions of the pseudonym were not changed to optimal values
                 (0666) due to the system error given.

ACTION: Look up what the chmod(2) man page states about the system error
        given and make the correction as appropriate.

(305) WARNING: Unable to enable packet mode on pty:
               system error <system_error>: <system_message>

CAUSE: Packet mode on the pty could not be enabled due to the system error given.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.

(306) WARNING: Unable to enable non-blocking mode on pty:
               system error <system_error>: <system_message>

CAUSE: Non-blocking mode on the pty could not be enabled due to the system error given.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.

(307) WARNING: Application canceled open request

CAUSE: The application closed the pty before the open request had completed.

ACTION: Restart the application.

(308) WARNING: Unable to enable non-blocking mode on network socket:
               system error <system_error>: <system_message>

CAUSE: Non-blocking mode on the network could not be enabled due to the system error given.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.
```

```
(309) WARNING: Close not completed as data processing not complete

CAUSE: There was still data to send to the device on the remote terminal
       server as the close was requested.

ACTION: No action is necessary.

(310) WARNING: Close request ignored as connection not open

CAUSE: The close request was received and ignored as the connection was not open.
       This occurred because of an application sending a close request at the
       wrong time or because of an error that previously closed the connection.

ACTION: Generally, no action is necessary unless a close request is not
        sent after an open request.

(311) WARNING: Binary mode cannot be turned off as permanently turned on

CAUSE: The application requested that the binary mode be turned off when the
       binary mode is turned on for the connection using the port configuration
       file and thus cannot be turned off.

ACTION: Determine if the binary mode is necessary at all times and disable it if it is not.

(312) WARNING: Unable to perform flush operation

CAUSE: The application requested a flush operation and ocd and ocdebug do not
       support that operation.

ACTION: Determine if a flush operation is necessary by the application and
        do not perform it if it is not.

(313) WARNING: Cannot open port configuration file:
               system error <system_error>: <system_message>

CAUSE: The open of the port configuration file failed due to the system error given.

NOTE: This message is followed by message (350).

ACTION: Verify the port configuration file name, location, and permissions.
```

```
(314) WARNING: Unknown parameter <parameter_name> in port configuration file (ignored)

CAUSE: The port configuration parameter specified is not valid and has been ignored.

ACTION: Verify the port configuration parameter name.

(315) WARNING: Unable to set keep alive option on network socket:
               system error <system_error>: <system_message>

CAUSE: The keep alive option on the network could not be set due to the system error given.

ACTION: Look up what the setsockopt(2) man page states about the system error
        given and make the correction as appropriate.

(316) WARNING: Unable to set option on network socket as unable to obtain protocol attributes:
               system error <system_error>: <system_message>

CAUSE: The no delay option on the network could not be set as the protocol
       could not be obtained due to the system error given.

ACTION: Look up what the getprotobyname(3N) man page states about the system
        error given and make the correction as appropriate.

(317) WARNING: Unable to set no delay option on network socket:
               system error <system_error>: <system_message>

CAUSE: The no delay option on the network could not be set due to the system error given.

ACTION: Look up what the setsockopt(2) man page states about the system error
        given and make the correction as appropriate.

(318) WARNING: Unable to reply to control device request:
               system error <system_error>: <system_message>

CAUSE: The control device reply to the control device request could not complete due to
       the system error given.  A reply is made for each request sent by
       an application and must be made for the application to continue.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.
```

```
(319) WARNING: Unable to obtain negotiation from network:
               system error <system_error>: <system_message>

CAUSE: There was no Telnet negotiation to receive from the network due to the
       system error given.

ACTION: Look up what the select(2) man page states about the system error
        given and make the correction as appropriate.

(320) WARNING: No negotiation to obtain from network within time interval allowed

CAUSE: There was no data and thus no Telnet negotiation to receive from
       the network during the time to wait for such negotiations (most likely
       because of the load on the terminal server, network, or system).

ACTION: Determine if the terminal server and device are operational and make
        them operational if they are not.

(321) WARNING: Unable to read negotiation from network:
               system error <system_error>: <system_message>

CAUSE: There was no Telnet negotiation to read from the network due to the
       system error given.

ACTION: Look up what the read(2) man page states about the system error
        given and make the correction as appropriate.

(322) WARNING: No negotiation to obtain from network

CAUSE: Unexpected data was received from a place other than the network
       before there was a Telnet negotiation to receive from the network.

ACTION: No action is necessary.

(323) WARNING: No status reply from device

CAUSE: No status reply was received in the data from the device on the remote
       terminal server.

ACTION: No action is necessary.
```

```
(324) WARNING: Status reply received <number_of_characters> characters

CAUSE: A status reply of more than one character was received in the data
       from the device on the remote terminal server.

ACTION: No action is necessary.

(325) WARNING: Unable to send interrupt signal to application:
               system error <system_error>: <system_message>

CAUSE: An interrupt could not be sent to the application due to the system error given.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.

(326) WARNING: Unable to send break to application:
               system error <system_error>: <system_message>

CAUSE: A break could not be sent to the application due to the system error given.

ACTION: Look up what the ioctl(2) man page states about the system error
        given and make the correction as appropriate.

(327) WARNING: Unable to write <number_of_bytes> bytes to pty

CAUSE: An attempt to write all data to the pty failed because the pty had
       no room for the amount of data specified.

ACTION: Determine if the application is running correctly and draining data
        from the pty regularly and restart the application if it is not.

(328) WARNING: Unable to write <number_of_bytes> bytes of normal data to network

CAUSE: An attempt to write all data to the network failed because the network
       had no room for the amount of data specified.

ACTION: Determine if the network, terminal server, and device are running correctly
        and draining data regularly and correct the situation as necessary.
```

```
(329) WARNING: Unable to write <number_of_bytes> bytes of urgent data to network

CAUSE: An attempt to write all data to the network failed because the network
       had no room for the amount of data specified.

ACTION: Determine if the network, terminal server, and device are running correctly
        and draining data regularly and correct the situation as necessary.

(330) WARNING: Unable to obtain information concerning existence of
               pseudonym <pseudonym_name>: system error <system_error>: <system_message>

CAUSE: There was no information available for the pseudonym specified due to
       the system error given.

ACTION: Look up what the stat(2) man page states about the system error
        given and make the correction as appropriate.

(331) WARNING: Unable to delete pseudonym <pseudonym_name> when terminating daemon:
               system error <system_error>: <system_message>

CAUSE: The pseudonym was not deleted due to the system error given.

ACTION: Delete the pseudonym.

(332) WARNING: Unknown logging level

CAUSE: A log message with an unknown logging level was encountered.

ACTION: No action is necessary.

MESSAGE(S) THAT ACCOMPANY SOME WARNING MESSAGES:

(350) WARNING: Using default port configuration values

CAUSE: The hard-coded default port configuration values were invoked for use
       as the port configuration file could not be used due to the warning
       given in the accompanying message (either message (301) or message (313)).

ACTION: No action is necessary.
```

Informational Messages

Informational messages are issued when an expected condition occurs during normal processing. No action is required.

```
(400) NOTE: Target device at node <node_name> board <board_number>
            port <port_number> TCP port <tcp_port_number>

CAUSE: Indicates the location of the device on the remote terminal server.

(401) NOTE: Created pseudonym <pseudonym_name>

CAUSE: Indicates the name of the pseudonym created for use.

(402) NOTE: Linked pseudonym <pseudonym_name> to master <master_pty_name>
            slave <slave_pty_name> file descriptor <file_descriptor>

CAUSE: Indicates the association of the pseudonym with a pty.

(403) NOTE: Closing connection

CAUSE: Indicates the closing of a network connection to the remote terminal
       server due to the serious error given in the accompanying message
       (either message (103) or message (108)).

(404) NOTE: Received open request from application <process_identifier>

CAUSE: Indicates the reception of an open request for a pseudonym by the application specified.

(405) NOTE: Set pty characteristics

CAUSE: Indicates the setting of the initial pty characteristics.
(406) NOTE: Sending initial status request to device

CAUSE: Indicates the sending of a status request to the device on the
       remote terminal server before any data is sent to that device.
```

```
(407) NOTE: Received close request from application <process_identifier>

CAUSE: Indicates the reception of a close request for a pseudonym by the application specified.

(408) NOTE: Received control device request from application <process_identifier>

CAUSE: Indicates the reception of a control device request for a pseudonym
       by the application specified.

(409) NOTE: Sending terminating status request to device

CAUSE: Indicates the sending of a status request to the device on the
       remote terminal server after all data is sent to that device.

(410) NOTE: Sent Telnet break to device

CAUSE: Indicates the sending of a Telnet break sequence to the remote terminal server.

(411) NOTE: Sent STOP character to device

CAUSE: Indicates the sending of a STOP data flow character (usually control-S)
       to the device on the remote terminal server.

(412) NOTE: Sent START character to device

CAUSE: Indicates the sending of a START data flow character (usually control-Q)
       to the device on the remote terminal server.

(413) NOTE: Sent are you there sequence to device

CAUSE: Indicates the sending of a response to a Telnet are you there sequence
       to the remote terminal server.

(414) NOTE: Aborted output for application

CAUSE: Indicates the abortion of output from the application to the device
       on the remote terminal server.
```

```
(415) NOTE: Erased character or line for application

CAUSE: Indicates the erasure of a character or line of data from the device
       on the remote terminal server to the application.

(416) NOTE: Received terminal type subnegotiation <terminal_name>

CAUSE: Indicates the reception of the device type from the device on the
       remote terminal server.

(417) NOTE: Connected to device at node <node_name> TCP port number <tcp_port_number>

CAUSE: Indicates the location of the device on the remote terminal server to which connected.

(418) NOTE: Sent timing mark negotiation

CAUSE: Indicates the sending of a timing mark negotiation request to the remote terminal server.

(419) NOTE: Waiting <telnet_timer> seconds for timing mark negotiation reply

CAUSE: Indicates the number of seconds to wait for a timing mark negotiation
       reply from the remote terminal server.

(420) NOTE: Timing mark negotiation reply received

CAUSE: Indicates the successful completion of a timing mark negotiation.

(421) NOTE: Waiting <timeout> seconds for Telnet negotiation reply

CAUSE: Indicates the number of seconds to wait for a Telnet negotiation
       reply from the remote terminal server.

(422) NOTE: Sent status request

CAUSE: Indicates the sending of a status request to the device on the
       remote terminal server.

(423) NOTE: Waiting <status_timer> seconds for status reply

CAUSE: Indicates the number of seconds to wait for a status reply from the
       device on the remote terminal server.

(424) NOTE: Printer ready

CAUSE: Indicates that the printer is ready for print requests.
```

(425) NOTE: Sent binary negotiation

CAUSE: Indicates the sending of a binary negotiation request to the remote terminal server.

(426) NOTE: Waiting <telnet_timer> seconds for binary negotiation reply

CAUSE: Indicates the number of seconds to wait for a binary negotiation reply from the remote terminal server.

(427) NOTE: Binary negotiation accepted by remote side

CAUSE: Indicates the successful completion of the binary negotiation.

(428) NOTE: Sent interrupt signal to application

CAUSE: Indicates the sending of an interrupt to the application.

(429) NOTE: Sent break to application

CAUSE: Indicates the sending of a break to the application.

(430) NOTE: Pseudonym <pseudonym_name> does not exist

CAUSE: Indicates the non-existence of the pseudonym specified and its availability for use.

(431) NOTE: Deleted pseudonym <pseudonym_name> to make it available for use

CAUSE: Indicates the successful deletion of the pseudonym specified and its availability for use.

(432) NOTE: Deleted pseudonym <pseudonym_name>

CAUSE: Indicates the successful deletion of the pseudonym specified.

NOTE: This message accompanies messages (108) and (297).

```
(433) NOTE: Closed pty <master_pty_name>

CAUSE: Indicates the closure of the pty specified after the deletion of its
       associated pseudonym.

(434) NOTE: Closing connection due to application completion

CAUSE: Indicates the clean closure of the connection to the remote terminal
       server after the close timer had expired.

(435) NOTE: Received <telnet_command> <recognized_telnet_option> and <reply_flag>

(436) NOTE: Received <telnet_command> <unrecognized_telnet_option> and  <reply_flag>

CAUSE: Indicate the reception of a Telnet command and option.

(437) NOTE: Sent <telnet_command> <recognized_telnet_option>
(438) NOTE: Sent <telnet_command> <unrecognized_telnet_option>

CAUSE: Indicate the sending of a Telnet command and option.
```

Index