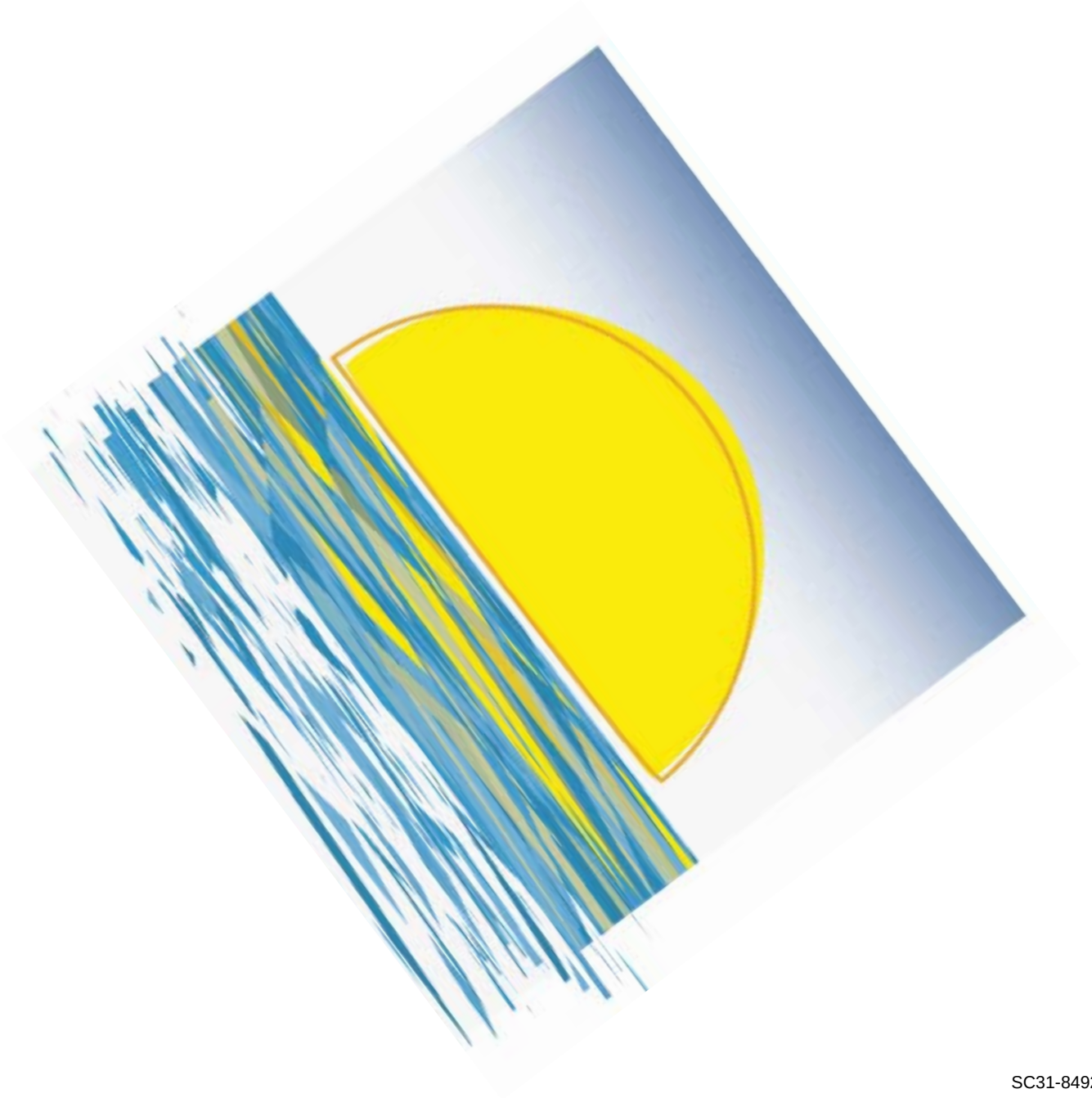
OS/390 TCP/IP OpenEdition

Diagnosis Guide



SC31-8492-00

OS/390 TCP/IP OpenEdition

Diagnosis Guide

SC31-8492-00

Note:

Before using this information and the product it supports, be sure to read the general information under Appendix E, “Notices” on page 173.

First Edition (June 1997)

This edition applies to OS/390 (5645-001) and OS/390 TCP/IP OpenEdition. See the “Summary of Changes” for a description of the changes made in this edition. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. A form for readers' comments may be at the back of this publication. If the form has been removed, you may send your comments to the following address:

International Business Machines Corporation
Department CGMD
P.O. Box 12195
Research Triangle Park, North Carolina 27709
USA

If you prefer to send comments electronically, use one of the following methods:

Fax (USA and Canada):	1-800-227-5088
Internet e-mail:	usib2hpd@vnet.ibm.com
World Wide Web:	http://www.s390.ibm.com/os390
IBMLink:	CIBMORCF at RALVM13
IBM Mail Exchange:	USIB2HPD at IBMMAIL

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

About This Book

This book is intended to provide information for diagnosing problems occurring in IBM's OS/390 Transmission Control Protocol/Internet Protocol (TCP/IP). This book can help you determine whether a specific problem is a result of the implementation, and how to report it to the IBM Software Support Center. Internal product information is provided as additional guidance for diagnosing problems with OS/390 TCP/IP OpenEdition.

OS/390 TCP/IP OpenEdition is an integral part of the OS/390 family of products. For an overview and mapping of the documentation available for OS/390, see the *OS/390 Information Roadmap*.

Who Should Use This Book

This book is used by system programmers for diagnosing problems. You should use this book to:

- Analyze a problem in a OS/390 TCP/IP OpenEdition implementation
- Classify the problem as a specific type
- Describe the problem to the IBM Software Support Center

You should be familiar with OS/390 TCP/IP and the protocol commands to use this book.

What is Not in This Book

This book **does not** cover the following topics, which are not supported in OS/390 TCP/IP OpenEdition:

- SNALINK LU0, SNALINK LU 6.2, and X.25 NPSI trace subcommands
- LESSTRACE, MORETRACE, TRACE and NOTRACE commands, which are replaced by CTRACE

Where to Find Related Information on the Internet

You may find the following information helpful.

For current updates to the TCP/IP Version 3 Release 2 for MVS documentation described in “Bibliography” on page 175, check out the TCP/IP for MVS home page :

`http://www.networking.ibm.com/tcm/tcmprod.html`

To keep in close touch with OS/390, we suggest you look at the OS/390 home page :

`http://www.s390.ibm.com/os390`

To keep abreast of new products and technologies from IBM Networking, take a look at the IBM Networking home page :

`http://www.networking.ibm.com/`

The IBM Networking Software Glossary is now available in HTML format as well as PDF. You can access it directly at the following URL:

`http://www.networking.ibm.com/nsg/nsggls.htm`

How to Contact IBM Service

For telephone assistance in problem diagnosis and resolution (in the United States or Puerto Rico), call the IBM Software Support Center anytime (1-800-237-5511). You will receive a return call within 8 business hours (Monday – Friday, 8:00 a.m. – 5:00 p.m., local customer time).

Outside of the United States or Puerto Rico, contact your local IBM representative or your authorized IBM supplier.

Summary of Changes

Summary of Changes
for SC31-8492-00

This is the first edition of this book. It contains information previously presented in *TCP/IP for MVS: Diagnosis Guide*, LY43-0105-02, which supports TCP/IP Version 3 Release 2 for MVS. This book is new for OS/390 TCP/IP OpenEdition, which provides OpenEdition function for TCP/IP in the OS/390 environment. For information about previously available TCP/IP function, continue to use the TCP/IP Version 3 Release 2 for MVS library.

This book describes:

- Diagnosis procedures for OE Features (OE Telnet, OE FTP, and so forth). This information was previously found in *TCP/IP Version 3 for OpenEdition MVS: Applications Feature Guide* (SC31-8069).
- Information on using OE NETSTAT and OE PING to diagnose network connectivity problems.
- Diagnosis procedures for SNMP and OE RouteD.
- Information on using CTRACE to produce trace output. CTRACE now features the following enhancements:
  - Eliminates B37 abends (using the CTRACE external writer)
  - Allows merge of TCP/IP CTRACE data with GTF packet trace data
  - Allows other product CTRACE/GTF trace data to be merged with TCP/IP trace data
  - Supports multiple data sets
  - Supports external control of tracing from a console

Overview of the Diagnosis Procedure

Chapter 1. Overview of the Diagnosis Procedure

To diagnose a problem suspected to be caused by OS/390 TCP/IP OpenEdition, you first identify the problem, then determine if it is a problem with TCP/IP, and, finally, if it is a problem with TCP/IP, gather information about the problem so that you can report the source of the problem to the IBM Software Support Center.

With this information available, you can work with IBM Software Support Center representatives to solve the problem. The object of this book is to help you identify the source of the problem.

Figure 1 on page 4 summarizes the procedure to follow to diagnose a problem. The text following the figure provides more information about this procedure.

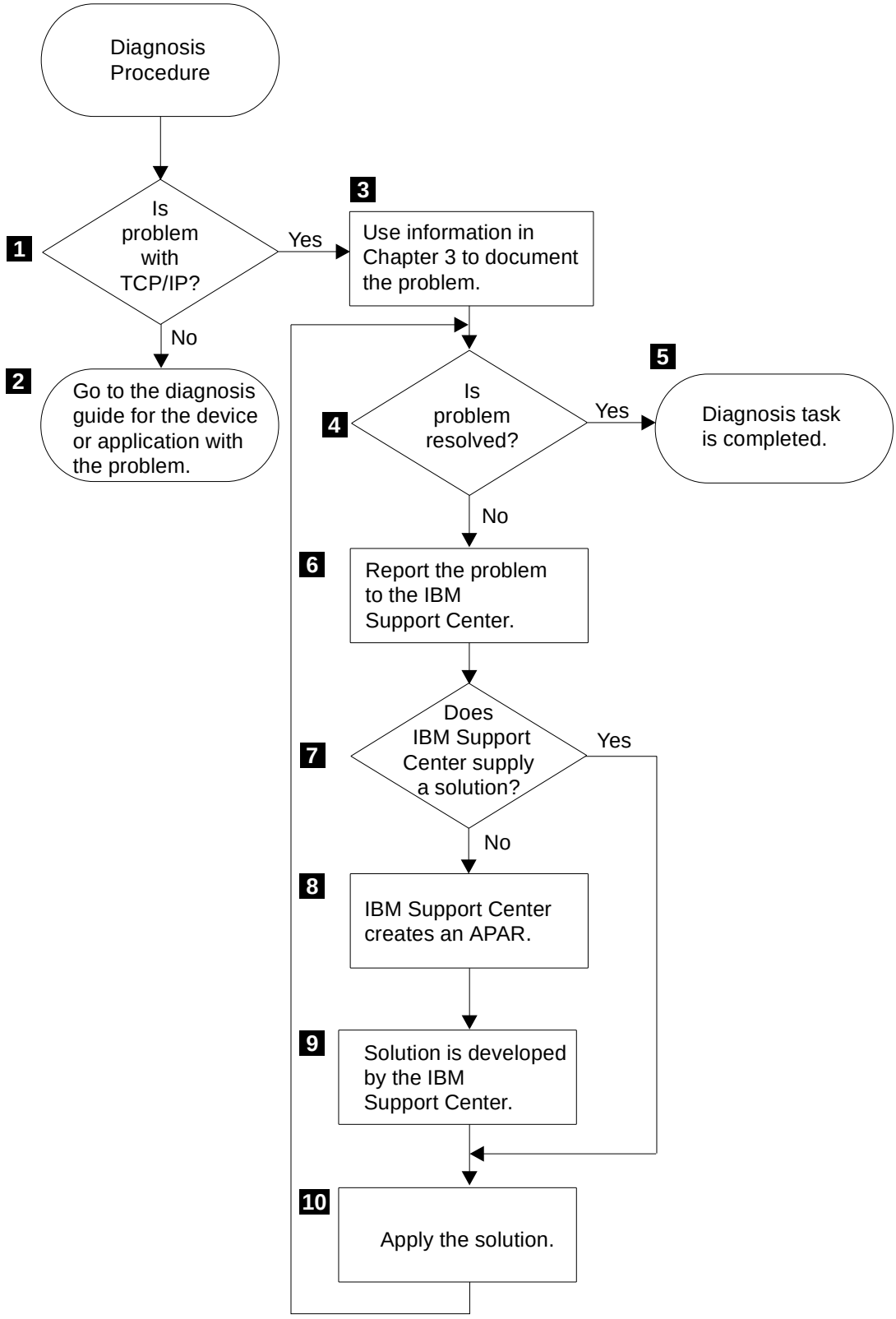


Figure 1. Overview of the Diagnosis Procedure

1 Determine if the source of the problem is TCP/IP.

Various messages appearing in the console log or in the SYSPRINT or SYSERROR data sets, together with alerts and diagnostic aids provide infor-

mation that helps you to find the source of a problem. If the problem is with TCP/IP, go to Step 3 ; otherwise, go to Step 2 .

2 Check appropriate books.

Refer to the diagnosis guide of the hardware device or software application that has the problem.

3 Gather information.

Refer to Chapter 2, Selecting Tools and Service Aids, for a detailed explanation of diagnostic procedures and how to collect information relevant to the problem.

4 Try to solve the problem.

If you cannot solve the problem, go to Step 6 .

5 The diagnosis task is completed.

The problem has been solved.

6 Report the problem to the IBM Software Support Center.

After you have gathered the information that describes the problem, report it to the IBM Software Support Center. If you are an IBMLINK user, you can perform your own RETAIN* searches to help identify problems. Otherwise, a representative uses your information to build keywords to search the RETAIN* database for a solution to the problem.

The object of this keyword search using RETAIN is to find a solution by matching the problem with a previously reported problem. When IBM develops a solution for a new problem, it is entered into RETAIN with a description of the problem.

7 Work with IBM Support Center representatives.

If a keyword search matches a previously reported problem, its solution might also correct the problem. If so, go to Step 10 . If a solution to the problem is not found in the RETAIN database, the IBM Software Support Center representatives will continue to work with you to solve the problem. Go to Step 8 .

8 Create an APAR.

If the IBM Software Support Center does not find a solution, they will create an authorized program analysis report (APAR) on the RETAIN database.

9 A solution is developed by the IBM Software Support Center.

Using information supplied in the APAR, IBM Software Support Center representatives determine the cause of the problem and develop a solution for it.

10 Apply the solution.

Apply the corrective procedure supplied by the IBM Software Support Center to correct the problem. Go to Step 4 to verify that the problem is corrected.

Selecting Tools and Service Aids

Chapter 2. Selecting Tools and Service Aids

This chapter introduces the tools and service aids that MVS TCP/IP provides for diagnosis. For the purposes of this book, *tools* includes dumps and traces, while *service aids* includes the other facilities provided for diagnosis. For example:

- SVC dump and system trace are tools.
- LOGREC data set and IPCS are service aids.

Major Topics: There are two topics in this chapter:

- "How Do I Know Which Tool or Service Aid to Select?" This topic lists problem types and matches them with the appropriate tool or service aid. Use this topic to select the tool or service aid you need for a particular problem.
- "What Tools and Service Aids Are Available?" on page 11. This topic describes each tool and service aid, including when to use it for diagnosis. Use this topic when you need an overview of tools and service aids or to find the appropriate time to use a particular tool or service aid.

How Do I Know Which Tool or Service Aid to Select?

This topic provides criteria for selecting a tool or service aid, depending on the problem or need. There are three tables:

- Selecting a Dump, Table 1
- Selecting a TCP/IP Component Trace, Table 2 on page 10
- Selecting a System Service Aid, Table 3 on page 10

The tables show the problem, the corresponding tool or service aid, and the chapter or book that covers it in complete detail. Use these tables to find a tool or service aid quickly.

Refer to "Guidelines for Machine-Readable Documentation" on page 16 for information about submitting dumps and traces to the IBM* Software Support Center.

Note: The traces given in this book are only examples. Traces in your environment can differ from these examples because of different options selected.

Table 1. Selecting a Dump

What Is the Problem?	Type of Dump to Use
Abnormal end of an authorized program or a problem program	ABEND dump See "Abends" on page 21 for detailed information.
TCP/IP server or client address space stops processing or is stopped by the operator because of slowdown or looping	SVC dump The SVC dump is created using the DUMP command. See "Analyzing a Loop" on page 22 for detailed information.

Table 2. Selecting a Trace

What Is the Problem?	Type of Trace to Use	Trace Output Location
Network Connectivity See "Network Connectivity" on page 25, for detailed information.	oping trace, ARP trace (onetstat –R)	For information on oping, see "Using OE PING (oping)" on page 30. For information on onetstat –R, see "onetstat -R" on page 34.
	Packet trace Refer to Chapter 6, Generating an IP Packet Trace, for detailed information about packet trace.	GTF-managed data set
OE FTP Server See Chapter 7, "OE File Transfer Protocol (OE FTP) Diagnosis" on page 79, for detailed information.	OE FTP server trace	Server traces appear on the console if syslogd is not started. If it is started, traces appear in the file designated in the syslog.conf file. Refer to Appendix C, "Description of Syslog Daemon (syslogd)" on page 163
OE Telnet	OE Telnet traces	See "OE Telnet Diagnosis" on page 101, for detailed information.
OE RouteD See Chapter 12, OE RouteD Diagnosis, for detailed information.	OE RouteD trace	Defaults to syslogd. User can use other parameters to send output to the STDOUT DD statement in the OROUTED cataloged procedure. Refer to "Where to Send OE RouteD Trace Output" on page 149.
OE REXEC See Chapter 9, "OE REXEC, OE REXECD, and OE RSHD Diagnosis" on page 111.	OE REXEC debug trace	syslogd.
OE REXECD See Chapter 9, "OE REXEC, OE REXECD, and OE RSHD Diagnosis" on page 111.	OE REXECD debug trace	syslogd.
OE RSHD See Chapter 9, "OE REXEC, OE REXECD, and OE RSHD Diagnosis" on page 111.	OE RSHD debug trace	syslogd.

Table 3 (Page 1 of 2). Selecting a Service Aid

What Is the Problem?.	Type of Service Aid to Use
System or hardware problem: need a starting point for diagnosis or when diagnosis requires an overview of system and hardware events in chronological order.	LOGREC data set or EREP See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.
Information about the contents of load modules and program objects or a problem with modules on the system.	AMBLIST See *OS/390 MVS Diagnosis: Tools and Service Aids* for detailed information.

Table 3 (Page 2 of 2). Selecting a Service Aid	
What Is the Problem?.	**Type of Service Aid to Use**
Diagnosis requires a trap to catch problem data while a program is running.	Service Level Indication Processing (SLIP) See *OS/390 MVS System Commands*, GC28-1781-02 for detailed information.
Diagnosis requires formatted output of problem data, such as a dump or trace.	IPCS See *OS/390 MVS IPCS User's Guide*, GC28-1756-02 for detailed information.

What Tools and Service Aids Are Available?

This topic provides an overview of the tools and service aids in a little more detail. The tables that follow contain a brief description of each tool or service aid, some reasons why you would use it, and a reference to the chapter or book that covers the tool or service aid in detail. (Most of the detailed information on tools and service aids is in this book.) The tools and service aids are covered in three tables:

- Description of Dumps, Table 4 on page 12.
- Description of Traces, Table 5 on page 13.
- Description of Service Aids, Table 6 on page 14.

In each table, the dumps, traces, or service aids are listed in order by frequency of use.

Note: The traces given in this book are only examples. Traces in your environment can differ from these examples because of different options selected.

Dumps

Table 4 on page 12 describes the types of dumps that can be used.

Table 4 (Page 1 of 2). Description of Dumps

Type of Dump	Description
ABEND Dump	Use an ABEND dump when ending an authorized program or a problem program because of an uncorrectable error. These dumps show: • The virtual storage for the program requesting the dump. • System data associated with the program. The system can produce three types of ABEND dumps— SYSABEND, SYSMDUMP, and SYSUDUMP. Each one dumps different areas. Select the dump that gives the areas needed for diagnosing your problem. The IBM-supplied defaults for each dump are: • SYSABEND dumps: The largest of the ABEND dumps, containing a summary dump for the failing program plus many other areas useful for analyzing processing in the failing program. • SYSMDUMP dumps: Contains a summary dump for the failing program, plus some system data for the failing task. In most cases, SYSMDUMP dumps are recommended, because they are the only ABEND dumps that are formatted with IPCS. • SYSUDUMP dumps: The smallest of the ABEND dumps, containing only data and areas about the failing program. **Reference:** See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.
SVC Dumps	SVC dumps can be used in two different ways: • Most commonly, a system component requests an SVC dump when an unexpected system error occurs, but the system can continue processing. • An authorized program or the operator can also request an SVC dump when they need diagnostic data to solve a problem. SVC dumps contain a summary dump, control blocks and other system code, but the exact areas dumped depend on whether the dump was requested by a macro, command, or SLIP trap. SVC dumps can be analyzed using IPCS. **Reference:** See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.

Table 4 (Page 2 of 2). Description of Dumps	
Type of Dump	**Description**
Stand-Alone Dump	Use a stand-alone dump when: • The system stops processing. • The system enters a wait state with or without a wait state code. • The system enters an instruction loop. • The system is processing slowly. These dumps show central storage and some paged-out virtual storage occupied by the system or stand-alone dump program that failed. Stand-alone dumps can be analyzed using IPCS. **Reference:** See “Analyzing a Loop” on page 22 for detailed information.

Traces

Table 5 (Page 1 of 2). Description of Traces	
Trace	**Description**
Component Trace	Use a component trace when you need trace data to report a client/server component problem to the IBM Software Support Center. Component tracing shows processing between the client and server. **Reference:** See Chapter 5, “ TCP/IP MVS Component Traces and IPCS Support” on page 53 for detailed information.
GTF Trace	Use a Generalized Trace Facility (GTF) trace to show system processing through events occurring in the system over time. The installation controls which events are traced. Use GTF when you're familiar enough with the problem to pinpoint the one or two events required to diagnose your system problem. GTF can be run to an external data set. Packet trace uses a GTF trace configured with the USR parameter. **Reference:** See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.
Master Trace	Use the master trace to show the messages to and from the master console. Master trace is useful because it provides a log of the most recently issued messages. These can be more pertinent to your problem than the messages accompanying the dump itself. You can either take a dump or write this trace to GTF. **Reference:** See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.

Table 5 (Page 2 of 2). Description of Traces	
Trace	**Description**
VTAM Trace	OS/390 TCP/IP OpenEdition uses two VTAM components, CSM and MPC. VTAM traces contain entries for many TCP/IP events, especially I/O and storage requests. **Reference:** See *VTAM Diagnosis*, LY43-0078-00 for detailed information.
System Trace	Use system trace to see system processing through events occurring in the system over time. System tracing is activated at initialization and, typically, runs continuously. It records many system events, with minimal details about each. The events traced are predetermined, except for branch tracing. You can either take a dump or write this trace to GTF. **Reference:** See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.
OE Applications	Open Edition applications send debug and trace output to syslogd. For more information on individual components such as OE FTP or OE RouteD, refer to those chapters in this manual. **Reference:** See Appendix C, “Description of Syslog Daemon (syslogd)” on page 163 for more information.

System Service Aids

Table 6 (Page 1 of 2). Description of Service Aids	
Service Aid	**Description**
AMBLIST	Use AMBLIST when you need information about the contents of load modules and program objects or you have a problem related to the modules on your system. AMBLIST is a program that provides lots of data about modules in the system, such as a listing of the load modules, map of the CSECTs in a load module or program object, list of modifications in a CSECT, map of modules in the LPA, and a map of the contents of the DAT-on nucleus. **Reference:** See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.

Table 6 (Page 2 of 2). Description of Service Aids

Service Aid	Description
Common Storage Tracking	Use common storage tracking to collect data about requests to obtain or free storage in CSA, ECSA, SQA, and ESQA. This is useful to identify jobs or address spaces using an excessive amount of common storage or ending without freeing storage. Use Resource Measurement Facility* (RMF*) or the IPCS VERBEXIT VSMDATA subcommand to display common storage tracking data. **References:** • See the *RMF User's Guide*, SC28-1949-00 for more information about RMF. • See the *OS/390 MVS Initialization and Tuning Guide*, SC28-1751-02 for detailed information about requesting common storage tracking. • See the VSM chapter of the F for information about the IPCS VERBEXIT VSMDATA subcommand.
IPCS	Use IPCS to format and analyze dumps, traces, and other data. IPCS produces reports that can help in diagnosing a problem. Some dumps, such as SNAP and SYSABEND and SYSUDUMP ABEND dumps, are preformatted and are not formatted using IPCS. **Reference:** See the *OS/390 MVS IPCS User's Guide*, GC28-1756-02 for detailed information.
LOGREC Data Set	Use the LOGREC data set as a starting point for problem determination. The system records hardware errors, selected software errors, and selected system conditions in the LOGREC data set. LOGREC information gives you an idea of where to look for a problem, supplies symptom data about the failure, and shows the order in which the errors occurred. **Reference:** See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for detailed information.
SLIP Traps	Use serviceability level indication processing (SLIP) to set a trap to catch problem data. SLIP can intercept program event recording (PER) or error events. When an event that matches a trap occurs, SLIP performs the problem determination action that you specify: • Requesting or suppressing a dump • Writing a trace or a LOGREC data set record • Giving control to a recovery routine • Putting the system in a wait state **Reference:** See the SLIP command in *OS/390 MVS System Commands*, GC28-1781-02 for detailed information.

Guidelines for Machine-Readable Documentation

If, after talking to the Level 2 Support Center representative about a problem, it is decided that documentation should be submitted to the TCP/IP support team, it might be more convenient for the customer or the TCP/IP support team that documentation be submitted in machine-readable form (that is, on tape). Machine-readable documentation can be handled most efficiently by the IBM Software Support Center if it conforms to the following guidelines when creating the tape (or tapes).

When preparing machine-readable documentation for submission in an MVS environment, the following guidelines should be followed:

1. Submit dumps and traces on tape.
   - For dumps:

     Dump data should not be formatted in any way prior to or during the transfer of the dump to tape.
     The DCB parameters of the dump data set should not be changed. The DCB parameters should be:
     LRECL=4160, BLKSIZE=4160, RECFM=F (for OS/390)

     The IPCS commands COPYDUMP and COPYTRC can also be used. For more information, see *IPCS Command Reference*.
   - For GTF traces:

     GTF trace data should be moved from the trace data set (which is usually SYS1.TRACE) to tape using IEBGENER only.
     The DCB parameters for a GTF trace should be:
     LRECL=4092, BLKSIZE=4096, RECFM=VB
     or
     LRECL=4092, BLKSIZE=32760, RECFM=VB

   For both traces and dumps, do not reblock the data (that is, do not use a different BLKSIZE value) when moving the information to tape. Only the DCB parameters shown above should be used.

   Note: Use of any other utility (IBM or non-IBM) to transfer dump or trace data to tape might result in a processing delay and could result in the APAR being returned to the customer (closed RET) due to the inability of the change team to process the tape.
2. Submit other types of information (such as TCP/IP traces, configuration files, console logs, and so forth) on paper or tape. **Tape is preferable**. If submitted on tape, the data should be written to tape using IEBGENER only. The DCB parameters used when writing this type of data to tape should be the same as the input data set (that is, the same DCB parameters as the source of the data).
3. Tapes that are submitted to the TCP/IP support team may be standard label (SL) or nonlabel (NL). Cartridge (3480) or reel tapes may be used. Each tape should contain an external label to identify the tape and its contents in some way. The problem number or APAR number should appear on the label. If multiple tapes, or multiple files on one tape, are used, a separate explanation should be included itemizing the contents of each tape or file.

4. Include the output from the job used to create each tape with the tapes. It is very important that the IBM Software Support Center have the output from the job that created the tape (not simply the JCL that was used) in order to verify that the tape was created correctly and that the job completed normally.

Necessary Documentation

Before you call the IBM Software Support Center, have the following information available:

Information	Description
Customer Number	The authorization code that allows you to use the IBM Software Support Center. Your account name, your TCP/IP license number, and other customer identification should also be available.
Problem Number	The problem number previously assigned to the problem. If this is your first call about the problem, the support center representative assigns a number to the problem.
Operating System	The operating system that controls the execution of programs (such as MVS/ESA). Include the MVS or OS/390 release level.
LE Runtime Library	The release level of the link edit runtime library is also needed if you are compiling user-written applications written in C or C++.
Component ID	A number that is used to search the database for information specific to TCP/IP. If you do not give this number to the support center representative, the amount of time taken to find a solution to your problem increases.
Release Number	An identification uniquely identifies each TCP/IP release.

The following tables provide TCP/IP-specific information that you should have before calling the IBM Software Support Center.

Table 7. TCP/IP Component Name and Release Level

Component Name and Release Level	System Maintenance Program	Field Maintenance Identifier/CLC
TCP/IP (MVS/ESA)	SMP/E	JTCP329

Table 8. TCP/IP Component ID

Licensed IBM Program	Component ID Number
OS/390 TCP/IP OpenEdition	5655HALR2M0-5753

A complex problem might require you to talk to several people when you report your problem to the IBM Support Center. Therefore, you should keep all the information that you have gathered readily available.

Note: You might want to keep the items that are constantly required, such as the TCP/IP component ID, in a file for easy access.

Diagnosing Abends, Loops, and Hangs

Chapter 3. Diagnosing Abends, Loops, and Hangs

Note: Information in this chapter is relevant only to OS/390 TCP/IP OpenEdition (V3R3). If you are having problems with a V3R2 stack, please refer to *TCP/IP for MVS: Diagnosis Guide*, LY43-0105-02.

This chapter contains information about abends, loops, and hangs. More information is given in the individual component chapters in this book.

Abends

An abend is an abnormal end. Several types of abends can occur:

Table 9. Types of Abends

Type of Abend	Description	Where to Find Help
User abends	User abends are generated by C runtime routines. They usually start with U409x.	See *OS/390 C/C++ IBM Open Class Library Reference*, SC09-2364-01.
Platform abends	Abend 3C5 and abend 4C5 are internal abends generated by TCP/IP. Note the reason code stored in register 15 and check the IBM database for known problems.	For further assistance, call the IBM Support Center.
System Abends	0C4, 0C1, and 878 are system abends.	See *OS/390 MVS System Codes*, GC28-1780-02
CEEDUMPs	LE produces certain types of abends detected for OE applications such as OE RouteD. CEEDUMPs are usually written to the current working directory in the hierarchical file structure.	See *Language Environment for OS/390 Debugging Guide and Run Time Messages* (SC28-1942).

General Information

A dump is usually produced when TCP/IP or a TCP/IP component address space abends. If an abend occurs and no dump is taken, the dump files or spools might be full or a SYSMDUMP DD statement might not have been specified in the failing procedure. If TCP/IP or a TCP/IP component was not able to complete the dump, you will have to re-create the abend or wait for it to occur again.

Note: For more information about debugging the abends and the system abends (for example; abends 0C4, 0C1, and 878), refer to *OS/390 MVS Diagnosis: Procedures* (LY28-1082).

Analyzing a Loop

If an operation, such as a message or printed output, repeats endlessly, TCP/IP could be in a loop. Some indicators of a loop are:

- Slow response time
- No response at all
- Inordinately high CPU utilization by TCP/IP

If the problem is a loop, use the following procedure to collect documentation.

1. **Get dump output**.
   - **Enabled**

     Get an SVC dump of TCP/IP or the looping TCP/IP component by issuing the DUMP command from the MVS system console, or press the Program Restart key. Refer to *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for more information about the DUMP command.

     Note: Please make sure that the following storage areas are dumped: CSM, VTAM's aspace and any dataspace used by the DLCS, RGN, CSA, LSQA, NUC, PSA, LPA, and TCPIPDS1. TCPIPDS1 is the data space containing the TCPIP component trace records. For information on dumping this data space, see “Obtaining Component Trace Data with a Dump” on page 58.

   - **Disabled**

     If the loop is disabled, the MVS system console is not available for input. Try the following:

     – Use a PSW RESTART to terminate a looping task. This process creates a LOGREC entry with a completion code of X'071'. Use the LOGREC record and the RTM work area to locate the failing module. Depending on the PSW bit 32, the last three bytes (24–bit mode) or four bytes (31–bit mode) contain the address being executed at the time of the dump. Scan the dump output to find the address given in the PSW. For more information on using PSW RESTART, refer to *VTAM Diagnosis*.

     – Take a stand-alone dump. Refer to *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for information about stand-alone dumps.

2. **Get the MVS system console log (SYSLOG), the job log from the started procedure, and the LOGREC output**.

   The MVS system console log might contain information, such as error messages, that can help you diagnose the problem. Also, print the LOGREC file.

   Use the LOGDATA option to print the in-core LOGREC buffers. See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 or *OS/390 MVS IPCS Commands*, GC28-1754-02 for more information about the LOGDATA option.

   Note: The SYSERROR data set might contain additional information to help you diagnose the problem.

3. **Is a message involved?**

   Determine whether there are any messages associated with the loop, such as a particular message always preceding the problem, or the same message

being issued repeatedly. If so, add the message IDs to your problem documentation.

4. **Examine the dump entries using IPCS**.

   By examining all of the trace entries in the system trace table, you might be able to determine whether there is a loop. The most obvious loops would be a module or modules getting continual control of the TCP/IP system.

   Use the PSW to determine the names of the modules in the loop. Refer to the *OS/390 MVS IPCS User's Guide*, GC28-1756-02 for information about using IPCS.

   In the output shown in Figure 2, the CLKC entries indicate an enabled loop. The PSW addresses on the CLKCs identify the looping program. Use the WHERE subcommand to locate the responsible program.

```
02-0029 008E7220   CLKC      078D2600 83A8192C  00001004  00000000

02-0029 008E7220   CLKC      078D2600 83A81934  00001004  00000000

02-0029 008E7220   CLKC      078D2600 83A81930  00001004  00000000

02-0029 008E7220   CLKC      078D2600 83A8192A  00001004  00000000

02-0029 008E7220   CLKC      078D2600 83A81930  00001004  00000000

02-0029 008E7220   CLKC      078D2600 83A81938  00001004  00000000
```

Figure 2. Example of Output from VERBEXIT TRACE

Analyzing Hangs

If the problem is a hang, use the following procedure to collect documentation:

1. **Determine the extent of the hung state in the operation of the TCP/IP network.**

   Determine whether all TCP/IP processing stopped or only processing with respect to a single device, or something in between. Also determine what, if any, recovery action was taken at the time the hang was encountered by the operator or user. Some information about the activity that immediately preceded the hang might be available on the system log or in application program transaction logs.

2. **Does TCP/IP respond to any commands?**

   Determine if TCP/IP responds to commands, such as oping or onetstat. If TCP/IP does not respond to these commands, take an SVC dump of TCP/IP address space and contact the IBM Software Support Center. If TCP/IP does respond to the commands, it is not hung.

3. **Is a particular application (such as OE FTP or a user-written application) hung?**

   Take a dump of the OMVS address space, the TCP/IP address space, and the application address space.

Network Connectivity

Chapter 4. Network Connectivity

This chapter is divided into two sections. The first section contains specific guidance for diagnosing network problems with OS/390 TCP/IP. The second section is a general overview of internetworking in the TCP/IP environment. It is an introduction to the protocols and functions of IP routing.

Diagnosing Network Connectivity Problems

Interconnectivity between network hosts encompasses the physical layer or hardware layer, the protocols such as TCP and IP, and the applications that use the services of TCP and IP. Isolating network problems is an essential step in successful implementation of a network application.

Notes:

1. Throughout this chapter, the host IP address is used. The RESOLVER and NAMESERVER functions, which translate symbolic names to IP addresses, should be avoided when diagnosing network problems.
2. The otracert command can also be helpful in diagnosing network connectivity problems. See “Using OE Traceroute (otracert)” on page 36 for more information.
3. “Data sets,” are used with a non-OpenEdition stacks (V3R2). They are written in the capital letters (such as *hlq*.TCPIP.DATA). “Files” come from the hierarchical file system (HFS) used in OpenEdition stacks (V3R3); they are written in lower-case (such as /etc/hosts).

A Note on Multiple Stacks

If you are running multiple stacks, the first question to ask is "Am I addressing the right stack?" Use `onetstat -h -p` *`procname`* to view host addresses for the stack you're using. Make sure the correct values are coded on the HOME statement in the *hlq*.PROFILE.TCPIP data set.

Note to Users Running V3R3 and V3R2 stacks: You cannot run onetstat against a V3R2 stack. Use NETSTAT instead. For details, see *TCP/IP for MVS: Diagnosis Guide*, LY43-0105-02.

If you are not addressing the right stack, change the target address in your command. In the case of OpenEdition RouteD connectivity errors for multiple stacks, see “OE RouteD Connection Problems” on page 143. In the case of SNMP connectivity problems, see “SNMP Connection Problems” on page 125.

It is also helpful to understand the search order used by OS/390 TCP/IP OpenEdition. Refer to Appendix A, “A Note on Search Paths” on page 159, or to *OS/390 TCP/IP OpenEdition Planning and Release Guide*.

For more information on running multiple stacks, refer to *OS/390 TCP/IP OpenEdition Configuration Guide*, SC31-8304-00 or *Accessing OS/390 OpenEdition MVS from the Internet* (SG24-4721).

Problems Connecting to the Server

Figure 3 shows the steps to take in diagnosing server connection problems.

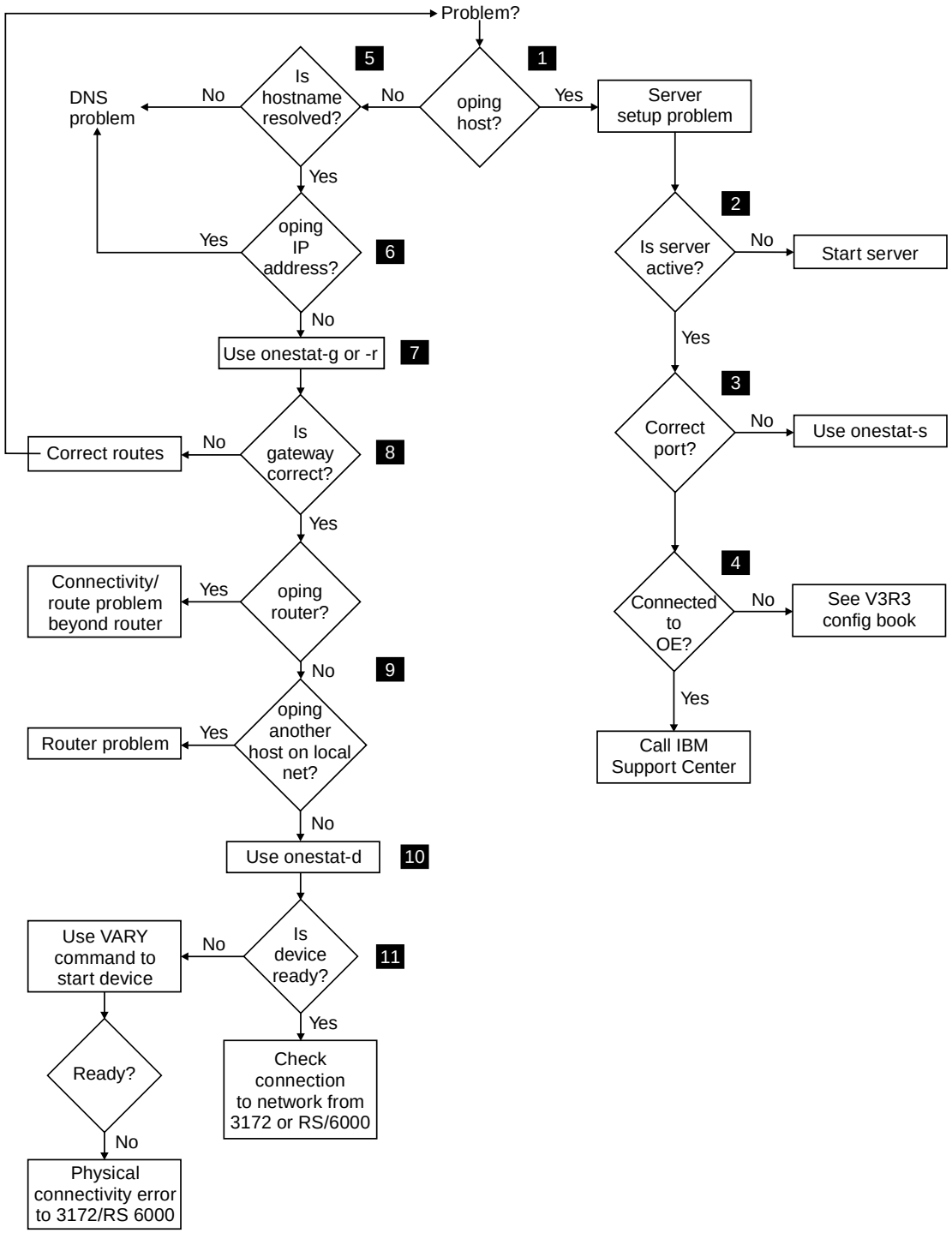


Figure 3. Overview of the Diagnosis Procedure for Server Connection Problems

The following explains the steps shown in Figure 3.

1. Can I reach the other host? Use `oping <hostname>` (see Using OE PING (oping)). If you are successful, go to Step 5 on page 29. If not, go to Step 2 on page 29.

2. Is the server started? If not, start the server. Go to Step 3.
3. Is the server started on the port you think it's on? If not, use `onetstat -s` to determine what port the server is on. See *OS/390 TCP/IP OpenEdition User's Guide* for details. Go to Step 4.
4. Is the server properly connected to OE? If not, refer to *OS/390 TCP/IP OpenEdition Configuration Guide* for details. If yes, call the IBM Support Center.
5. Is the hostname resolved? If yes, go to Step 6. If no, do the following:

   Check the resolver trace. The first thing the resolvers do is look for the value of the RESOLVER_CONFIG environment variable. This can be set in the JCL of the started proc by way of the parameter list, using the keyword ENVAR. Look for an example in *OS/390 TCP/IP OpenEdition Configuration Guide*. [1] If the hostname is not resolved, you have a name resolution problem. Check the following files to be sure the name is correct:

   - /etc/revolv.conf or *hlq*.TCPIP.DATA
   - /etc/hosts
   - *hlq*.HOSTS.LOCAL

   There are several other files you need to check. See *OS/390 TCP/IP OpenEdition Configuration Guide* for details.
6. If the hostname resolves, try to `oping` the IP address (see “Using OE PING (oping)” on page 30). If you can, you have a name resolution problem. If not, go to Step 7.
7. If you cannot `oping` the IP address, use `onetstat -r` to display routes to the network. (See “onetstat -r” on page 33.) Go to Step 8.
8. `oping` to the gateway (router). If that does not work, go to Step 9. If it works, you have a route problem. To resolve this, do the following:
   a. Verify routes at your host.
   b. Check for connectivity/route problems beyond the router.
9. If the `oping` to the gateway fails, either the router is down, or you are not properly connected to the network. `oping` to another host in the local network. If that works, the router is down. If it does not work, go to Step 10.
10. If the `oping` does not work, use `onetstat -d` to verify device status. (See “onetstat -d” on page 32.) If the device is not ready, go to Step 11. If the device is ready, Check the physical connectivity to the network (i.e., 3172 or RS/6000 LAN connection).
11. If the device is not ready, try to start the device from `VARY TCPIP procname CMD=Obeyfile,DSN=datasetname` (which is documented in *OS/390 TCP/IP OpenEdition Configuration Guide*). If that is unsuccessful
    a. Verify connectivity to the device (channel connector).
    b. Check to see if you have a problem at the device itself.

[1] For more information on using the resolver trace, refer to Chapter 3 of *Accessing OS/390 Open/Edition MVS from the Internet* (SG24-4721).

Using OE PING (oping)

Issue one of the `oping` commands to determine the start of the problem. Since the command is case sensitive, be sure to use lowercase.

- **oping loopback to verify the installation of TCP/IP in OS/390 environment**

  oping loopback is essentially an internal software test. This command uses IP address 127.0.0.1, which is the standard *loopback* address. An IP packet is ***not*** sent to a physical device.

```
oping loopback
Ping V3R3: Pinging host  (127.0.0.1).
PING: Ping #1 response took 0.001 seconds.
***
```

- **oping a home address to verify the information from the onetstat -h command**. This is an internal software test. An IP packet is ***not*** sent to a physical device.

```
oping 193.9.200.1
Ping V3R3: Pinging host 193.9.200.1.
PING: Ping #1 response took 0.001 seconds.
***
```

- **oping a host on a directly-attached network to verify the following:**
  - The directly-attached network is defined correctly.
  - The device is properly connected to the network.
  - The device is able to send and receive packets on the network.
  - The remote host (193.9.200.2) is able to receive and send packets.

```
oping 193.9.200.2
Ping V3R3: Pinging host 193.9.200.2.
PING: Ping #1 response took 0.021 seconds.
***
```

- **oping a host on a remote network to verify the following:**
  - The route to the remote network is defined correctly.
  - The router is able to forward packets to the remote network.
  - The remote host is able to send and receive packets on the network.
  - The remote host (9.67.43.126) has a route back to the local host.

```
oping 9.67.43.126
Ping V3R3: Pinging host 9.67.43.126.
PING: Ping #1 response took 0.221 seconds.
***
```

Correcting oping Timed Out Problems

An oping Timed Out message can occur for many reasons; through various steps, an attempt will be made to isolate the problem to the local OS/390 server, or a remote host or router. Possible reasons for a time-out are:

- The device is not transmitting packets to the local network.
- The remote host is not receiving or transmitting packets on the network.
- The remote host does not have a route back to the local OS/390 server.
- An intermediate router or gateway is not correctly forwarding IP packets.
- The ipconfig reassembletimeout value may be set too low.

oping Command Return Codes

The following is a list of the return codes generated by the oping command:

Code	Description
0	Response on at least one attempt
4	No response
12	Socket API failure
100	Incorrect parameter

Using OE NETSTAT (onetstat)

Use `onetstat` commands to verify the TCP/IP configuration. Since it is case-sensitive, be sure to type the command in lowercase. The following commands are covered in this section:

- “onetstat -h”
- “onetstat -d” on page 32
- “onetstat -g” on page 33
- “onetstat -r” on page 33
- “onetstat -R” on page 34

For details about the onetstat command, refer to the *OS/390 TCP/IP OpenEdition User's Guide*, GC31-8305-00.

Notes:

1. You cannot run OE NETSTAT against a V3R2 stack. Use NETSTAT instead. For details, see *TCP/IP for MVS: Diagnosis Guide*, LY43-0105-02.
2. The default configuration data set is *hlq*.PROFILE.TCPIP. Refer to the TCPIP started task procedure, the PROFILE DD statement, for the name of the configuration data set.

onetstat -h

Use the onetstat -h command to verify the ADDRESS and LINK values returned by onetstat. Are the correct values coded on the HOME statement in the *hlq*.PROFILE.TCPIP data set? An example follows.

```
onetstat -h
MVS TCP/IP onetstat V3R3           TCPIP Name: TCPV3   12:36:45
Home address list:
Address            Link          Fig
9.67.113.29        TR2           P
9.67.116.15        CTCD06
```

onetstat -d

Use the onetstat -d to display the status and associated configuration values for a device and defined links, as coded in the *hlq*.PROFILE.TCPIP data set.

```
MVS TCP/IP ONetstat V3R3      TCPIP Name: TCPV3          12:34:56

  DevName: CTCMVS3          DevType: CTC       DevNum: 0CC2
    LnkName: LOCAL123         LnkType: CTC       Status: Ready
      NetNum: 1   QueSize: 0  ByteIn:  123000    ByteOut: 4000
    BSD Routing Parameters:
      MTU Size: 00000576           Metric: 00000000
      DestAddr: 8.67.113.22        SubnetMask: 255.255.0.0
    Packet Trace Setting:
      Protocol: *        TrRecCnt: number    PckLength: ABBREV size
      SrcPort:   *       DestPort:  *
      IpAddress: *                   Subnet:    ipaddressmask
```

The following is a description of the fields in this example:

DevName — The device name.

DevType — The device type. For device CTCMVS3, the device type is CTC.

DevNum — The device number.

LnkName — The link name. In this example, it is LOCAL123.

LnkType — The link type. LOCAL123 is a channel-to-channel (CTC) link.

Status — The status of the link. For device CTCMVS3, the status is `Ready`.

NetNum — This field is significant only for links on LCS and CTC devices. For CTC, a net number 0 indicates the read device and a net number 1 indicates the write device. For LCS, this is the link adapter number that corresponds to the 3172 or OSA configuration. For more information, see *OS/390 TCP/IP OpenEdition Configuration Guide*.

QueSize — The size of the queue. This field is significant only for links on LCS devices.

ByteIn — Number of bytes received, displayed in decimal . For device CTCMVS3, it is 123000.

ByteOut — Number of bytes transmitted, displayed in decimal .

BSD routing parameters — For more information, see *OS/390 TCP/IP OpenEdition User's Guide*.

Packet trace setting The effective values of each of the packet trace options for the link. This information is displayed only when packet trace settings are defined and set to on.

onetstat -g

The `onetstat -g` command displays the current routing tables for TCP/IP. In order to establish connectivity to a remote host, the remote host must also have a route back to the server. The following shows an example of `onetstat -g` output.

```
onetstat -g
 MVS TCP/IP Onetstat V3R3      TCPIP Name: TCPV3          12:34:56
 Known gateways:

 NetAddress      FirstHop        Link    Pkt Sz  Subnet Mask     Subnet Value
 ----------      --------        ----    ------  -----------     ------------
 Default         9.67.113.1      TR2     576                     <none>
 9.0.0.0         <direct>        TR2     2000    0.255.255.12    0.67.113.0
 9.67.116.16     <direct>        CTCD06  4000    HOST
READY
```

The `onetstat -g` command provides the following information about each gateway:

- Address of the network
- First hop address
- Link name used by the first hop
- Packet size used by the first hop
- Subnet mask and subnet value

If you note any errors, check *hlq*.PROFILE.TCPIP for the following:

- Make sure no statements were flagged in either the initial profile or in any subsequent VARY TCPIP commands. (For information on the VARY TCPIP command, refer to *OS/390 TCP/IP OpenEdition Configuration Guide*.)
- Make sure the HOME statement has been coded correctly.
- Make sure the GATEWAY entries correlate to a valid link name.
- Make sure there is a GATEWAY or DEFAULTNET entry that correlates to the NETWORK or HOST addresses available on the network.

onetstat -r

`onetstat -r` displays the following routing information:

Destination The address of a destination host or network

Gateway The gateway used in forwarding packets

Flags Flags identifying the state of the route

Refcnt The current number of active uses for the route

Interface The link name for the route (the Interface field)

Following is an example of `onetstat -r` output:

```
onetstat -r

MVS TCP/IP onetstat V3R3      TCPIP Name: TCPV3          12:34:56

Destination     Gateway         Flags  Refcnt  Interface
-----------     --------        -----  ------  ---------
9.67.116.16     9.67.113.61     UGH    000000  TR1
Default         9.255.255.10    UG     000001  TR2
9.0.0.0         0.255.255.12    UH     000000  CTCD06
9.67.113.10     9.67.113.43     U      000004  CTCD05
```

onetstat -R

Use `onetstat -R <net address>` to query the ARP cache for a given address. Use `onetstat -R ALL` to query an entire ARP cache table. Make sure a onetstat -R displays an ARP entry for the remote hosts.

The following shows an example of onetstat -R output.

```
onetstat -R 9.67.112.25

  MVS TCP/IP onetstat V3R3      TCPIP Name: TCPV3            12:34:56

  Querying ARP cache for address 9.67.112.25
  Link: TR1              IBMTR: 10005A0019F5
     Route info:  0000
***
```

The ARP entry for the host on a remote network will contain the destination IP address and the MAC address for the router.

Review the routing tables on the remote host, to ensure the host has a route back to the OS/390 server. This could be a HOST route or NETWORK route. Intermediate routers must also be configured correctly.

ARP Frame Format: Figure 4 shows the format of the ARP frame.

<table>
<tr><td colspan="2">0(0)
Hardware Type</td><td>2(2) Protocol Type
(x'0806')</td></tr>
<tr><td>4(4) Hardware
Address Length</td><td>5(5) Protocol
Address Length</td><td>6(6)
Operation</td></tr>
<tr><td colspan="3">8(8) Sender Hardware Address
(Bytes 0-3)</td></tr>
<tr><td colspan="2">12(C) Sender Hardware Address
(Bytes 4 and 5)</td><td>14(E) Sender IP Address
(Bytes 0-1)</td></tr>
<tr><td colspan="2">16(10) Sender IP Address
(Bytes 2 and 3)</td><td>18(12) Target Hardware Address
(Bytes 0-1)</td></tr>
<tr><td colspan="3">20(14) Target Hardware Address
(Bytes 2-5)</td></tr>
<tr><td colspan="3">24(18) Target IP Address
(Bytes 0-3)</td></tr>
</table>

Figure 4. ARP Frame Format

Where:

Field	Description
Hardware Type	Specifies the type of hardware interface.
Protocol Type	Specifies the type of protocol, in this case ARP.
Hardware Address Length	Specifies the length in bytes of the hardware addresses in this packet.
Protocol Address Length	Specifies the length in bytes of the protocol addresses in this packet.
Operation Code	Specifies whether this is an ARP request (X'01') or ARP reply (X'02').
Sender Hardware Address	Specifies the sender physical network hardware address.
Sender IP Address	Specifies the sender protocol address.
Target Hardware Address	Specifies the target physical network hardware address. For an ARP request, this field is undefined.
Target IP Address	Specifies the target protocol address.

Note: For more information on using onetstat -R, see *OS/390 TCP/IP OpenEdition User's Guide*.

Using OE RouteD (orouted)

Comparing OE RouteD to GATEWAY statements

The essential difference between running OE RouteD and using GATEWAY statements is that gateway statements are static, while OE RouteD is dynamic. That is, OE RouteD, OS/390 TCP/IP “learns” information about the network and updates the routing tables accordingly.

Review the information in “Dynamic Routing” on page 49 and “Dynamic Routing Tables” on page 50 when you are diagnosing OE RouteD problems. Refer to Chapter 12, “OE RouteD Diagnosis” on page 141, for more information.

The output from onetstat -g is especially important to review when running OE RouteD. When TCPIP and OE RouteD are first started, an onetstat -g will show only those routes defined in the BSDROUTINGPARMS and in the OE RouteD gateways file or data set . Within a minute or so, onetstat -g should display routes to other hosts and networks, which can be connected to via routers on the network. The onetstat -g output will usually stabilize, and only as changes occur on the network, will the routing tables change.

OE RouteD Problem Determination

Note: For information on OE RouteD traces, refer to “OE RouteD Traces and Debug Information” on page 147.

Follow these steps for OE RouteD problem determination:

1. onetstat -g does not display any routes. Verify the BSDROUTINGPARMS statement is coded correctly in the *hlq*.PROFILE.TCPIP data set. Verify that the routes are defined correctly in in the OE RouteD gateways file or data set .
2. onetstat -g initially displays routes, but they are deleted after a few minutes. Verify OE RouteD is started and is connected to the TCPIP address space. Review the OE RouteD output (on the console or in syslogd, depending on where you started OE RouteD.) Issue `onetstat -c`, and look for a connection to the OE RouteD address space. Is OE RouteD receiving RIP packets from the network? Review the "-t -t -t" trace. Make sure OE RouteD has superuser authority.
3. If you receive a “destination network unreachable” message, and OE RouteD appears to be receiving RIP packets properly, verify IGNOREICMPREDIRECTS is specified in the *hlq*.PROFILE.TCPIP data set.
4. Make sure the NOFWD(ASSORTEDPARMS) or NOFWDDATAGRAM(IPCONFIG) are also specified in the *hlq*.PROFILE.TCPIP data set.
5. Refer to Chapter 12, “OE RouteD Diagnosis” on page 141, for more information.

Using OE Traceroute (otracert)

OE Traceroute provides a useful routing diagnostic tool. It displays the route that a packet takes to reach the requested target. Trace starts at the first router and uses a series of UDP probe packets with increasing IP time-to-live (TTL) values to determine the sequence of routers that must be traversed in order to reach the target host.

The packetSize otracert option lets you increase the IP packet size to see how size affects the route that the otracert probe will take. It also shows the point of failure if a destination address cannot be reached.

For the complete syntax of the otracert command, refer to *OS/390 TCP/IP OpenEdition User's Guide*, GC31-8305-00.

Figure 5 shows a successful otracert to a valid destination.

```
 otracert 9.130.40.72
------------------------------------------------------------
Traceroute to 9.130.40.72 (9.130.40.72).
Use escape C sequence to interrupt
1 buzz-113.tcp.raleigh.ibm.com (9.67.113.1)  7 ms  56 ms  6 ms
2 b500-503-1.raleigh.ibm.com (9.37.32.1)  25 ms  66 ms  19 ms
3 rtp2wf-atm.raleigh.ibm.com (9.37.1.132)  8 ms  19 ms  23 ms
4 9.32.204.30 (9.32.204.30)  60 ms  16 ms  19 ms
5 9.32.1.69 (9.32.1.69)  49 ms  65 ms  48 ms
6 9.32.44.2 (9.32.44.2)  58 ms * *
7 pok008-1.pok.ibm.com (9.117.1.1)  41 ms  44 ms  58 ms
8 * sqv014-1.endicott.ibm.com (9.130.104.12)  148 ms  93 ms
9 gdlvm7.endicott.ibm.com (9.130.40.72)  91 ms  89 ms  87 ms
```

Figure 5. Example of otracert to Valid Destination

- Each line in this example shows a “hop” to a different router.

- An asterisk (*) represents a lost packet.
- The time displayed (7 ms 56 ms, and so on) is in milliseconds. It shows the round trip time calculated from when the probe was sent from otracert and when otracert received the ICMP reply. Each time displayed is independent; it is the time it takes for otracert to send a probe and receive a reply.

Figure 6 shows an example of otracert to an address that results in unreachable host (!H).

```
    otracert 1.1.1.1
-------------------------------------------------------------
TRACEROUTE TO 1.1.1.1 (1.1.1.1).
USE ESCAPE C SEQUENCE TO INTERRUPT
1 buzz-113.tcp.raleigh.ibm.com (9.67.113.1)  5 ms  5 ms  5 ms
2 b500-503-1.raleigh.ibm.com (9.37.32.1)  8 ms  12 ms  8 ms
3 rtp2wf-atm.raleigh.ibm.com (9.37.1.132)  8 ms  12 ms  11 ms
4 9.32.204.30 (9.32.204.30)  30 ms  9 ms  8 ms
5 9.32.204.30 (9.32.204.30)  14 ms !H * *
6 * * *
7 * * 9.32.204.30 (9.32.204.30)  10 ms !H
8 * * *
9 * * *
```

Figure 6. Example of otracert to Unreachable Host

Here are some flags that you may see when using otracert:

" unreachable port

!H unreachable host

!P unreachable protocol

Using `otracert -d` turns on debug information, as shown in the following example:

```
$ otracert -d hugo
 <EZACDTRT C> Line <1606>: SETIADDR: dest set to:
 <EZACDTRT C> Line <1606>: SETIADDR: sin_len:  16 sin_family:   2 sin_port
 sin_addr: 0x07432B3E
 Traceroute to hugo (11.7.45.62).
 Use escape C sequence to interrupt
 TRT1623 parsOE Traceroute: host = hugo
 TRT1624 parsOE Traceroute: retryCount = 3
 TRT1625 parsOE Traceroute: port = 4096
 TRT1626 parsOE Traceroute: waitTime = 5
 TRT1627 parsOE Traceroute: maxTTL = 30
 TRT1628 parsOE Traceroute: maxPacket = 40
 TRT1629 Parsed otracert parameters¨
 TRT0700 openSock: recv socket failed 111(12FC00B0): EDC5111I Permission Denied
 EZZ6354I OE Traceroute: Recv socket() error detected (111/12FC00B0)
 TRT1633 openSock() failed with rc = 12
```

Figure 7. Example of otracert -d with Debug Information

host name of the host you are trying to reach

retryCount Number of times the probe is sent with the same TTL value.

port The number of the unused port used at the destination (defaults to 4096).

waitTime The wait time in seconds (defaults to five).

maxTTL The number of "hops" the trace will take before it dies.

maxPacket The size of the probe packet.

Notes:

1. otracert uses the site tables for inverse name resolution rather than the Domain Name Server. See *OS/390 TCP/IP OpenEdition Configuration Guide*, SC31-8304-00 for more information about using the site tables. If a host name is found in the site table, it will be printed along with its IP address.
2. To use OE Traceroute sockets, you must have authority to use the `VARY TCPIP` command, which is documented in Chapter 3 of *OS/390 TCP/IP OpenEdition Configuration Guide*.

Contacting the IBM Branch Office

Persistent error conditions, in most cases, indicate an installation or configuration problem. Contact the local IBM Branch Office for installation assistance. If a software defect is suspected, collect the following information prior to contacting the IBM Software Support Center:

- *hlq*.PROFILE.TCPIP
- *hlq*.TCPIP.DATA
- Output from various onetstat commands
- Output from onetstat -r or oping traces
- A network diagram or layout
- Error messages received (Refer to *OS/390 TCP/IP OpenEdition Messages and Codes* for relevant information.)
- When using OE RouteD, OE RouteD "-t -t -t -t" trace
- When using OE RouteD, a copy of the OE RouteD gateways file or data set.

Overview of Internetworking

Note: The following information was current when this document went to press. For the most up-to-date information, refer to *TCP/IP Tutorial and Technical Overview*.

Networking with TCP/IP connects different networks so that they form one logical interconnected network. This large overall network is called an *inter-network*, or more commonly, an *internet*. Each network uses its own physical layer, and the different networks are connected to each other by means of machines that are called *internet gateways* or simply *gateways*.

Gateways transfer IP datagrams between networks. This function is called *routing*; therefore, the internet gateways are often called *routers*. Within this book, the terms router and gateway are synonymous; both refer to a machine that transfers IP datagrams between different networks.

Note: If IP datagrams are not passed properly over a bridge, none of the higher TCP/IP protocols or applications will work correctly. For a discussion of bridges, refer to *TCP/IP Tutorial and Technical Overview*.

Linking networks in this way takes place at the network level of International Organization for Standardization (ISO). It is possible to link networks at a lower level layer using *bridges*. Bridges link networks at the ISO data link layer. Bridges pass packets or frames between different physical networks regardless of the protocols contained within them. An example of a bridge is the IBM 8209, which can interconnect an Ethernet network and a Token-Ring network.

Note: A bridge does ***not*** connect TCP/IP networks together. It connects physical networks together that will still form the same TCP/IP network. (A bridge does ***not*** do IP routing.)

Figure 8 depicts a router and a bridge. The router connects Network 1 to Network 2 to form an internet.

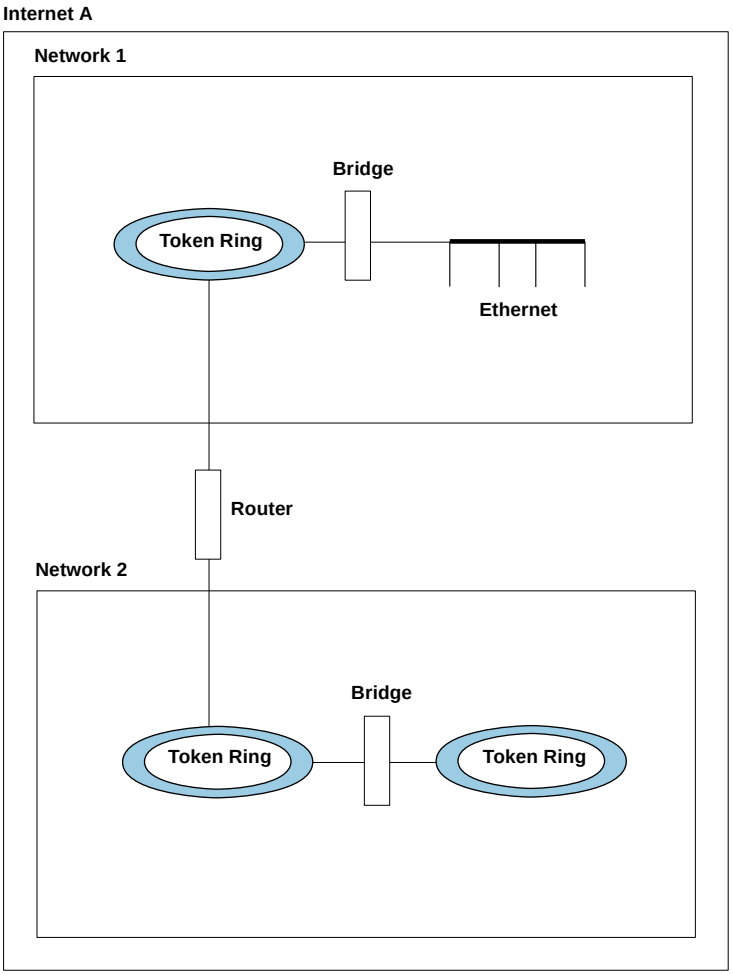


Figure 8. Routers and Bridges within an Internet

Maximum Transmission Unit (MTU)

Different physical networks have different maximum frame sizes. Within the different frames, there is a maximum size for the data field. This value is called the *maximum transmission unit* (MTU), or maximum packet size in TCP/IP terms.

Figure 9 on page 40 shows the relationship of MTU to frame size.

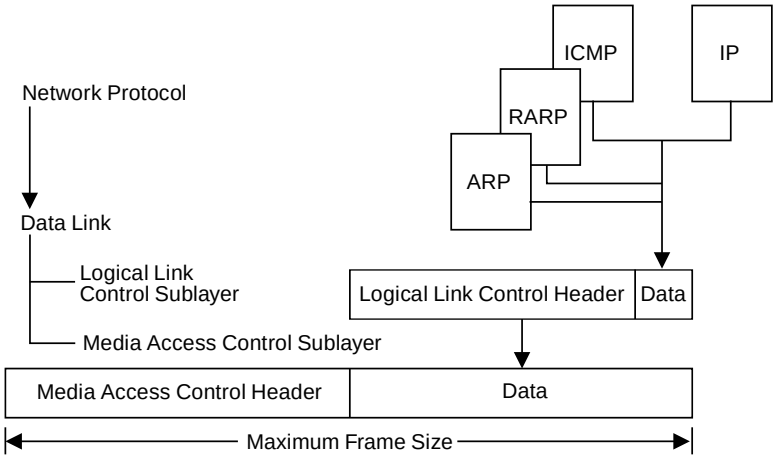


Figure 9. Relationship of MTU to Frame Size

If an IP datagram is to be sent out onto the network and the size of the datagram is bigger than the MTU, IP will fragment the datagram, so that it will fit within the data field of the frame. If the MTU is larger than the network can support, then the data is lost.

The value of MTU is especially important when bridging is used because of the different network limits. *RFC 791 - Internet Protocols* states that all IP hosts must be prepared to accept datagrams of up to 576 bytes. For this reason, use an MTU of 576 bytes if bridging (or routing) problems are suspected.

Note: MTU is equivalent to the MAX_PACKET_SIZE value on the GATEWAY statement, or the MTU value specified on BSDROUTINGPARMS when TRUE.

Fiber Distributed Data Interface (FDDI)

The FDDI specifications define a family of standards for 100 Mbps fiber optic LANs that provide the physical layers and media access control sub-layer of the data link layer as defined by the ISO/OSI Model.

IP-FDDI defines the encapsulating of IP datagrams and ARP requests and replies in FDDI frames.

All frames are transmitted in standard IEEE 802.2 LLC Type 1 Unnumbered Information format, with the DSAP and SSAP fields of the 802.2 header set to the assigned global SAP value for SNAP (decimal 170). The 24-bit Organization Code in the SNAP header is set to zero, and the remaining 16 bits are the EtherType from Assigned Numbers, that is:

- 2048 for IP
- 2054 for ARP

Typically, the MTU is set to 4352.

Mapping of 32-bit Internet addresses to 48-bit FDDI addresses is done via the ARP dynamic discovery procedure. The broadcast Internet addresses (whose <host address> is set to all ones) are mapped to the broadcast FDDI addresses (all ones).

IP datagrams are transmitted as a series of 8-bit bytes using the usual TCP/IP transmission order called “big-endian” or “network byte order.”

For more information on FDDI architecture, please refer to *LAN Concepts and Products* (GG24-3178).

Token Ring IEEE 802.5

When a token-ring frame passes through a bridge, the bridge adds information to the routing information field (RIF) of the frame (assuming that the bridge supports source route bridging). The RIF contains information concerning the route taken by the frame and, more importantly, the maximum amount of data that the frame can contain within its data field. This is called the maximum information field (I-field). The value specified for the maximum I-field is sometimes referred to as the largest frame size, but this means the largest frame size ***excluding*** headers. See Figure 10 for details on the relationship of the I-field to the header fields.

Note: It is important to be aware that IBM's implementation limits the number of bridges through which a frame can be passed to seven. An attempt to pass a frame through an eighth bridge will fail.

The maximum I-field is always decreased by a bridge when it cannot handle the value specified. So, for a given path through a number of token-ring bridges, the maximum I-field is the largest value that ***all*** of the bridges will support. This value is specified in the Routing Control (RC) field within the RIF as shown in Figure 10.

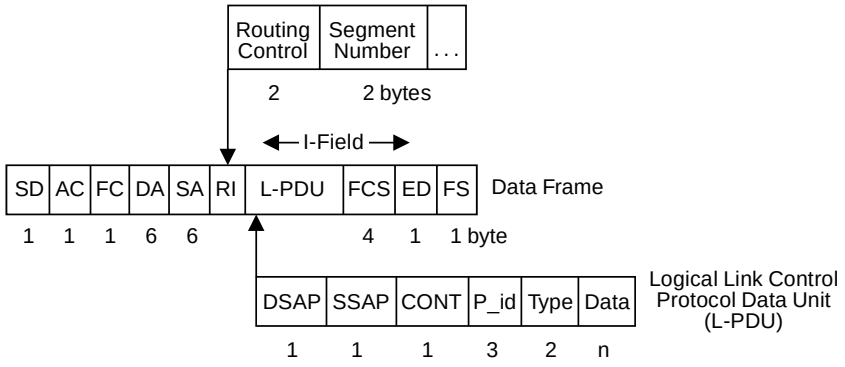


Figure 10. Format of an IEEE 802.5 Token-Ring Frame

The size of the MTU is the maximum amount of data that is allowed within a frame. The token-ring architecture specifies the maximum value of the I-field in the data frame, which corresponds to the maximum size of the L-PDU. The maximum I-field is determined by the bit configuration in the RC field, and is present in all routed frames.

Table 10 on page 42 shows the relationship between the RC field and the maximum I-field values.

Table 10. Relationship between RC Field and Maximum I-Field Value	
Routing Control Field	**Maximum I-Field in Bytes**
x000 xxxx xxxx xxxx	516
x001 xxxx xxxx xxxx	1500
x010 xxxx xxxx xxxx	2052
x011 xxxx xxxx xxxx	4472
x100 xxxx xxxx xxxx	8144
x101 xxxx xxxx xxxx	11407
x110 xxxx xxxx xxxx	17800

In Figure 10 on page 41 we can see that, within the L-PDU, the Logical Link Control (LLC) header uses eight bytes. Thus the MTU value is eight bytes less than the maximum I-field. (Note that the L-PDU contains a SNAP header, as described in “Sub-Network Access Protocol (SNAP)” on page 43.) Follow this example to calculate the MTU for a token ring. The token-ring bridges always adjust the value of the maximum I-field to that of the smallest one in the path. Ensure that the MTU value is less than the value specified by the bridge.

Typically, within a 4Mbps token-ring network, the value of maximum I-field will be 2052 bytes, and so the MTU would be set to 2044 bytes (2052 minus 8 bytes for the LLC header).

IEEE 802.3

The frame used in IEEE 802.3 Ethernet networks is shown in Figure 11.

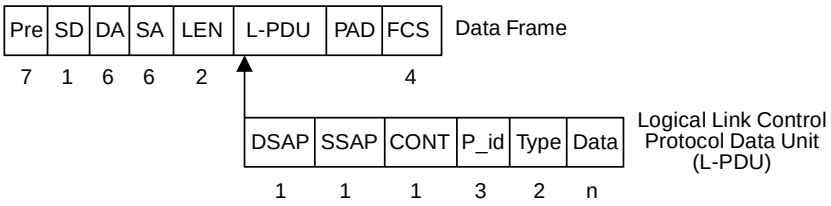


Figure 11. Format of an IEEE 802.3 Frame

The maximum size of the L-PDU for a 10Mbps network is 1500 bytes. Because 8 bytes are used within the L-PDU for the LLC header, this means that the maximum size of the data field is 1492 bytes. Therefore, the MTU for IEEE 802.3 networks should be set to 1492 bytes.

Ethernet - DIX V2

The frame used in DIX Ethernet networks is shown in Figure 12.

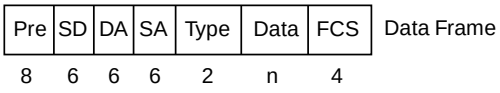


Figure 12. Format of an Ethernet V2 Frame

There is no LLC data in an Ethernet V2 frame. The maximum size for the frame is 1526 bytes. This means that the data field can be 1500 bytes maximum. The MTU for Ethernet V2 can be set to 1500 bytes.

It is possible to bridge Ethernet V2 frames to either IEEE 802.3 or IEEE 802.5 networks; a LLC header is added or removed from the frame, as required, as part of the conversion when bridging.

Sub-Network Access Protocol (SNAP)

The TCP/IP software provides protocol support down to the ISO network layer. Below this layer is the data link layer, which can be separated into two sublayers. These are the *Logical Link Control* (LLC) and the *Media Access Control* (MAC) layers.

The IEEE 802.2 standard defines the LLC sub-layer, and the MAC sub-layer is defined in IEEE 802.3, IEEE 802.4, and IEEE 802.5.

The format of an IEEE 802.2 LLC header with the SNAP header is shown in Figure 13.

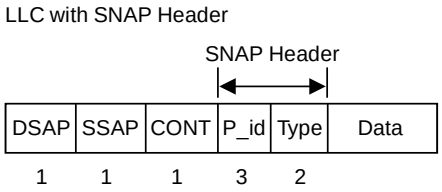


Figure 13. SNAP Header

The values of the fields in the LLC header when a SNAP header is used are specified in *RFC 1042 - Standard for Transmission of IP Datagrams over IEEE 802 Networks* The values specified are:

Field	**Value**
DSAP	X'AA'
SSAP	X'AA'
CONT	X'03' Specifies unnumbered information (UI)
P_id	X'00 00 00'
Type	X'08 00' - IP X'08 06' - ARP X'08 35' - RARP

IP Routing

IP routing is based on routing tables held within a router or internet host. These tables can either be *static* or *dynamic*. Typically, static routes are predefined within a configuration file, and dynamic routes are “learned” from the network, using a *routing* protocol.

Internet Addressing

A host on an internet is identified by its *IP address*. *Internet Protocol* (IP) is the protocol that is used to deliver datagrams between such hosts. It is assumed the reader is familiar with the TCP/IP protocols. Details of some of the protocols can be found in the *TCP/IP Tutorial and Technical Overview* (GG24-3376). Specific information relating to the Internet Protocol can be found in RFC 791.

An IP address is a 32-bit address that is usually represented in dotted decimal notation, with a decimal value representing each of the 4 octets (bytes) that make up the address. For example:

```
00001001010000110110000100000010        32-bit address
00001001 01000011 01100001 00000010     4 octets
   9        67       97       2         dotted decimal notation (9.67.97.2)
```

The IP address consists of a *network address* and a *host address*. Within the Internet, the network addresses are assigned by a central authority, the *Network Information Center* (NIC). The portion of the IP address that is used for each of these addresses is determined by the *class of address*. There are three commonly used classes of IP address (see Figure 14). (A fourth class, Class D, which is used for multicast, will not be discussed in this book.)

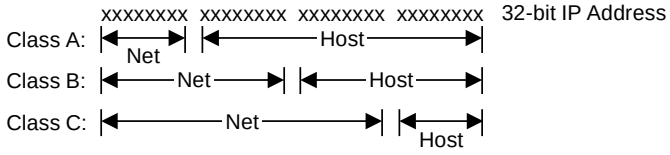


Figure 14. Classes of IP Addresses

The class of address of the IP network is determined from the first two bits in the first octet of the IP address. Figure 15 shows how the class of address is determined.

```
32-bit address          xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx

Class A                 0xxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
               min      00000000
               max      01111111
               range    1 - 126    (decimal notation; 0 and 127 are reserved)

Class B                 10xxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
               min      10000000
               max      10111111
               range    128 - 191  (decimal notation)

Class C                 110xxxxx xxxxxxxx xxxxxxxx xxxxxxxx
               min      11000000
               max      11011111
               range    192 - 223  (decimal notation)
```

Figure 15. Determining the Class of an IP Address

As shown in Figure 15, the value of the bits in the first octet determine the class of address, and the class of address determines the range of values for the network and host segment of the IP address. For example, the IP address 9.67.97.2 would

be a class A address, since the first 2 bits in the first octet contain B'00'. The network part of the IP address is “9” and the host part of the IP address is “67.97.2.”

Refer to *RFC 1166 - Internet Numbers* for more information about IP addresses. Refer to *RFC 1060 - Assigned Numbers* for more information about reserved network and host IP addresses, such as a *network broadcast address*.

Figure 16 shows a simple network with a bridge and a router.

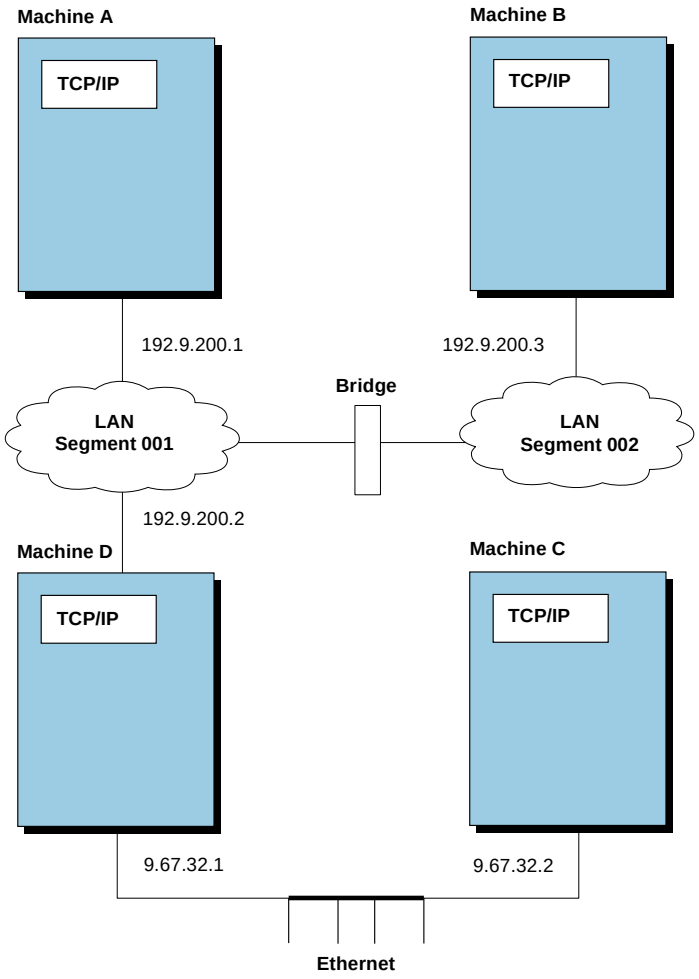


Figure 16. Routing and Bridging

Machine D is acting as an IP router and will transfer IP datagrams between the class C, 192.9.200, network and the class A, 9.67.32 network. It is important to note that for Machine B to communicate with Machine C using TCP/IP, both Machine D and the bridge have to be correctly configured and working.

TCP/IP uses the HOME statements, defined in the *hlq*.PROFILE.TCPIP data set, to assign home addresses and associated link names. HOME statements can be updated using the VARY TCPIP command. Refer to *OS/390 TCP/IP OpenEdition Configuration Guide* for more information on both the HOME statements and the VARY TCPIP command.

Direct Routing

Direct routing can take place when two hosts are directly connected to the same physical network. This can be a bridged token-ring network, a bridged Ethernet, or a bridged token-ring network and Ethernet. The distinction between direct routing and indirect routing is that with direct routing an IP datagram can be delivered to the remote host without subsequent interpretation of the IP address, by an intermediate host or router.

In Figure 16 on page 45, a datagram traveling from Machine A to Machine B would be using direct routing, although it would be traveling through a bridge.

Indirect Routing

Indirect routing takes place when the destination is ***not*** on a directly attached IP network, forcing the sender to forward the datagram to a router for delivery.

In Figure 16 on page 45, a datagram from Machine A being delivered to Machine C would be using indirect routing, with Machine D acting as the router (or gateway).

Simplified IP Datagram Routing Algorithm

To route an IP datagram on the network, the algorithm shown in Figure 17 is used.

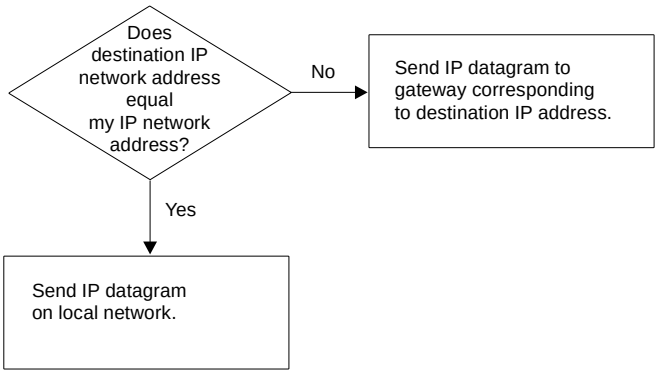


Figure 17. General IP Routing Algorithm

Using this general routing algorithm, it is very easy to determine where an IP datagram will be routed. Following is a simple example based on the configuration shown in Figure 16 on page 45.

```
Machine A   IP Address = 192.9.200.1

Routing Table

Destination     Gateway

192.9.200.1     192.9.200.1   (Machine A's network interface)

9.0.0.0         192.9.200.2   (Route to the 9.n.n.n address is
                               via Machine D, 192.9.200.2)
```

Machine A sends a datagram to host 192.9.200.3 (Machine B), using the direct route, 192.9.200.1 (its own network interface). Machine A sends a datagram to host 9.67.32.2 (Machine C), using the indirect route, 192.9.200.2 (Machine D), and Machine D then forwards the datagram to Machine C.

Subnetting

A variation of the network and host segments of an IP address, known as *subnetting*, can be used to physically and logically design a network. For example, an organization can have a single internet network address (NETID) that is known to users outside the organization, yet configure its internal network into different departmental subnets. Subnetwork addresses enhance local routing capabilities, while reducing the number of network addresses required.

To illustrate this, let us consider a simple example. Assume that we have an assigned class C network address of 192.9.200 for our site. This would mean that we could have host addresses from 192.9.200.1 to 192.9.200.254. If we did not use subnetting, then we could only implement a single IP network with 254 hosts. To split our site into two logical subnetworks, we could implement the following network scheme:

```
Without Subnetting:
                                          Network        Host Address
                                          Address        Range
       192      9        200       host
  11000000 00001001 11001000 xxxxxxxx   192.9.200      1 - 254


With Subnetting:
                                          Subnet         Host Address  Subnet
                                          Address        Range         Value
       192      9        200   64  host
  11000000 00001001 11001000 01xxxxxx   192.9.200.64   65 - 126      01


                                          Subnet         Host Address  Subnet
                                          Address        Range         Value
       192      9        200   128 host
  11000000 00001001 11001000 10xxxxxx   192.9.200.128  129 - 190     10

The subnet mask would be

       255      255      255     192
  11111111 11111111 11111111 11000000
```

OS/390 TCP/IP uses a slightly different scheme for the *subnet mask* when defining the GATEWAY statements in the *hlq*.PROFILE.TCPIP data set and for displaying the subnet mask within a onetstat -g command. The subnet mask is applied only to the host segment of the IP address, and onetstat displays the subnet mask for only the host segment of the IP address. The subnet mask in the preceding chart as defined for OS/390 TCP/IP would be:

```
        0        0        0       192       0.0.0.192
 00000000 00000000 00000000 11000000
```

Although OS/390 TCP/IP defines the subnet mask differently, the application of the subnet mask and subnet value to the IP address is consistent with RFC-architected routing algorithms. A subnet mask of 255 is used for the remainder of this section of the chapter, to retain symmetry with other routing documents that use 255 as the subnet value for the network segment of an IP address.

Because subnets B'00' and B'11' are both reserved, only two subnets are available. All 0s and all 1s have a special significance in internet addressing and should be used with care. Also notice that the total number of host addresses that we can use is reduced for the same reason. For instance, we cannot have a host address of 16 because this would mean that the subnet/host segment of the address would be B'0001000', which with the subnet mask we are using, would mean a subnet value of B'00', which is reserved.

The same is true for the host segment of the fourth octet. A fourth octet value of B'01111111' is reserved because, although the subnet of B'01' is valid, the host value of B'1' is reserved.

The network segment of the subnet mask is always assumed to be 1, so each octet has a decimal value of 255. For example, with a class B address, the first 2 octets are assumed to be 255.255.

Simplified IP Datagram Routing Algorithm with Subnets

When subnetting is used, the algorithm to find a route for an IP datagram is similar to the one for general routing, with the exception that the addresses being compared are the result of a logical AND of the subnet mask and the IP address.

For example:

```
IP address:       9.67.32.18    00001001 01000011 00100000 00010010
                                                  <AND>
Subnet Mask: 255.255.255.240    11111111 11111111 11111111 11110000

Result of
Logical AND:      9.67.32.16    00001001 01000011 00100000 00010000
```

The subnet address is 9.67.32.16, and it is this value that is used to determine the route used.

Figure 18 shows the routing algorithm used with subnets and Figure 19 on page 49 shows how a subnet route is resolved.

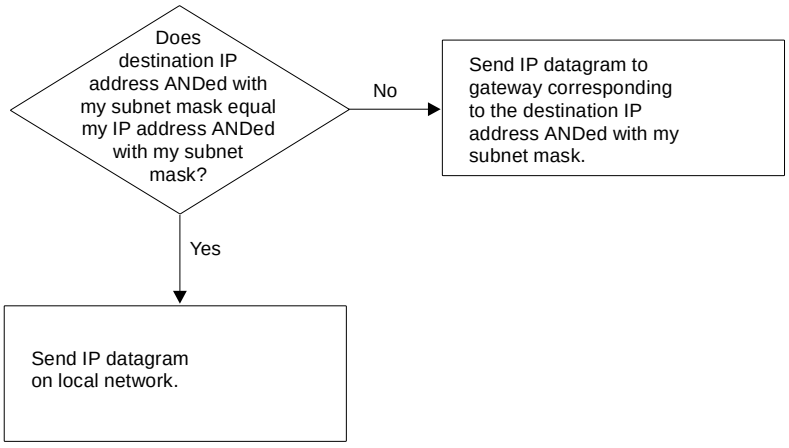


Figure 18. Routing Algorithm with Subnets

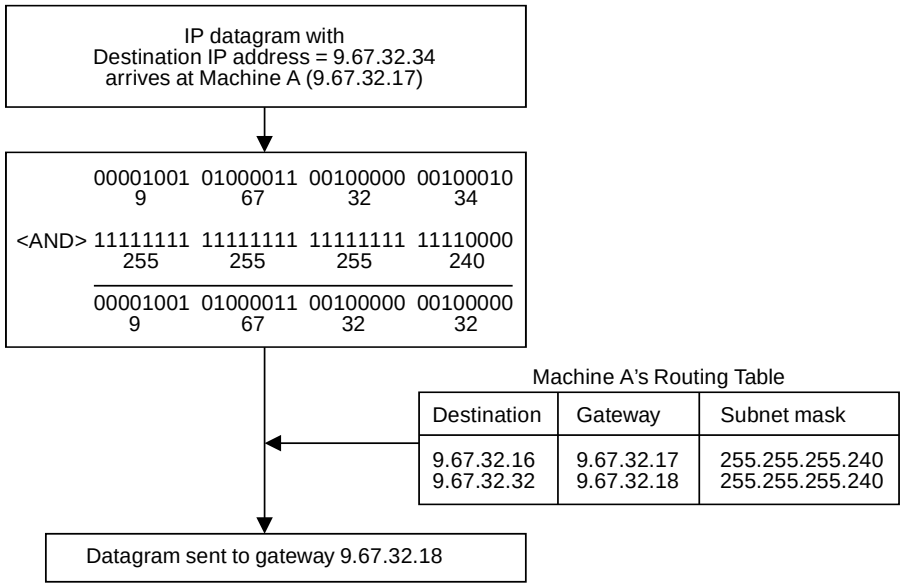


Figure 19. Example of Resolving a Subnet Route

Static Routing

Static routing, as the name implies, is defined within the local host, and must be manually changed as the network changes. Typically, a configuration file will contain the definitions for directly-attached networks, routes for specific hosts, and a possible default route that directs packets to a destination for networks that are not previously defined.

OS/390 TCP/IP uses the GATEWAY statements to configure the internal routing tables; these statements are defined in the *hlq*.PROFILE.TCPIP data set. The internal routing tables for OS/390 TCP/IP can be modified by either: (1) changing the GATEWAY statements and recycling the TCP/IP address space or (2) using the VARY TCPIP command. Refer to the *OS/390 TCP/IP OpenEdition Configuration Guide* for details about defining the GATEWAY statements and using the VARY command.

Note: When the GATEWAY statements are updated using VARY TCPIP, all previously-defined routes are discarded and replaced by the new GATEWAY definitions.

Dynamic Routing

Dynamic routing is the opposite of static routing. A TCP/IP protocol is used to dynamically update the internal routing tables when changes to the network occur. OS/390 TCP/IP uses the Routing Information Protocol (RIP) and the OE RouteD address space to monitor network changes. For more details about OE RouteD, see Chapter 12, “OE RouteD Diagnosis” on page 141 of this manual and *OS/390 TCP/IP OpenEdition Configuration Guide*.

Dynamic Routing Tables

When OS/390 TCP/IP is configured to use OE RouteD, there are two routing tables. The first routing table is managed by OE RouteD, and is updated dynamically based on the RIP protocol. OE RouteD will then update the internal routing table of the TCPIP address space. The two routing tables ***might not be identical*** for the following reasons:

- ICMP redirects are received by the TCPIP address space. TCPIP updates its internal routing table, but these changes are not propagated to OE RouteD. To prevent this situation, the parameter IGNOREREDIRECTS on the IPCONFIG statement should be coded in the *hlq*.PROFILE.TCPIP data set.
- There are GATEWAY statements in the *hlq*.PROFILE.TCPIP data set. In this situation, TCPIP will route packets based on the GATEWAY statements, and then based on the updates by OE RouteD. This is similar to a condition in UNIX** environments known as “kernel” routes.

If a network design forces the use of both static and dynamic routes, OE RouteD should be configured using the OE RouteD gateways file or data set to define the static routes. Customizing both the GATEWAY and BSDROUTINGPARMS statements should only be attempted by network programmers familiar with IP routing, RIP, and the ramifications of having distinct routing tables.

TCP/IP Traces, Component Traces, and IPCS Support

Chapter 5. TCP/IP MVS Component Traces and IPCS Support

This chapter describes how TCP/IP uses MVS Component Trace support and how to use the IPCS commands to format control block information. Refer to “MVS Component Trace Support” and “Other IPCS Support” on page 63.

Note: For OS/390 TCP/IP OpenEdition, CTRACE replaces LESSTRACE, MORETRACE, TRACE and NOTRACE commands. Those commands still work with a V3R2 stack.

MVS Component Trace Support

MVS Component Trace support is provided for use with TCP/IP stacks. Component tracing provides selected data about events that occur in the components being traced. Data indicating these events will be available in a dump taken at the point where the problem is indicated. Component Trace also allows external writers to capture trace buffers. This leads to increased diagnostic capability and a decreased likelihood of losing trace data. This trace data is intended for use in:

- Diagnosing problems in the component
- Determining how the component is running

You will typically use Component Trace while re-creating a problem. A default minimal component trace is always started during TCP/IP initialization.

Note: If you are using the TCP/IP default Component Trace SYS1.PARMLIB member, CTIEZB00, only minimum tracing is active even though the Component Trace is started at TCP/IP initialization. This will aid in first failure data capture.

Each time a new Component Trace is initiated, all previously set trace options are turned OFF. The options specified in the new Component Trace are then in effect.

Starting a Component Trace

The TCP/IP Component Trace options can be specified at TCP/IP initialization or after TCP/IP has initialized. Table 11 on page 56 lists the TCP/IP traces.

Specifying Options at Initialization

Besides specifying the desired TCP/IP traces, you can also change the TCP/IP Component Trace buffer size. The buffer size can only be changed during TCP/IP initialization. The trace options may be specified using the following methods:

- Updating the default SYS1.PARMLIB member, CTIEZB00, with the desired options. Review member CTIEZB00 in *hlq*.SEZAINST for information on TCP/IP supplied default options.
- Creating a new CTIEZBxx SYS1.PARMLIB member and specifying this member on the CTRACE() keyword in the PARM parameter of the TCP/IP started procedure's EXEC statement. For more information on the CTRACE() keyword, see “IPCS Support” on page 59, or the *OS/390 TCP/IP OpenEdition Configuration Guide*.

The following SYS1.PARMLIB member, CTIEZB00, is shipped in the SEZAINST data set.

```
/*********************************************************************/
/*                                                                   */
/*  TCP/IP for MVS                                                   */
/*  SMP/E Distribution Name: CTIEZB00                                */
/*                                                                   */
/*  MEMBER:  CTIEZB00                                                */
/*                                                                   */
/*     COPYRIGHT = Licensed Materials - Program Property of IBM.     */
/*                 This product contains "Restricted Materials of IBM"*/
/*                 5645-001 5655-HAL (C) Copyright IBM Corp. 1996.   */
/*                 All rights reserved.                              */
/*                 US Government Users Restricted Rights -           */
/*                 Use, duplication or disclosure restricted         */
/*                 by GSA ADP Schedule Contract with IBM Corp.       */
/*                 See IBM Copyright Instructions                    */
/*                                                                   */
/*  DESCRIPTION = This parmlib member causes component trace for     */
/*                 the TCP/IP product to be initialized with a       */
/*                 trace buffer size of 256K                         */
/*                 This parmlib members only lists those TRACEOPTS   */
/*                 value specific to TCP/IP.  For a complete list    */
/*                 of TRACEOPTS keywords and their values see        */
/*                 OS/390 MVS INITIALIZATION AND TUNING REFERENCE.   */
/*                                                                   */
/* $MAC(CTIEZB00) PROD(TCPIP): Component Trace SYS1.PARMLIB member  */
/*                                                                   */
/*  CHANGE-ACTIVITY =                                                */
/*  CFD List:                                                        */
/*                                                                   */
/*  $L0=D005509  HTCP330 960619 MWS: RAS DCR - Trace Enhancements   */
/*                                                                   */
/*  End CFD List:                                                    */
/*********************************************************************/
TRACEOPTS
/* ------------------------------------------------------------------ */
/*   ON OR OFF: PICK 1                                               */
/* ------------------------------------------------------------------ */
          ON

/*        OFF                                                       */
/* ------------------------------------------------------------------ */
/*   BUFSIZE: A VALUE IN RANGE 128K TO 16M                           */
/* ------------------------------------------------------------------ */
          BUFSIZE(256K)
/* ------------------------------------------------------------------ */
/*   OPTIONS: NAMES OF FUNCTIONS TO BE TRACED, OR "ALL"              */
/* ------------------------------------------------------------------ */
/*        OPTIONS(             */
/*                 'ALL     '   */
/*                ,'MINIMUM '   */
/*                ,'INIT    '   */
/*                ,'SOCKET  '   */
/*                ,'OPMSGS  '   */
/*                ,'PFS     '   */
/*                ,'ENGINE  '   */
```

```
/*              ,'RAW     '   */
/*              ,'UDP     '   */
/*              ,'TCP     '   */
/*              ,'ICMP    '   */
/*              ,'ARP     '   */
/*              ,'CTC     '   */
/*              ,'CLAW    '   */
/*              ,'LCS     '   */
/*              ,'INTERNET'   */
/*              ,'QUEUE   '   */
/*              ,'MESSAGE '   */
/*              ,'IOCTL   '   */
/*              ,'VTAMDATA'   */
/*              ,'VTAM    '   */
/*               )            */
```

Specifying Options after TCP/IP Initialization

After TCP/IP initialization, you must use the MVS 'TRACE CT' command to change the Component Trace options. All options except for the size of the TCP/IP Component Trace buffer can be changed using this command. The options may be specified via the following methods:

- Creating a new CTIEZBxx SYS1.PARMLIB member and specifying this member on the PARM= keyword of the `TRACE CT` command.
- Issuing the `TRACE CT` command without the PARM= parameter and specifying the options on the reply.

The `TRACE CT` command has the following syntax:

TRACE CT,ON,COMP=SYSTCPIP,SUB=(tcpipprocname)[,PARM=CTIEZBxx]

SUB The **SUB** keyword indicates the started procedure name of the TCP/IP for which the trace is to be run. When the 'S procname.jobname ' method of starting TCP/IP is used, the value specified for 'jobname' must be used on the SUB parameter. There can be up to 8 TCPIP sessions active in one system.

PARM The parmlib member containing trace options to be set. All options except size of trace buffers can be re-specified. This size cannot be changed during the execution of TCP/IP. If a different size is required, TCP/IP must be stopped, and then restarted using a CTIEZBxx member that specifies the different size.

If an incorrect CTIEZBxx member is specified on the TRACE CT,ON command, the following messages are issued:

```
IEE538I CTIEZBZZ MEMBER NOT FOUND IN SYS1.PARMLIB
ITT010I COMPONENT TRACE PROCESSING FAILED FOR PARMLIB MEMBER=CTIEZBZZ:
PARMLIB MEMBER NOT FOUND.
```

If an incorrect CTIEZBxx member is specified on the CTRACE() keyword of the EXEC statement of the TCP/IP started procedure, the following message is issued:

```
IEE538I CTIEZBZZ MEMBER NOT FOUND IN SYS1.PARMLIB
```

After issuing the TRACE CT command, you will be prompted to specify the trace options to be in effect, if you did not specify the PARM= keyword. You must respond using the following syntax:

```
REPLY    nn
[,ASID=(asid-list)]
[,JOBNAME=(jobname-list)]
[,OPTIONS=(name[name]...)]
[,WTR={membername|DISCONNECT}]
[,CONT|END]
```

ASID The address space identifiers (ASIDs) of the client whose TCP/IP requests are to be traced.

JOBNAME The JOBNAME of the client whose TCP/IP requests are to be traced.

OPTIONS

The valid options for use with the Component Trace facility are:

Table 11. Valid Trace Facility Options

Trace Event	Description
ALL	All types of records
ARP	Address Resolution protocol
CLAW	CLAW device
CTC	CTC device
ENGINE	Stream head management
ICMP	ICMP protocol
INIT	Initialization/Termination
INTERNET	Internet protocol layer
IOCTL	IOCTL processing
LCS	LCS device
MINIMUM	
MESSAGE	Steam message management
OPMSGS	Operator messages
PFS	Presentation Services
QUEUE	Stream queue management
RAW	RAW transport protocol
SOCKET	Sockets API
TCP	TCP transport protocol
UDP	UDP transport protocol
VTAM	VTAM Interface
VTAMDATA	VTAM Interface data

membername the member containing the source JCL that invokes the external writer. The *membername* in the WTR parameter must match the *membername* in a previous TRACE CT,WTRSTART command. (See “Obtaining Component Trace Data with an External Writer” on page 58).

WTR=DISCONNECT Disconnects the component trace external writer and the trace. You must also specify a TRACE CT,WTRSTART or TRACE CT,WTRSTOP command to start or stop the writer. (See “Connecting to an External Writer” on page 58 and “Disconnecting from an External Writer” on page 59.)

CONT or END CONT specifies that the reply continues on another line. Specify END to complete the response.

You can get detailed information about a Component Trace SYS1.PARMLIB member, the TRACE CT command, and the REPLY command in the following MVS publications:

- *OS/390 MVS Initialization and Tuning Reference*, SC28-1752-02
- *OS/390 MVS System Commands*, GC28-1781-02
- *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02

Stopping a Component Trace

The following command will turn off the current TCP/IP Component Traces:

```
TRACE CT,OFF,COMP=SYSTCPIP, SUB=(tcpipprocname)
```

TCP/IP will always maintain a minimum level of tracing to aid in first failure data capture.

Displaying Component Trace Status

Information about the status of current traces may be displayed using the MVS **DISPLAY TRACE** command. The format of the command is as follows:

```
DISPLAY TRACE,COMP=SYSTCPIP, SUB=(tcpipprocname)
```

This command displays information about the status of the TCP/IP component trace for one TCP/IP address space.

The following command displays information about the status of the TCP/IP component trace for all TCP/IP CTRACEs.

```
DISPLAY TRACE,COMP=SYSTCPIP, SUBLEVEL,N=8
```

Obtaining Component Trace Data with a Dump

You can view trace data using IPCS Component Trace formatting. The input can be either a dump of the TCP/IP address space and the dataspace containing the trace table or the data set(s) produced by the external writer.

If an abend occurs in the TCP/IP address space or in a user's address space, TCP/IP recovery will dump the home ASID, primary ASID, secondary ASID, TCPIPDS1, and the CSM dataspace.

To view the trace records for a problem where no abend has occurred, you can use the MVS DUMP command. The following example illustrates an MVS DUMP command:

```
DUMP COMM=(your dump title here)
R n,JOBNAME=tcpipprocname,DSPNAME='TCPIPPROCNAME'.TCPIPDS1,CONT
R n,SDATA=(nuc,rgn,csa,sqa),END
```

TCPIPDS1 is the name of the dataspace for each TCP/IP in an MVS image. CSA and SQA must be the defaults for the SDATA parameter.

Obtaining Component Trace Data with an External Writer

The following command starts a Component Trace external writer:

```
TRACE CT,WTRSTART=CTTCP
```

where **CTTCP** is the name of a started procedure.

Note: To make this command work, place the following procedure in SYS1.PROCLIB:

```
//CTTCP  PROC
//*CTWDASD  PROC
//* REFER:  SYS1.PROCLIB(CTWDASD)
//* COMPID: OPER
//* DOC:    THIS PROCEDURE IS THE IPCS CTRACE EXTERNAL WRITER PROCEDURE.
//*         USED BY TCP/IP   .
//*
//IEFPROC  EXEC  PGM=ITTTRCWR
//TRCOUT01 DD DSNAME=MEGA.TCPIP33.CTRACE,UNIT=SYSDA,                  X
//             VOL=SER=STORGE,
//             SPACE=(4096,(100,10)),DISP=(NEW,CATLG),DSORG=PS
```

For information on inserting this procedure in SYS1.PROCLIB, see *OS/390 TCP/IP OpenEdition Configuration Guide*.

Connecting to an External Writer

The following commands connect a TCP/IP stack to the external writer.

```
TRACE CT,ON,COMP=SYSTCPIP,SUB=(procedure_name)
R xx,WTR=CTTCP,END
```

In this example, **CTTCP** is the name of the procedure used to start the writer.

Disconnecting from an External Writer

The following commands disconnect a TCP/IP stack from the external writer.

```
TRACE CT,ON,COMP=SYSTCPIP,SUB=(procedure_name)

R xx,WTR=DISCONNECT,END
```

Stopping an External Writer

The following command stops a Component Trace external writer

```
TRACE CT,WTRSTOP=CTTCP
```

where **CTTCP** is the name of the procedure used to start the writer.

IPCS Support

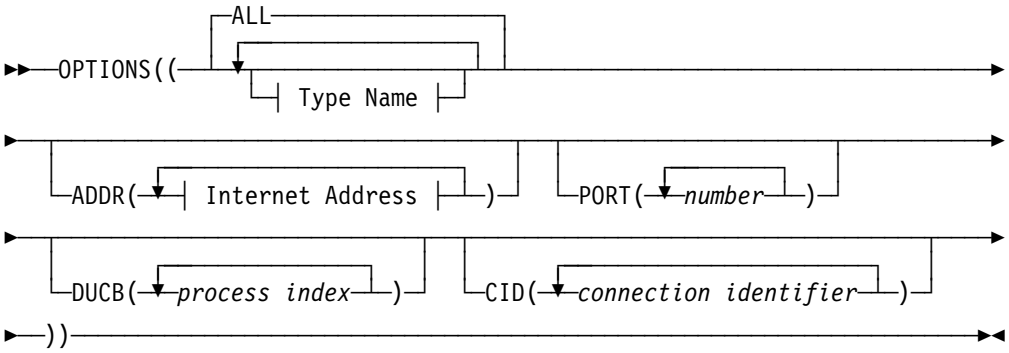
CTRACE Options for TCP/IP

Interactive Problem Control System (IPCS) is a standard system component that lets you format TCP/IP Component Trace records from a dump or from CTRACE external writer trace files.

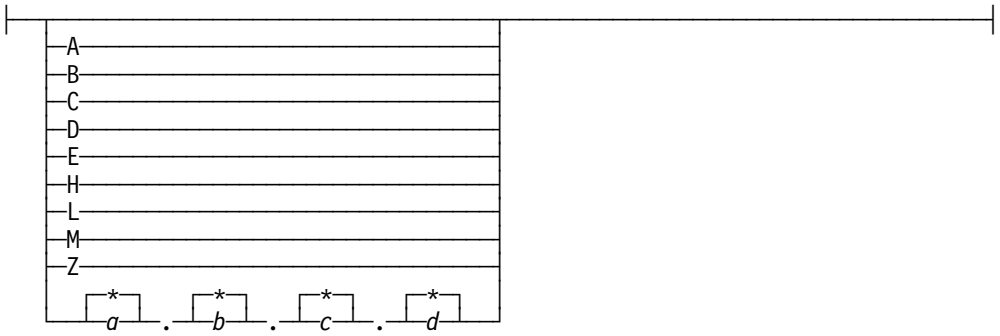
The code for the CTRACE record formatter can be found in the *hlq*.SEZBMIG data set. This data set should be added as a concatenation to the STEPLIB data set.

This section covers the CTRACE options that are unique to TCP/IP. For more information about other CTRACE options, see *OS/390 MVS IPCS Commands*, GC28-1754-02 .

CTRACE has the following format:



Internet Address:



Where:

ALL
All types of records are to be formatted. This is the default.

Type Name
Is the name of a trace type. Only records of these types will be formatted. For a list of types, refer to Table 11 on page 56.

Internet Address
Is a keyword for a predefined Internet address or an intranet address. Predefined internet addresses are:

A	A class **A** internet address
B	A class **B** internet address
C	A class **C** internet address
D	A class **D** internet address
E	A class **E** internet address
H	Is a local host address, 0.0.*.*
L	Is the loop back address, 127.*.*.*
M	Is the local net broadcast, 255.255.255.255
Z	Is a zero internet address, 0.0.0.0

An internet address is specified using the dotted decimal form. Where:

a	The first byte is a value from 0 to 255.
b	The second byte is a value from 0 to 255.
c	The third byte is a value from 0 to 255.
d	The fourth byte is a value from 0 to 255.
*****	indicates that the byte is not to be checked. Any value for the byte is acceptable.

Port Address
A port address from 0 to 65535. The address may be specified as a decimal or hexadecimal integer.

CTRACE Examples

Figure 20 shows an example of CTRACE output:

```
SYSNAME   MNEMONIC  ENTRY ID    TIME STAMP     DESCRIPTION
-------   --------  --------  --------------  -------------
SY1       ENGINE    50010002  12:42:20.182756  STREAMOP Request
HASID..001C        PASID...001C        SASID..001C       MODID..EZBSKSTO
TCB....009E29A8    REG14...89BD7392    USER...TCPIP33    DUCB...00000401   CID..00000000
  ADDR...00000000  0A052058  LEN....00000004  Stream Descriptor
    +0000  00000000                                                          | ....                            |
  ADDR...00000000  09B33700  LEN....00000020  Execution Block
    +0000  00000001  00000001  00000000  00000000  00000000  00000000  00000000  00000000  | ................................ |
  ADDR...00000000  09B33720  LEN....00000010  Operation Descriptor
    +0000  00000000  00000000  00000000  00000000                            | ................                |
  ADDR...00000000  09B33EC0  LEN....00000080  Operation Block
    +0000  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  | ................................ |
    +0020  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  | ................................ |
    +0040  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  | ................................ |
    +0060  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  FEFEFEFE  | ................................ |
SY1       ENGINE    50010101  12:42:20.182967  Enable Stream Head Access
HASID..001C        PASID...001C        SASID..001C       MODID..EZBSKSAC
TCB....009E29A8    REG14...89BD55C0    USER...TCPIP33    DUCB...00000401   CID..00000000
  ADDR...00000000  09E08098  LEN....00000188  Stream Access Control Block
    +0000  E2D2E2C3  00000000  00000000  00000000  E2D2C1E3  00000000  00000000  00000000  | SKSC............SKAT............ |
    +0020  00000000  00000000  00000000  00030100  E2E3D9C5  C1D4C1C3  E2A39985  81944040  | ................STREAMACStream   |
    +0040  E2D2C1E3  00000000  00000000  00000000  00000000  00000000  00000000  00030200  | SKAT............................ |
    +0060  E2E3D9C5  C1D4C1C3  E2A39985  81944040  E2D2C1E3  00000000  00000000  00000000  | STREAMACStream  SKAT............ |
    +0080  00000000  00000000  00000000  00030300  E2E3D9C5  C1D4C1C3  E2A39985  81944040  | ................STREAMACStream   |
    +00A0  E2D2C1E3  00000000  00000000  00000000  00000000  00000000  00000000  00030400  | SKAT............................ |
    +00C0  E2E3D9C5  C1D4C1C3  E2A39985  81944040  E2D2C1E3  00000000  00000000  00000000  | STREAMACStream  SKAT............ |
    +00E0  00000000  00000000  00000000  00030500  E2E3D9C5  C1D4C1C3  E2A39985  81944040  | ................STREAMACStream   |
    +0100  00000005  E2E3C1D9  E340C9D7  40D5C1D4  40C4D9C9  E5C5D940  40404040  40404040  | ....START IP NAM DRIVER         |
    +0120  09E08098  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | .\.q........................... |
    +0140  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ................................ |
    +0160  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ................................ |
    +0180  00000000  00000000                                                | ........                        |
```

Figure 20. Sample CTRACE record

The major fields in the output are explained below:

SY1
The system identifier

ENGINE
The TYPE of trace record.

50010002
The hexadecimal value of the format code.

12:42:20.182756
The time the record was written

STREAMOP Request
The value of the DESCRIPTION of the trace record.

HASID
The home ASID (from the creation of the dispatchable unit)

PASID
The primary ASID (from EPAR instruction)

SASID
The secondary ASID (from ESAR instruction)

MODID
The module name

TCB
The address of the TCB (from the creation of the dispatchable unit)

REG14
The contents of register 14 when the trace record was created.

USER
The job name (from the creation of the dispatchable unit)

DUCB
The Dispatchable Unit index

CID
Is a communication or connection ID. It is a 32-bit binary integer used to correlate trace records.

The next four fields are present only when FILTER data is present.

IADRL
The WLOIADRL field, which establishes the length of the Socket address structure.

IADRF
The WLOIADRF field, which establishes the address family.

IPORT
The WLOIPORT field, which establishes the port address.

IIADR
The WLOIIADR field, which establishes the internet address. This field is shown in hex and in dotted decimal formats.

The trace data element is dumped showing the address, length, and a description of the dumped data.

ADDR
The access register and the address of the data are being traced.

LEN
The number of bytes of data recorded in the trace buffer.

Stream Descriptor
The description of the data area.

+0000
The hexadecimal offset of data.

00000001 00000001 00000000
The trace data formatted in hexadecimal words.

| . . . |
The trace data formatted in EBCDIC. Non printable characters are translated to periods.

If the external writer defined multiple output data sets, then all of these data sets must be used as input to the CTRACE command. The COPYTRC or MERGE

IPCS commands can be used to make all the trace data appear as one single file, in GTF Time sequence.

For more information on the IPCS CTRACE command and IPCS, see *OS/390 MVS IPCS User's Guide*, GC28-1756-02.

Following is an example of CTRACE using the PFS option:

```
SYSNAME   MNEMONIC  ENTRY ID    TIME STAMP     DESCRIPTION
 -------   --------  --------  ---------------  -------------
 SY1       PFS       6001000F  12:42:23.515659  Socket IOCTL Entry
 HASID..001C        PASID...0042        SASID..001C       MODID..EZBPFIOC
 TCB....009E29A8    REG14...8A7D0B14    USER...TCPIP33    DUCB...00000405   CID..00000000
   ADDR...00000000  0A61ED54  LEN....00000040  Cookie Jar
     +0000  C3D1C1D9  00000040  00000000  0A400800  00000000  00000000  00000000  7F5A1CD0  | CJAR... ..... ....."!.} |
     +0020  01010024  00001000  01010024  000015B0  01010024  00210000  00000000  00000000  | ............................... |
   ADDR...00000000  7F5A1CD0  LEN....00000190  Socket Control Block
     +0000  E2C3C2C4  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | SCBD........................... |
     +0020  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0040  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0060  00000000  00000000  01010024  00210000  00000000  00000000  00000000  00000000  | ............................... |
     +0080  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +00A0  00000000  00000000  00000000  00000000  00000004  00000002  FFFFFFFF  00000000  | ............................... |
     +00C0  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +00E0  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0100  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0120  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0140  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0160  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0180  00000000  00000000  00000000  00000000                                            | ................               |
 SY1       PFS       60010011  12:42:23.515695  Socket IOCTL Part2 Entry
 HASID..001C        PASID...0042        SASID..001C       MODID..EZBPFIO2
 TCB....009E29A8    REG14...8A7D20AA    USER...TCPIP33    DUCB...00000405   CID..00000000
   ADDR...00000000  0A61ED54  LEN....00000040  Cookie Jar
     +0000  C3D1C1D9  00000040  00000000  0A400800  00000000  00000000  00000000  7F5A1CD0  | CJAR... ..... ....."!.} |
     +0020  01010024  00001000  01010024  000015B0  01010024  00210000  00000000  00000000  | ............................... |
   ADDR...00000000  7F5A1CD0  LEN....00000190  Socket Control Block
     +0000  E2C3C2C4  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | SCBD........................... |
     +0020  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0040  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0060  00000000  00000000  01010024  00210000  00000000  00000000  00000000  00000000  | ............................... |
     +0080  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +00A0  00000000  00000000  00000000  00000000  00000004  00000002  FFFFFFFF  00000000  | ............................... |
     +00C0  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +00E0  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0100  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0120  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0140  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0160  00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000  | ............................... |
     +0180  00000000  00000000  00000000  00000000                                            | ................               |
```

Figure 21. Example of CTRACE with PFS option

Other IPCS Support

If you need to locate a problem in your TCP/IP network, try using the IPCS commands. You can format, search, list, and analyze TCP/IP control blocks using one of the following types of IPCS support:

- CBFORMAT command
- CBSTAT command

You can enter the IPCS commands either under the IPCS COMMAND function or, on the command line of any IPCS screen by specifying the identifier 'IP' before the command.

For more information on IPCS commands, see *OS/390 MVS IPCS User's Guide*, GC28-1756-02 .

Using the CBFORMAT Command

TCP/IP provides several control block formatter exit routines for you to use with the IPCS CBFORMAT command.

You can format the following control blocks using the CBFORMAT command syntax in column one. Column two identifies the control block that will be formatted.

CBFORMAT Command format	Control block name	Description
CBF addr. STR(EZATASB)	TASB	TCP/IP Address Space related information
CBF addr. STR(EZATCA)	TCA	Trace control area
CBF addr. STR(EZATSAB)	TSAB	TCP/IP Server Anchor block
CBF addr. STR(EZATSDB)	TSDB	TCP/IP Server Data block
CBF addr. STR(EZATSDX)	TSDX	TCP/IP Server Data block extension
CBF addr. STR(EZATSEB)	TSEB	TCP/IP Server Anchor block entry

Using the CBSTAT Command

The TCP/IP CBSTAT exit routine displays messages regarding an application's TCP/IP connections for a particular TCB. Table 12 shows the CBSTAT command format.

Table 12. Example of CBSTAT command

CBSTAT Command format	Control block name	Description
CBSTAT addr. STR(TCB)	TCB	MVS Task control block

Generating an IP Packet Trace

Chapter 6. Generating an IP Packet Trace

TCP/IP Packet Trace is a diagnostic method for obtaining GTF traces of IP packets flowing from and into a TCP/IP on an OS/390 host. The PKTTRACE statement lets you copy IP packets as they enter or leave TCP/IP, and then examine the contents of the copied packets. To be traced, an IP packet must meet all the conditions specified on the PKTTRACE statement.

While the CTRACE function collects event data about TCP/IP's internal processing, PKTTRACE collects data records that flow over the links.

Features

There are several key features in the IP packet tracing facility. The most significant feature is two-stage processing, which consists of an initial data capture stage followed by a data formatting and reporting stage. The aim of this two-stage process is to minimize the impact of tracing on the performance of the host system and on the processing of network traffic. This is achieved by passing the traced packets to the Generalized Trace Facility (GTF), which handles their storage. Furthermore, the two-stage process allows the captured data to be studied at length and formatted in several different ways.

Another key feature of the facility is the capability to specify selection criteria to identify the IP packets that are to be traced. The selection criteria are available at both data capture and data reporting time. For example, the selection criteria can be used at capture time to confine tracing to traffic from a particular host. Further selection criteria can be applied to the captured data at report generation time to examine, for example, packets within a particular time span.

The Trace Process

The trace data is collected as IP packets enter or leave TCP/IP. The actual collection occurs within the device drivers of TCP/IP, which capture the data that has just been received from or sent to the network.

Note: Packets that are captured have extra information added to them before they are stored. This extra information is used during the formatting of the packets.

The captured data reflects exactly what the network sees: for example, the trace contains the constituent packets of a fragmented packet exactly as they are received or sent.

The selection criteria for choosing packets to trace are specified through the PKTTRACE statement for the TCPIP address space. Refer to *OS/390 TCP/IP OpenEdition Configuration Guide* for more information about the PKTTRACE statement and subcommand.

The PKTTRACE statement and subcommand are applied to device links that are defined in the TCPIP address space through the LINK statement. Figure 22 on page 68 illustrates the overall control and data flow in the IP packet tracing facility.

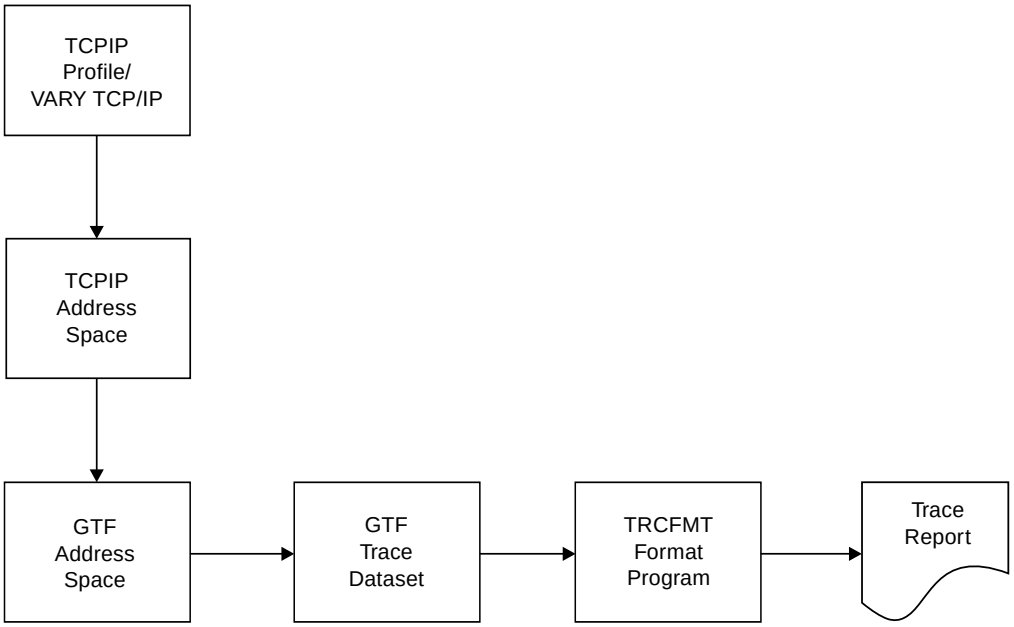


Figure 22. Control and Data Flow in the IP Packet Tracing Facility

Supported Devices

IP packet tracing is supported in the following device interfaces:

- LAN Channel Station: IBM 8232 LAN channel station, IBM 3172 Interconnect Controller
  - Ethernet and 802.3 protocols
  - Token-ring
  - FDDI
- Channel-to-Channel
- RS/6000* Parallel Channel Attachment (CLAW) or ESCON (CLAW)

Tracing IP Packets

Before you can start packet tracing, the following conditions must have be met:

1. GTF must be active, with TRACE=USR. You need to trace 5E4 type records. See *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 for details about starting GTF.
2. Packet tracing must be enabled.
   - For devices that are controlled by the TCP/IP address space, enable packet tracing by using

     `VARY tcpip,Tcpipprocname,CMD=OBEYFILE,DSN=datasetname`

     The OBEYFILE data set contains a PKTTRACE command to either set or CLEAR packet tracing. For more information on using the VARY TCPIP command, refer to *OS/390 TCP/IP OpenEdition Configuration Guide*.
   - Packet tracing can also be set in *hlq*.PROFILE.TCPIP.

Notes:

a. The appropriate tracing criteria must be specified using the PKTTRACE statement for TCPIP.

b. If GTF is not active or is stopped while the TCP/IP packet trace facility is active, then packet tracing will continue (where in previous releases it was stopped). When GTF is started again, packet trace will write out records. Records will be lost when GTF is not active. No messages will be written to show this change of state.

1. **Start Packet Tracing**

   You select what you want to trace using the PKTTRACE statement. This statement has two parts. The first part defines to TCP/IP the links that are to be traced and characteristics of how they are to be traced. The second part turns packet tracing ON. For example:

   ```
   PKTTRACE PROT=UDP IP=127.0.0.1
   PKTTRACE ON
   ```

   This specifies that UDP protocols will be traced for the given IP address.

   The PKTTRACE statement can be included in the TCPIP profile, or it can be added using the VARY TCPIP command. For the complete syntax of the PKTRACE statement, refer to *OS/390 TCP/IP OpenEdition Configuration Guide* or *OS/390 MVS Diagnosis: Tools and Service Aids*.

2. **Generate the Trace Data**

   You can generate trace data using OE applications such as FTP or Telnet, or by using test utilities such as oping. This step usually involves reproducing a network problem while the IP packet trace facility is recording packets passing through the network.

3. **Terminate Packet Tracing**

   For devices controlled by the TCP/IP address space, you can disable packet tracing by processing the PKTTRACE CLEAR statement. This is done through the VARY TCPIP command using the OBEYFILE data set (see *OS/390 TCP/IP OpenEdition Configuration Guide*). If packet tracing is enabled in the *hlq*.PROFILE.TCPIP data set and is not required the next time TCPIP is started, this data set should also be modified.

   Note:

   If GTF is stopped or disabled, packet tracing will not be stopped. There will be no message, and packet tracing will continue when GTF is again available.

Figure 23 on page 70 is an example of a packet trace.

```
  PKT  0000001 DATE=96/05/13 TIME=12:19:40.604325

       FROM LINK=TR1              DEV=LCS_TOKEN_RING
IP     SRC=9.24.104.79        DST=9.24.104.126   1
       VER=4  HDLEN=5  TOS=X'00' TOTLEN=44    ID=316   FLAGS=B'000'
       FRAGOFF=0    TTL=64   PROTOCOL=TCP         CHECKSUM=X'9693'
TCP    SRC=1024        DST=TELNET   2    SEQ=1253260801 ACK=0          HDLEN=6
       WINDOW=28672 CHECKSUM=X'B85D' URGPTR=0     SYN
       OPTION=MAX_SEG_SIZE SIZE=1460

PKT    0000002 DATE=96/05/13 TIME=12:19:40.609589
       TO   LINK=TR1              DEV=LCS_TOKEN_RING
IP     SRC=9.24.104.126       DST=9.24.104.79        VIA=9.24.104.79
       VER=4  HDLEN=5  TOS=X'00' TOTLEN=44    ID=22551 FLAGS=B'000'
       FRAGOFF=0    TTL=60   PROTOCOL=TCP         CHECKSUM=X'43B8'
TCP    SRC=TELNET      DST=1024        SEQ=878223024  ACK=1253260802 HDLEN=6
        WINDOW=28672 CHECKSUM=X'E34F' URGPTR=0     ACK SYN
        OPTION=MAX_SEG_SIZE SIZE=1960

 PKT   0000003 DATE=96/05/13 TIME=12:19:40.614593
       FROM LINK=TR1              DEV=LCS_TOKEN_RING
 IP    SRC=9.24.104.79        DST=9.24.104.126
       VER=4  HDLEN=5  TOS=X'00' TOTLEN=40    ID=317   FLAGS=B'000'
       FRAGOFF=0    TTL=64   PROTOCOL=TCP         CHECKSUM=X'9696'
 TCP   SRC=1024        DST=TELNET      SEQ=1253260802 ACK=878223025  HDLEN=5
       WINDOW=28672 CHECKSUM=X'FD00' URGPTR=0     ACK
```

Figure 23. Example of Trace Produced with PKTTRACE

1 This an IP packet coming in over the TR1 link from the IP host with IP address 9.24.104.79.

2 It is a TCP segment coming from an ephemeral port number and going to the telnet server port number (23) on the 9.24.104.126 host, which is the OS/390 TCP/IP OpenEdition stack.

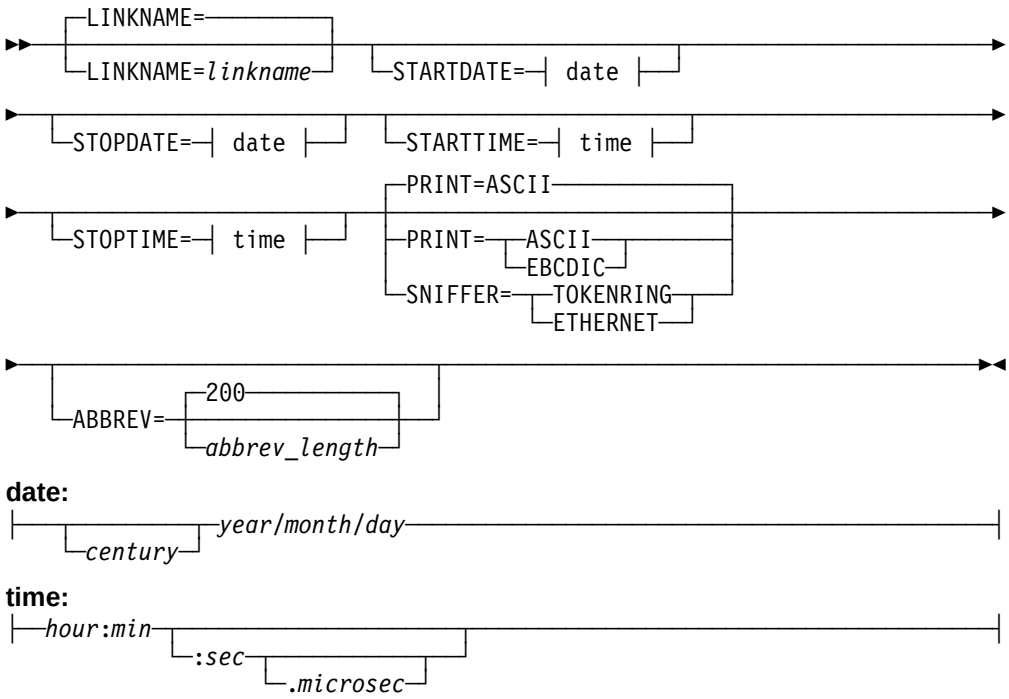
Figure 23 shows the packet sequence that is the standard TCP “three-way handshake” connection setup (SYN, ACK+SYN, ACK). In this situation the TCP connection is between a telnet client and the OpenEdition TelnetD server.

Formatting a Trace Report using the TRCFMT Utility

Trace output can be generated by the TRCFMT utility, which uses the GTF trace data set as input.

The TRCFMT utility has several options that allow the user to select the packets to be formatted and to specify how the trace output is to be formatted. The IP packets that are formatted must meet all the conditions specified by the options passed to the utility. If no options are specified, all packet information stored in the GTF trace data set is formatted.

The syntax of the options for the TRCFMT utility is:



LINKNAME Specifies the name of a link. The actual name specified depends on which device traced the packets. For devices in the TCPIP address space, specify the link name defined on the LINK statement.

STARTDATE

Specifies the start date for the trace report. Only packets that were captured on or after this date will appear in the report. The start date refers to the local date of the host where the packets were traced. If this option is omitted, formatting will start with the earliest packet traced.

The subfields of the value associated with this option are described below. Each of the subfields has a fixed length and must be supplied with leading zeros.

century	Specifies the century as a 2-digit number.
year	Specifies the year as a 2-digit number.
month	Specifies the month as a 2-digit number.
day	Specifies the day as a 2-digit number.

STOPDATE Specifies the stop date for the trace report. Only packets that were captured before or on this date will appear in the report. The stop date refers to the local date of the host where the packets were traced. If this option is omitted, formatting will stop with the last packet traced.

The subfields of the value associated with this option are described above in the description of the STARTDATE option.

STARTTIME

Specifies the start time for the trace report. Only packets that were captured on or after this time will appear in the report. The start time refers to the local time of the host where the packets were traced. If

this option is omitted, formatting will start with the earliest time, that is 00:00:00.000000.

The subfields of the value associated with this option are described below. Each of the subfields has a fixed length and must be supplied with leading zeros.

hour Specifies the hour as a 2-digit number.
min Specifies the minutes as a 2-digit number.
sec Specifies the seconds as a 2-digit number.
microsec Specifies the microseconds as a 6-digit number.

STOPTIME Specifies the stop time for the trace report. Only packets that were captured before or at this time will appear in the report. The stop time refers to the local time of the host where the packets were traced. If this parameter is omitted, formatting will stop with the latest time, that is 23:59.59.999999.

The subfields of the value associated with this option are described above in the description of the STARTTIME option.

PRINT Specifies that the trace records are to be formatted as a text report for display on a terminal or printer. This option is the default.

ASCII

Specifies that the uninterpreted fields in the IP packet are assumed to contain ASCII data. The data in these fields is therefore translated to displayable EBCDIC characters according to the following table before it is presented in the character equivalent dump. The ASCII keyword is the default.

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
1_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
2_																
3_	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4_	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5_	P	Q	R	S	T	U	V	W	X	Y	Z	.	\	.	.	_
6_	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7_	p	q	r	s	t	u	v	w	x	y	z	{	|	}	.	.
8_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
9_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
A_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
B_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
C_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
D_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
E_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
F_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

EBCDIC

If this keyword is specified, the uninterpreted fields in the IP packet are assumed to contain EBCDIC data. The data in these fields is translated, according to the following table, only to remove nondisplayable EBCDIC characters before it is presented in the character equivalent dump.

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
1_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
2_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
3_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
4_		.	.	.	.	.	.	.	.	.	¢	.	<	(	+	|
5_																
6_	-	/	.	.	.	.	.	.	.	.	¦	,	%	_	>	?
7_	.	.	.	.	.	.	.	.	.	`	:	#	@	'	=	"
8_	.	a	b	c	d	e	f	g	h	i	.	.	.	.	.	.
9_	.	j	k	l	m	n	o	p	q	r	.	.	.	.	.	.
A_	.	.	s	t	u	v	w	x	y	z	.	.	.	.	.	.
B_	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
C_	{	A	B	C	D	E	F	G	H	I	.	.	.	.	.	.
D_	}	J	K	L	M	N	O	P	Q	R	.	.	.	.	.	.
E_	\	.	S	T	U	V	W	X	Y	Z	.	.	.	.	.	.
F_	0	1	2	3	4	5	6	7	8	9	.	.	.	.	.	.

SNIFFER Formats the trace records so that they are suitable for downloading to a DatagLANce* Network Analyzer or a Sniffer** Network Analyzer for analysis.

TOKENRING
If this keyword is specified, the output is formatted for the token-ring analysis application of the analyzer. This keyword specifies the file format only and does not imply that only packets traced on a token ring will be displayed. Packets from all devices can be displayed using this option.

ETHERNET
If this keyword is specified, the output is formatted for the Ethernet analysis application of the analyzer. This keyword specifies the file format only and does not imply that only packets traced on an Ethernet will be displayed. Packets from all devices can be displayed using this option.

ABBREV The amount of user data in the trace record is abbreviated if this keyword is specified. The default length is 200 bytes.

Note: The data may have been abbreviated when the trace data was written. See the PKTTRACE statement in *OS/390 TCP/IP OpenEdition Configuration Guide* for more information.

As shown in the syntax diagram on page 71, the date and time selection criteria can be specified separately. The date and time options are tested independently of each other. Thus, if a date specification is given without a time specification, only the date when the packet was traced is examined to see if it meets the criteria. Similarly, if a time specification is given without a date specification, only the time when the packet was traced is examined to see if it meets criteria. This will, for example, allow the same time slot to be examined over more than one day.

The TRCFMT program uses three optional ddnames to specify the input, output, and error data sets during processing. If data sets other than the default are desired, the ddnames must be assigned to the required data sets when invoking the TRCFMT utility.

The ddnames are:

DDNAME	Description
FMTIN	Identifies the input GTF trace data set. If this ddname is omitted, the default data set is SYS1.TRACE.
FMTOUT	Identifies the destination for the trace report for the PRINT option or the destination of the analyzer records for the SNIFFER option. If this ddname is omitted, the default data set is TCPIP.TRACE.OUTPUT. The data set associated with this ddname must have the attributes of variable-length records and logical record length of 137 bytes.
FMTERR	Identifies the destination for error messages produced when creating a DatagLANce Network Analyzer or a Sniffer Network Analyzer output file in FMTOUT. Reporting of inconsistencies in an IP packet is an integral part of the trace report generated by the PRINT option, but these cannot be sent to FMTOUT when it is to contain data in a format to be read by a DatagLANce Network Analyzer or a Sniffer Network Analyzer. If this ddname is omitted, the default data set is TCPIP.TRACE.ERROR. This ddname is only required for the SNIFFER option. The data set associated with this ddname must have the attributes of variable-length records and logical record length of 137 bytes.

The TRCFMT utility can be invoked in either the TSO environment or as a batch job using JCL statements.

TRCFMT in the TSO Environment

The following example demonstrates the steps required to use TRCFMT in the TSO environment.

```
System:     READY
User:       allocate dd(fmtin) ds('sys1.trace')
System:     READY
User:       allocate dd(fmtout) ds(output.trc)
System:     READY :user.
User:       trcfmt linkname=tr1 sniffer=ethernet starttime=08:30 stoptime=17:30
System:     EZA2070W OPENFILE: DDname FMTERR not defined by user, using default
System:     EZA2072I OPENDD:   FMTERR DDname has been defined to TCPIP.TRACE.ERROR
System:     READY
```

Figure 24. Invoking TRCFMT from TSO

In the example shown in Figure 24, TRCFMT reads input from the GTF trace data set SYS1.TRACE, the output is written to the data set *userid*.OUTPUT.TRC, and error messages are written to the default data set TCPIP.TRACE.ERROR.

TRCFMT As a Batch Job

The following example illustrates a set of JCL statements to invoke TRCFMT via a batch job.

```
//JOB#1   JOB 'JOB1','TRCFMT BATCH',MSGLEVEL=(1,1),MSGCLASS=X,REGION=4096K,
//    CLASS=A
//*--------------------------------------------------------------------
//*
//*  IKJEFT01 - RUN A TSO COMMAND IN BATCH
//*
//*--------------------------------------------------------------------
//IKJEDF01 EXEC PGM=IKJEFT01
//FMTIN    DD DSN=SYS1.TRACE,DISP=SHR
//FMTOUT   DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
TRCFMT STARTTIME=09:00 PRINT=EBCDIC
/*
//
```

Figure 25. Invoking TRCFMT in a Batch Job

In the example shown in Figure 25, as with the previous example, TRCFMT reads its input from the data set SYS1.TRACE. Its output and error messages are written to system-managed spool files.

Generating a File for the DatagLANce Network Analyzer or the Sniffer Network Analyzer

The TRCFMT utility supports the generation of a file in a format suitable for downloading to a DatagLANce Network Analyzer or a Sniffer Network Analyzer. This feature is selected with the SNIFFER option, which requires specification of either the ETHERNET or TOKENRING suboption to indicate the format of the file to be generated for the analyzer. Refer to “Formatting a Trace Report using the TRCFMT Utility” on page 70 for information about these suboptions. Once the file is generated, it is downloaded as a binary file to the analyzer and loaded using the standard features of the analyzer. If you are using the Ethernet analyzer application, the DOS file type must be ENC. If you are using the token-ring analyzer application, the DOS file type must be TRC.

OE File Transfer Protocol (OE FTP) Diagnosis

Chapter 7. OE File Transfer Protocol (OE FTP) Diagnosis

Overview

The OE FTP server has a multi-process design. The main ftp daemon performs initialization and configuration functions, then waits for incoming client connections. When an incoming client connection is received, a new process is forked, and a second load module, which provides the actual session support for the client connection, is loaded into the new address space .

The socket connection information and configurable variables are passed to the new address space so the variables in the client session program will have the same value of the variables in the main daemon process at the time the new client connection is received. (In other words, the client session program has the *current* settings of all variables, not the initial settings).

After the new client process has been created , the main daemon process and all client processes are totally independent from one another. Any changes to the variables in the main daemon process, or in other client processes, are not reflected in any other process.

After the new client process is created, the client is prompted for a user ID and password. The C Runtime Library function *seteuid()* is used to set the effective user ID for the new address space to the client's user ID.

Definitions and Setup

Start Procedure

The start procedure for the FTP server is EZAFTPAP in the *hlq*.SEZAINST data set. Changes might be necessary to customize the start procedure for your MVS host system. The following should be kept in mind for the FTP server start procedure.

- The STEPLIB DD statement points to the library containing FTPD and FTPDSRV. This library must be APF authorized.
- The C runtime libraries are needed for FTPD and FTPDSRV. They must be APF authorized. If the C runtime library is not in the MVS link list, it must be included on the STEPLIB DD statement.
- If the FTP server will be used for SQL queries, the DB2 DSNLOAD library must be APF authorized and must be either in the MVS link list or included on the STEPLIB DD statement.
- Several start options are available for the FTP server. If specified in the start procedure, these values will override the default values for the FTP server and any values specified in the FTP.DATA data set.
- Refer to the *OS/390 TCP/IP OpenEdition Configuration Guide* for more information about the FTP server start procedure.

FTP.DATA Data Set

The FTP.DATA data set is an optional data set that allows the FTP server configuration parameters to be customized. Refer to the *OS/390 TCP/IP OpenEdition Configuration Guide* for more information about the FTP.DATA data set.

TCPIP.DATA Data Set

The TCPIP.DATA data set provides information to the FTP server such as the high-level qualifier to be used for configuration data sets, whether messages are to be written in upper or mixed case, and which DBCS translation tables are to be used. Refer to the *OS/390 TCP/IP OpenEdition Configuration Guide* for more information about the TCPIP.DATA data set.

Error Exit Codes

OE FTP uses the following error exit codes:

12 Daemon initialization failed; unable to accept an incoming connection. An EZY message identifying the specifice problem is sent to syslogd.

24 The client sessions initialization terminated because the FTP server load module cannot be loaded/executied. Message EZYFT53E is sent to syslogd.

Name Considerations for OE FTP

MVS Naming Conventions

MVS data set names used with all FTP commands sent to the OE FTP server must meet MVS data set naming conventions as follows:

1. Data set names can be no longer than 44 characters.

   If the *pathname* parameter sent with an FTP command is not enclosed in single quotation marks, then the path name is appended to the current working directory to create the data set name. The combination of current working directory and the path name cannot be longer than 44 characters. The current working directory can be displayed by issuing the `PWD` command.

2. Each qualifier in a data set name must conform to the following:

   a. It must be no longer than eight characters

   b. It must begin with a letter or the special characters $, @, or #.

   c. It may contain only numbers, letters, or the special characters $, @, #, -, or }

3. The rules described in 2 also apply to member names for a member of a partitioned data set.

4. Generation data group data set names must be in the format *gdg_name(generation_level)*. *generation_level* is either 0, +*nn*, or -*nn*, where *nn* is the generation number. For example, the GDG data set MYGDG could be specified as `MYGDG(0)` for the current generation level, `MYGDG(-1)` for the next to the latest generation level, or `MYGDG(+1)` for the new generation level.

HFS Naming Conventions

Here are some naming conventions you should know about when using HFS files with the OE FTP server:

1. The HFS name is case-sensitive.
2. If a name begins with a single quote, specify `QUOTESOVERRIDE FALSE` in FTP.DATA, or use the `SITE NOQUOTESOVERRIDE` command.
3. Names may contain imbedded blanks for special characters. Be aware that
   - The subcommand is interpreted as:

     `<ftp_subcommand><one blank space><pathname><new_line>.`
   - Some FTP clients may truncate trailing blanks.
4. The LIST and NLST subcommands, including all client subcommands that invoke the NLST subcommand, such as MGET or MDELETE, require special handling for certain special characters. For more information, refer to *OS/390 TCP/IP OpenEdition User's Guide*.
5. The START and SITE paramaters have added additional restrictions on the path name to use with SBDATACONN. Refer to *OS/390 TCP/IP OpenEdition Configuration Guide* and *OS/390 TCP/IP OpenEdition User's Guide*.
6. When specifying an OE FTP subcommand with a filename containing special characters, some FTP clients may truncate trailing blanks, compress multiple internal blanks, or interpret special characters to have special meanings. Special specification of the filename such as enclosing in double or single quotes, or escaping special characters, may be necessary to get the client to send the filename to the server correctly. Refer to your client documentation to see if this is necessary.

Common OE FTP Problems

Abend During Initialization

If the FTP server abends during initialization, check for the following problems:

- Are you starting FTP in an OS/390 Relese 3 system? Message EZYFT51I should indicate Version 1, Release 03.
- The C runtime library must be APF authorized.
- The C runtime library name must be either on the STEPLIB DD statement or in the MVS link list.
- The library containing FTPD and FTPDSRV must be APF authorized.

Socket Failures

The FTP server fails during initialization if it cannot complete the basic socket calls necessary to set up the FTP server control port. A message appears in the syslog if a socket error occurred.

If the FTP server is unable to set up the specified server port, check for the following problems:

- Are the control and data ports correctly reserved in the PORT statement of the PROFILE.TCPIP data set and the /etc/services file?

- Has the FTP server job been stopped and immediately restarted?

  If the server is restarted immediately after stopping, the port might not have had time to clean up and become available again. The FTP server job should be able to be restarted after a few minutes.

Data Set Allocation Fails

If data set allocation is failing (MKD, STOR/STOU, or APPE), check for the following:

- Issue the STAT command and check for problems with the variables that define data set characteristics (LRECL, RECFM, BLOCKSIZE, PRIMARY, SECONDARY, DIRECTORY).
  - Do they all have a valid value defined?
  - If the variable is not listed in the STAT command output, no value is assigned to this variable. If no value is assigned to the variable, then the value must be picked up from another source—either a model DCB or SMS. Does either the DCBDSN or DATACLASS (SMS) parameter have a valid value to provide a source for the missing variables?
  - If an SMS data class is specified, is SMS active at the server system? (Current SMS status is displayed as part of the output for the STAT command).
  - If an SMS data class is specified, do the data class definitions contain values for the missing variables?
  - Are PRIMARY and SECONDARY both either specified or not specified? If either PRIMARY or SECONDARY are specified, neither of the values will be picked up from an SMS data class. Both must be unspecified to pick up the value from SMS or both must be specified to override the SMS values.
  - If a model DCB is specified, are the characteristics of this data set valid for the data set being allocated?
- Issue the STAT command and check the PRIMARY, SECONDARY, and SPACETYPE values to determine how large the new data set will be. The VOLUME and UNIT value of the STAT command will tell you where the data sets are being allocated. (If neither volume or unit are shown by the STAT command, data sets will be allocated on the system default SYSDA DASD.) Does the server system have sufficient space where the data sets will be allocated to allocate the data set? The SITE QDISK command will provide information about the space available at the server system.
- Is the destination at the server site writeable? Check with the operator at the server system to verify that the destination of the new data set is not write protected.

Data Set Allocation Not Picking Up Correct Characteristics

If the data set is being allocated successfully, but the resulting data set does not have the expected data set characteristics, check for the following:

1. All values obtained from SITE variables
   - Issue the STAT command to verify that the settings of all the SITE variables are correct. If any variables are missing from the STAT output, check for values specified for the DCBDSN or DATACLASS parameters. If a

value is specified for the DCBDSN data set, refer to step 3 on page 83. If a value is specified for the DATACLASS parameter, refer to step 2 on page 83.

- Check for variables overridden by a client. The VM and MVS FTP clients both automatically issue SITE commands when doing a STOR, STOU, or APPE command. The values sent automatically by the client could be overriding values set by specific SITE commands issued by the user. To prevent the VM or MVS client from automatically sending new SITE settings, issue the SENDSITE command at the client.

2. Values from SMS

   If the DATACLASS parameter has been specified, but the actual data set characteristics do not match the values in the specified SMS data class, issue the STAT command and check the information shown in the output from the STAT command for the following:

   - Is SMS active at the server system? If SMS is not active, the SMS data class cannot be used to define the data set.
   - Are values specified for any of the data set characteristic variables (LRECL, RECFM, BLOCKSIZE, PRIMARY, SECONDARY, RETPD, DIRECTORY)? If these keywords are missing from the STAT output, then no value is assigned to them and the data set characteristics should be picked up from the SMS data class. If, however, a value is present for any of these variables, the setting shown by the STAT command will override any information in the SMS data class. To pick up the value from the data class, issue the SITE command with the keyword with no value (for example, `SITE RECFM`) to turn off the parameter setting.
   - Is a value specified for the DCBDSN parameter? If a DCBDSN data set is specified, the values for LRECL, RECFM, BLOCKSIZE, and RETPD will be obtained from the model DCB data set and will override any values in the SMS data class. Issue the `SITE DCBDSN` command to turn off the DCBDSN parameter setting.
   - Check for variables overridden by a client. The VM and MVS FTP clients both automatically issue SITE commands when doing a STOR, STOU, or APPE command. The values sent automatically by the client could be overriding values set by specific SITE commands issued by the user. To prevent the MVS or VM client from automatically sending new SITE settings, issue the SENDSITE command at the client.

3. Values from DCBDSN

   If the DCBDSN parameter has been specified, but the actual data set characteristics do not match the characteristics of the specified data set, issue the STAT command and check the information shown in the output from the STAT command for the following.

   - Are values specified for any of the data set characteristic variables (LRECL, RECFM, BLOCKSIZE, or RETPD)? If these keywords are missing from the STAT output, then no value is assigned to them and the data set characteristics should be picked up from the DCBDSN data set. If, however, a value is present for any of these variables, the setting shown by the STAT command will override the values of the DCBDSN data set. To pick up the value from the DCBDSN data set, issue the SITE command with the

keyword with no value (for example, `SITE RECFM`) to turn off the parameter setting.

- Check for variables overridden by a client. The VM and MVS FTP clients both automatically issue SITE commands when doing a STOR, STOU, or APPE command. The values sent automatically by the client could be overriding values set by specific SITE commands issued by the user. To prevent the VM or MVS client from automatically sending new SITE settings, issue the SENDSITE command at the client.

MVS Data Set Not Found

If the server is not able to find the MVS data set, check for the following problems:

- Issue the DIR command to display the data set. Can the server find the data set to list it?
- Is the MVS data set at the server in the catalog? The server can only locate cataloged MVS data sets.
- Was the *pathname* on the FTP command entered in single quotation marks? If not, the *pathname* specified will be appended to the end of the current working directory. Issue the PWD command to display the current working directory. If *current_working_directory.pathname* is not the correct name of the file, either change the current working directory with the CWD command or issue the correct data set name in single quotation marks as the *pathname*.

RETR, STOR, RNFR, RNTO, APPE, or DELE of Data Set Fails

Check for the following problems:

- Is the data set protected by a security system such as RACF or HFS permission bits?
- Is the data set "in use" at the server site by another program or user?
- Was the data set available to the system, or was it migrated or on an unmounted volume?
- Did the data set or member exist?

The following problems apply to MVS data sets only:

- Did the specified *pathname* follow MVS data set naming conventions?
- Was the requested data set a supported data set organization (PS, PDS, or PDS member) on a supported device type (dasd or tape)?
- Was the *pathname* specifications consistent with the type of data set? For example, if a member was requested, was the data set a PDS?

Data Transfer Terminated

Check for the following problems:

- Is the data set at the server large enough to receive the data being sent? If not, use the SITE command to change the space allocation for new data sets.
- If storing a member of a PDS, is there room in the PDS for an additional member? Is there room in the PDS directory for another directory entry?
- Did the client send an ABOR command?

- Is the filetype correct? For example, if filetype=SQL when it should be set to SEQ or JES, the host file being retrieved is assumed to be a SQL statement and FTP will attempt to connect to DB2 and submit the statement to DB2 for processing.

Client Abends during RETR Command Data Transfer

If the client abends while processing a RETR command, issue the STAT command and check the value of the Checkpoint interval. If this value is greater than 0, and data is being transferred in EBCDIC, either block mode or compressed mode, then the server is sending checkpoint markers with the data being transferred. If the client being used does not support checkpoint/restart, this checkpoint information can cause unpredictable results such as abends or data errors at the client. Change the setting of the checkpoint interval by issuing `SITE CHKPTI=0`.

Data Set Disposition Incorrect When Transfer Fails

- Data sets cataloged instead of deleted
  - Issue the STAT command and check the setting of the conditional disposition. If the STAT command output indicates `New data sets will be cataloged if a store operation terminates abnormally`, then the server will catalog new data sets even if the data transfer fails. To change this setting, issue the `SITE CONDDISP=DELETE` command.
  - Did the transfer fail due to the FTP server either abending or being terminated by a STOP or CANCEL command? If this is the case, the data set will be kept.
  - Is the client sending checkpoint information? If the data is being transferred in EBCDIC, either in block mode or compressed mode, and the client has sent at least one checkpoint marker, the FTP server will keep the data set even if the conditional disposition has been set to delete.
- Data sets deleted instead of cataloged
  - Issue the STAT command and check the setting of the conditional disposition. If the STAT command output indicates `New data sets will be deleted if a store operation terminates abnormally`, then the server will delete new data sets if the data transfer fails. To change this setting, issue the `SITE CONDDISP=CATALOG` command.

Checkpoint Markers Do Not Appear to Be Sent

Issue the STAT command and check the settings for data transfer. Checkpoint information is only transferred in EBCDIC, with either block or compressed mode. The checkpoint interval must be greater than 0.

The sender of the data initiates the checkpoint information; therefore, checkpointing must be set on at the client for a STOR, STOU, or APPE, (for the MVS V3R2 FTP client, this is done by issuing the `LOCSITE CHKPTINT=`*nn* command with a value larger than 0) and set on at the server (by issuing the `SITE CHKPTINT=nn` command with a value larger than 0) for a RETR.

LOADLIB Directory Information Is Not Sent with Module Transfer

Issue the STAT command and check the settings for data transfer. Load module directory information is only sent for EBCDIC with a mode of either block or compressed.

The client you are using must support the SDIR command.

Symptoms for SQL Problems

Before the FTP server can be used to submit queries to the DB2 subsystem, the following actions must be completed:

1. BIND the DBRM called EZAFTPMQ. This must be done anytime the part EZAFTPMQ.CSQLMVS has been recompiled.

   The DBRM must be bound into the plan named EZAFTPMQ unless the keyword DB2PLAN was used in your FTP.DATA file to specify a different plan name.

   If you are running multiple instances of the OE FTP server at different maintenance levels, you must use DB2PLAN in FTP.DATA for each server and specify unique plan names.
2. Grant execute privilege to the public for the plan created in the previous step.
3. Start the DB2 subsystem.

To submit a query to DB2 through FTP, the user must issue the following commands as necessary:

- `SITE FILETYPE=SQL`
- `SITE DB2=`*`db2name`* where *db2name* is the name of a DB2 subsystem at the host
- `RETR `*`fname1 fname2`* where *fname1* is a file at the host that contains a SQL SELECT statement

The following two tables shows some symptoms and possible causes of SQL problems. Table 13 on page 87 shows problems that generate a reply beginning with 55x. Table 14 on page 88 shows other SQL problems.

Table 13. SQL Problems Generating 55x Replies

Reply	Output file	Possible causes
Reply 551: Transfer aborted: SQL PREPARE/DESCRIBE failure	The output file contains the SQL code and error message returned by the DB2 subsystem.	• A syntax error in the SQL statement in the host file • The time stamp in the load module is different from the BIND time stamp built from the DBRM (SQL code = -818). This occurs if a BIND was not done for the EZAFTPMQ DBRM that corresponds to the current load module, or if the server is not configured to use the correct DB2 plan name. If this is the problem, every SQL query submitted through the FTP server will fail.
Reply 551: Transfer aborted: unsupport SQL statement	No output is sent from the host	• The file type is SQL, but the host file being retrieved does not contain an SQL SELECT statement. • Execute privilege was not granted for the DB2 plan the server is using.
Reply 551 Transfer aborted: attempt to connect to *db2name* failed (*code*)	No output is sent from the host	• The site db2name specifies a nonexistent DB2 subsystem. • The DB2 subsystem has not been started.
Reply 551: Transfer aborted: SQL not available. Attempt to open plan <planname> failed.	No output is sent from the host	• BIND was not done for the specified plan. • BIND was done for planname other than EZAFTPMQ, but FTP.DATA does not contain a DBZPLAN statement to specify this planname.
Reply 550: SQL query not available. Can't load CAF routines.	No output is sent from the host.	• The DSNLOAD library is not in the link list or the FTP server's STEPLIB.

Note: Refer to *OS/390 TCP/IP OpenEdition Messages and Codes* for more information about the message.

Table 14. Other SQL Problems	
Problem	**Possible causes**
Output file contains only the SQL SELECT statement	• The file type is SEQ, rather than SQL. If the file type is SEQ, a retrieve is done, but the host file is just sent back to the client. The query is not submitted to the DB2 subsystem. • The SELECT is for a VIEW for which the user ID does not have DB2 select privilege. The DB2 subsystem returns an empty table.
Client closes the connection because server is not responding	• The processing time needed by DB2 and FTP or both for the SQL query has exceeded the client's time limit for send or receive. An FTP server trace will indicate the amount of SQL activity through FTP and the approximate time when each query was processed.

JES Output Not Found

If the server is in FILETYPE=JES, and a job has been submitted but the output of the job cannot be found (that is, you get 0 spool files from a `DIR` command), check the following:

- Is the job name correct? The job name must be the user ID followed by a single character.
- Was the job output spooled to the hold queue? The server will only be able to retrieve job output that is in the hold queue.

Remote Job Submission Functions Fail

For problems with remote job submission, check for the following:

- Can't allocate internal storage
- JES is not communicating
- JES cannot find output for the specified job ID
- Unable to acquire JES access
- Unknown return code from GET JES spool request
- JES is unable to provide spool data set name now
- Unable to get a job ID for a PUT or GET request
- JES PUT or GET aborted, job not found
- JES PUT or GET aborted, internal error
- JES PUT or GET aborted, time-out exceeded
- JES internal reader allocation failed
- JES user exit error

To trace FTP's JES activity, use the JTRACE or UTRACE option on the MODIFY command.

Miscellaneous Server Problems

Listed below are other problems that you may encounter while running the OE FTP server.

1. The following entry appears in the FTP server trace: "RAnnnn pass2: seteuid failed (157/OB7F02AF): EDC5157I An internal error has occurred". This

happens when you have a “dirty” address space, which means that one of the libraries is not controlled or APF authorized.

2. The following entry appears in the FTP server trace: “SKnnnn data_connect:seteuid(0) error (111/OBOF02AF): EDC5111I Permission denied”. This occurs when you do not have BPX.daemon authority.
3. FTP server EZY messages are not being printed to the appropriate file.

   The diagnostic messages are printed out with the use of syslogd. Ensure that syslogd is currently active (check for /etc/syslog.pid). If syslogd is active, ensure that the syslog configuration file has an entry defined for daemon.info. If the syslog configuration file has an entry defined for daemon.info, ensure that the file specified for daemon.info exists. Ensure that the permissions on the file are at a minimum '666'.
4. OE FTP trace entries are not being printed to the appropriate file.

   The trace entries are printed out with the use of syslogd. Ensure that the syslogd is currently active (check for /etc/syslog.pid). If syslogd is active, ensure that the syslog configuration file has an entry defined for daemon.debug. If the syslog configuration file has an entry defined for daemon.debug, ensure that the file specified for daemon.debug exists. Ensure that the permissions on the file are at a minimum '666'.
5. Message "EZYFT14S listen error : EDC8115I Address already in use" appears when starting the server, and the server will not start.

   Another application has already been started using the same port that was specified for the FTP server. If the FTP server port is specified in one of the following:

   - /etc/services
   - ETC.SERVICES
   - or as a start parameter when starting the FTP server

   verify that the correct port number has been specified. If the port number has not been specified, the FTP server will attempt to use the well known FTP port 21. Verify that no other FTP servers have been started on this port. If the FTP server has been started using the correct port, issue the one of the following commands to determine which application is using the FTP server port:

   Against a V3R3 stack use the `onetstat -a` command For information on the onetstat command, refer to *OS/390 TCP/IP OpenEdition User's Guide*.

   Against a V3R2 stack use the NETSTAT command. For information on the NETSTAT command, refer to *TCP/IP for MVS: User's Guide*.

   This will also happen if you try to start the FTP daemon from INETD.
6. Server abends with U4088 when validating user ID/password.

   - Ensure that the sticky bit has been turned on for the files `/usr/sbin/ftpd` and `/usr/sbin/ftpdsrv`. Ensure that the FTPD and FTPDSRV modules reside in an APF authorized partitioned data set which is specified in the MVS linklist.
   - Ensure that all programs loaded into the FTP address space are APF authorized and are marked as controlled. This means that any FTP user exits, the SQL load library, and the loaded runtime library need to be

marked as controlled, using the RACF RDEFINE command. For more information, refer to *OS/390 OpenEdition Planning*, or see the RACF publications.

7. Message "EZYFT13S bind error : EDC5139I Operation not permitted" appears when starting the server and the server will not start.

   Ensure that the FTP server was started with a BPX.DAEMON authority.

8. PASS command failures

   - “550 error processing PASS command”
     - Ensure OE FTP server was started with BPX.DAEMON authority
     - If you started OE FTP from the shell, and the trace shows

       ```
       pass2:  initgroups failed (139/)bD60000):  EDC5139I Operation not permitted
       ```

       . Ensure OE FTP is started from superuser.
   - “550 PASS command failed — getpwnam() error: EDC5163I SAF/RACF extract error.”

     Ensure the user ID is defined to TSO, RACF and OMVS.

9. User exit routine does not get invoked.

   You can look in the FTP server trace to see if FTP is loading your exit. For example:

   ```
   main: ret code from fndmembr() for FTCHKIP is: 4
   main: user exit FTCHKIP not found. Bypassing fetch().
   ```

   or

   ```
   main: ret code from fndmembr() for FTCHKCMD is: 0
   main: chkcmdexit successfully loaded
   ```

   If you have user-written exit routines and the FTP server is not able to find them, ensure that the user-written exit routines exist in an APF-authorized partitioned data set which is specified in the MVS linklist.

10. Anonymous login fails.

    Ensure that you have specified ANONYMOUS as a start parameter or in FTP.DATA.

    If you **did not** specify a user ID on the ANONYMOUS start parameter or FTP.DATA statement, ensure that the user ID ANONYMO is defined to the MVS system as a TSO, RACF, and OMVS user ID.

    If you **did** specify a user ID on the ANONYMOUS start parameter or FTP.DATA statement, the specified user ID must be defined to MVS as a TSO, RACF, and OMVS user ID.

11. Message EZY2697, Message EZY2702, or reply 220 return zeros for the time and date information.

    The calendar time was not available to the C runtime library time() routine.

When All Else Fails

If the problem is not caused by any the common errors described in this section, collect the following documentation before calling the IBM Support Center:

1. OE FTP server dump (for abends).
2. OE FTP client output.
3. OE FTP server job output.
4. OE FTP server traces (refer to “Diagnosing OE FTP Problems with Traces” for information on collecting FTP server traces).
5. FTP.DATA data set.
6. TCPIP.DATA data set.
7. PROFILE.TCPIP data set.
8. ETC.SERVICES data set
9. The output from the STAT command issued to the server.

Diagnosing OE FTP Problems with Traces

Several types of traces are available to aid in debugging OE FTP server problems. Tracing of OE FTP server initializations is controlled by the TRACE start parameter or the TRACE keyword in FTP.DATA (see “Start Tracing” on page 92.) Once initialization is complete, tracing is controlled by options on the MODIFY command (see “Controlling the FTP Server Traces with MODIFY” on page 93).

You can trace activity for all clients connecting to the FTP server. When tracing for all clients, you can choose to trace general FTP activity, or JES-related activity, or both. You also can choose the level of detail to be included in the trace log. See “Tracing Activity for All Clients” on page 92.

Alternatively, you can trace all activity for a single user ID. See “Tracing Activity for One User ID” on page 93.

Where to Find Traces

The OE FTP server sends its trace entries to syslogd. The daemon.debug statement in /etc/syslog.conf specifies where syslogd will write FTP's trace records.

```
#
 # All ftp, rexecd, rshd
 # debug messages (and above
 # priority messages) go
 # to server.debug.a
 #
 daemon.debug              /tmp/syslogd/server.debug.a
```

All of OE FTP's trace entries (general or JES-related) are written to the same HFS file.

Note: The TRACE parameter and MODIFY options are issued to the FTP daemon and affect all client sessions that connect to the OE FTP server while tracing is active.

For more information on syslogd, refer to Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.

Start Tracing

This section discusses three methods of starting the FTP server traces. Two options start tracing during server initialization, while the third starts tracing after server initialization.

Start Tracing During FTP Initialization

You can use the TRACE start parameter or the TRACE statement in FTP.DATA to begin tracing during FTP daemon initialization; this continues tracing for all FTP events (except JES-related events) for all FTP sessions. The trace data is routed to a file in your HFS through a definition in your syslogd configuration file (/etc/syslog.conf).

Tracing remains active until you issue a MODIFY command to end it. Refer to “Controlling the FTP Server Traces with MODIFY” on page 93.

Note: When you issue a MODIFYcommand to end tracing, tracing will not occur for any subsequent client sessions; however, tracing will continue for any sessions that were already connected.

Start Tracing After FTP Initialization

After initialization, you can enable tracing using an MVS MODIFY command to the FTP server listener process. Refer to “Controlling the FTP Server Traces with MODIFY” on page 93. Already established FTP connections are not affected by a modify command. Only FTP connections that are established after the modify command was issued will be subject to tracing.

Stop Tracing

You stop global tracing via the MODIFY command, for example:

```
F FTPD1,NOTRACE
```

assuming you started your FTP listener process via a PROCLIB member with the name FTPD, which means the process to modify is FTPD1. You stop tracing for a single user like this:

```
F FTPD1,NOUTRACE
```

Already established FTP connections that were started with tracing enabled will continue to produce trace output until the connections are terminated, but new connections will start up without tracing enabled.

Tracing Activity for All Clients

The following options are used to control general and JES tracing for all user IDs. Both general and JES tracing can be active at the same time.

- **TRACE** begins the general trace for the FTP server. The general trace provides information such as logical paths followed and error conditions encountered for all clients connecting to the FTP server.
- **NOTRACE** ends general tracing.
- **JTRACE** begins the JES trace for the FTP server. The JES trace provides information such as logical paths followed and error conditions encountered for JES-related activity for all clients connecting to the FTP server.

- **NOJTRACE** ends JES tracing.

The general trace and JES trace are global traces, that is, they provide trace data for all user IDs. Additional options control the recording of detailed data, such as parameter lists and storage areas, in the trace logs.

- **DUMP** specifies that additional detailed data is to be included whenever the general trace (TRACE) is active.
- **NODUMP** specifies that detailed data is to be excluded from the trace log. This is the default.
- **JDUMP** specifies that additional detailed data is to be included whenever the JES trace (JTRACE) is active.
- **NOJDUMP** specifies that detailed data is to be excluded from the JES trace log. This is the default.

Tracing Activity for One User ID

Trace data for a specified user ID includes both general and JES-related activity and includes data such as parameter lists and storage areas. User tracing is controlled by the following options:

- **UTRACE** begins tracing for the specified user ID. Only one user ID can be traced at a time. All other tracing options in effect are suspended when the user trace is started and resumed when the user trace is stopped. No new trace options can be entered while the user trace is in effect, with the exception that a new user trace can be entered to change the user ID that is to be traced.
- **NOUTRACE** ends tracing for a specified user ID. Any global tracing options (for the general or JES traces) that were in effect when user tracing was begun are resumed when user tracing is ended.

Controlling the FTP Server Traces with MODIFY

The general trace for the FTP server can be started for all user IDs during initialization by specifying the TRACE parameter either as a start option in the FTP server start procedure or in the FTP.DATA data set.

Alternatively, tracing options can be started or stopped after initialization by issuing one of the following MVS MODIFY commands to the FTP server jobname.

Note: Jobname is the name associated with the FTP daemon background job. It is documented in EZYFT41I message in the FTP services log. If you started the OE server using a proc named FTPD, the jobname to use for the MODIFY command is probably FTPD1. As client sessions connect to the FTP server, the session process adapts the trace options currently active. These options remain in effect for the life of the client session process, regardless of subsequent MODIFY commands issued to the FTP daemon.

The following shows some examples of using the MODIFY command to control OE FTP server traces.

- MODIFY *ftp_server_jobname*,JTRACE
  - Starts the FTP server JES trace for all user IDs. This option can be issued anytime, except when the user trace is active.
- MODIFY *ftp_server_jobname*,NOJTRACE

  - Stops the FTP server JES trace for all user IDs. This option can be issued anytime, except when the user trace is active.

- MODIFY *ftp_server_jobname*,UTRACE=*user_id*
  - Starts the FTP server trace for the specified user ID.
  - If a previous user trace was in effect, the user ID previously being traced will no longer be traced. The new user ID will be traced instead.
  - If other trace options were in effect when the user trace was started they will be suspended until the user trace is stopped.
  - Any MODIFY commands to start other traces will be ignored while the user trace is in effect.
- MODIFY *ftp_server_jobname*,NOUTRACE
  - Stops the FTP server user trace.
  - Restores any global trace options (for the general or JES traces) that were in effect at the time the UTRACE option was entered to start tracing a specified user ID.
- MODIFY *ftp_server_jobname*,DUMP
  - Causes detailed data, such as parameter lists and storage areas, to be included in the general trace log.
  - This option can be issued before or after general tracing has been started. If it is issued while general tracing is active, it takes effect immediately. This option cannot be entered while a user trace is active.
- MODIFY *ftp_server_jobname*,NODUMP
  - Causes detailed data to be excluded from the general trace log.
  - This is the default.
  - This option can be issued before or after general tracing has been started. If issued while general tracing is active, it takes effect immediately. This option cannot be entered while a user trace is active.
- MODIFY *ftp_server_jobname*,JDUMP
  - Causes detailed data, such as parameter lists and storage areas, to be included in the JES trace log.
  - This option can be issued before or after JES tracing has been started. If issued while JES tracing is active, it takes effect immediately. This option cannot be entered while a user trace is active.
- MODIFY *ftp_server_jobname*,NOJDUMP
  - Causes detailed data to be excluded from the JES trace log.
  - This is the default.
  - This option can be issued before or after general tracing has been started. If issued while general tracing is active, it takes effect immediately. This option cannot be entered while a user trace is active.

Trace Examples and Explanations

Figure 26 shows an example of a trace of an OE FTP server. You can find a trace explanation in “Key to Figure” on page 98.

```
1  Mar 11 14:45:36  ftpd-6- EZYFT18I Using catalog '/usr/lib/nls/msg/C/ftpdmsg.cat' for FTP server messages.
2  Mar 11 14:45:37  ftpd-6- EZY2697I MVS TCP/IP Server-FTP V3R3 14:45:37 on 03/11/97
   Mar 11 14:45:37  ftpd-6- EZY2693I Unable to open /etc/ftp.data : EDC5129I No such file or directory.
   Mar 11 14:45:37  ftpd-6- EZY2693I Unable to open DD:SYSFTPD : EDC5129I No such file or directory.
   Mar 11 14:45:37  ftpd-6- EZY2693I Unable to open //'FTPD.FTP.DATA' : EDC5049I The specified file name
                               could not be located.
3  Mar 11 14:45:38  ftpd-6- EZY2640I Using 'SYS1.TCPPARMS(FTPDATA)' for local site configuration parameters.
   Mar 11 14:45:38  ftpd-6- EZYFT21I Using catalog '/usr/lib/nls/msg/C/ftpdrply.cat' for FTP replies.
   Mar 11 14:45:38  ftpd-6- EZY2693I Unable to open 'FTPD.SRVRFTP.TCPXLBIN' : EDC5049I The specified file
                               name could not be located.
   Mar 11 14:45:38  ftpd-6- EZY2693I Unable to open 'TCPIP.SRVRFTP.TCPXLBIN' : EDC5049I The specified file
                               name could not be located.
   Mar 11 14:45:38  ftpd-6- EZY2693I Unable to open 'FTPD.STANDARD.TCPXLBIN' : EDC5049I The specified file
                               name could not be located.
4  Mar 11 14:45:38  ftpd-6- EZYFT25I Using 'TCPIP.STANDARD.TCPXLBIN' for FTP translation tables for the
                              control connection.
   Mar 11 14:45:38  ftpd-6- EZYFT33I Unable to open DDNAME 'SYSFTSX' for the data connection:
                            EDC5129I No such file or directory.
   Mar 11 14:45:38  ftpd-6- EZY2693I Unable to open 'FTPD.SRVRFTP.TCPXLBIN' : EDC5049I The specified file
                               name could not be located.
   Mar 11 14:45:38  ftpd-6- EZY2693I Unable to open 'TCPIP.SRVRFTP.TCPXLBIN' : EDC5049I The specified file
                               name could not be located.
   Mar 11 14:45:38  ftpd-6- EZY2693I Unable to open 'FTPD.STANDARD.TCPXLBIN' : EDC5049I The specified file
                              name could not be located.
5  Mar 11 14:45:39  ftpd-6- EZYFT31I Using 'TCPIP.STANDARD.TCPXLBIN' for FTP translation tables for the data connection.
   Mar 11 14:45:39  ftpd-6- DM0623 main: __ipdspx returned TCPV33
6  Mar 11 14:45:39  ftpd-6- DM0639 dbcstables->__ip_dbcs_list-0¨ is: SJISKANJ.  Len is: 8
6  Mar 11 14:45:39  ftpd-6- DM0639 dbcstables->__ip_dbcs_list-1¨ is: EUCKANJI.  Len is: 8
6  Mar 11 14:45:39  ftpd-6- DM0639 dbcstables->__ip_dbcs_list-2¨ is: BIG5.  Len is: 4
7  Mar 11 14:45:40  ftpd-7- DM0868 main: actual length of NLSPATH environment variable: 18
   Mar 11 14:45:40  ftpd-7- DM0871 main: NLSPATH environment variable: /usr/lib/nls/msg/C/%N
   Mar 11 14:45:40  ftpd-7- DM0888 main: Using /usr/lib/nls/msg/C/%N for nlspath.
   Mar 11 14:45:40  ftpd-7- DM0915 main: LANG environment variable not defined.  Using default lang value.
   Mar 11 14:45:40  ftpd-7- DM0920 main: Using C for lang value.
   Mar 11 14:45:40  ftpd-7- DM0932 main: a2e table for control connection
   Mar 11 14:45:40  ftpd-7- 089AD2DA  00010203  372D2E2F  1605250B  0C0D0E0F   *.............*
   Mar 11 14:45:40  ftpd-7- 089AD2EA  10111213  3C3D3226  18193F27  221D351F   *.............*
   Mar 11 14:45:40  ftpd-7- 089AD2FA  405A7F7B  5B6C507D  4D5D5C4E  6B604B61   * !.#$&'()*+,-./*
      .
      .
      .
      .
   Mar 11 14:45:41  ftpd-7- 089AD3AA  D7D8D9E2  E3E4E5E6  E7E8E9AD  E0BD5F6D   *PQRSTUVWXYZ...._*
   Mar 11 14:45:41  ftpd-7- 089AD3BA  79818283  84858687  88899192  93949596   *.abcdefghijklmno*
   Mar 11 14:45:41  ftpd-7- 089AD3CA  979899A2  A3A4A5A6  A7A8A9C0  4FD0A107   *pqrstuvwxyx{.}..*
   Mar 11 14:45:41  ftpd-7- DM0934 main: e2a table for control connection
   Mar 11 14:45:41  ftpd-7- 089AD3DA  00010203  1A091A7F  1A1A1A0B  0C0D0E0F   *.............*
   Mar 11 14:45:41  ftpd-7- 089AD3EA  10111213  1A0A081A  18191A1A  1C1D1E1F   *.............*
   Mar 11 14:45:41  ftpd-7- 089AD3FA  1A1A1C1A  1A0A171B  1A1A1A1A  1A050607   *.............*
      .
      .
      .
      .
   Mar 11 14:45:41  ftpd-7- 089AD4AA  7D4A4B4C  4D4E4F50  5152A1AD  F5F4A38F   *'..<(+.&....54t.*
   Mar 11 14:45:41  ftpd-7- 089AD4BA  5CE75354  55565758  595AA085  8EE9E4D1   **X.......!.e.ZUJ*
   Mar 11 14:45:41  ftpd-7- 089AD4CA  30313233  34353637  3839B3F7  F0FAA7FF   *..........70.x.*
   Mar 11 14:45:41  ftpd-7- DM0937 main: a2e table for data connection
   Mar 11 14:45:41  ftpd-7- 089AD0DA  00010203  372D2E2F  1605250B  0C0D0E0F   *.............*
   Mar 11 14:45:41  ftpd-7- 089AD0EA  10111213  3C3D3226  18193F27  221D351F   *.............*
   Mar 11 14:45:41  ftpd-7- 089AD0FA  405A7F7B  5B6C507D  4D5D5C4E  6B604B61   * !.#$&'()*+,-./*
    .
    .
```

Figure 26 (Part 1 of 3). Example of a Trace of an OE FTP Server

```
     .
    Mar 11 14:45:41  ftpd-7- 089AD1AA  D7D8D9E2  E3E4E5E6  E7E8E9AD  E0BD5F6D   *PQRSTUVWXYZ...._*
    Mar 11 14:45:41  ftpd-7- 089AD1BA  79818283  84858687  88899192  93949596   *.abcdefghijklmno*
    Mar 11 14:45:41  ftpd-7- 089AD1CA  979899A2  A3A4A5A6  A7A8A9C0  4FD0A107   *pqrstuvwxyx{.}..*
    Mar 11 14:45:41  ftpd-7- DM0939 main: e2a table for data connection
    Mar 11 14:45:41  ftpd-7- 089AD1DA  00010203  1A091A7F  1A1A1A0B  0C0D0E0F   *................*
    Mar 11 14:45:41  ftpd-7- 089AD1EA  10111213  1A0A081A  18191A1A  1C1D1E1F   *................*
    Mar 11 14:45:41  ftpd-7- 089AD1FA  1A1A1C1A  1A0A171B  1A1A1A1A  1A050607   *................*
       .
     .
     .
     .
    Mar 11 14:45:41  ftpd-7- 089AD2AA  7D4A4B4C  4D4E4F50  5152A1AD  F5F4A38F   *'..<(+.&....54t.*
    Mar 11 14:45:41  ftpd-7- 089AD2BA  5CE75354  55565758  595AA085  8EE9E4D1   **X.......!.e.ZUJ*
    Mar 11 14:45:41  ftpd-7- 089AD2CA  30313233  34353637  3839B3F7  F0FAA7FF   *...........70.x.*
    Mar 11 14:45:41  ftpd-7- DM0947 main: msgcat for server msgs is; 144453664
 8  Mar 11 14:45:41  ftpd-7- DM0997 main: ret code from fndmembr() for FTCHKIP is: 4
 8  Mar 11 14:45:41  ftpd-7- DM1009 main: user exit FTCHKIP not found. Bypassing fetch().
    Mar 11 14:45:41  ftpd-7- EZYFT09I system information for MVSVIC03: OS/390 version 01 release 03.00 (9021)
    Mar 11 14:45:41  ftpd-7- DM1040 main: __librel returns 21030000
    Mar 11 14:45:41  ftpd-7- EZYFT51I OS/390 version 1, release 03, modification 0000.
 9  Mar 11 14:45:41  ftpd-7- DM1101 main: Initialization parameter values:     0, 1, 0, 0, 1, 1, 0, 1, 0, 1,
                              0, 0, 0, 621,     6144,   0, 15, 1, 0, 80, 600, 128, 128, 5, 80,-1, 2, 3,
                              C, , DB2, , , ,     MVSVIC03.TCP.RALEIGH.IBM.COM, FTPD1, TCPMGMT,
                              MIGRAT, N, , , , EZAFTPMQ
    Mar 11 14:45:41  ftpd-7- DM1103 main: daemon's code page is: IBM-1047
    Mar 11 14:45:41  ftpd-7- DM1105 main: daemon's locale is: C
    Mar 11 14:45:41  ftpd-7- EZY2700I Using port FTP control (621)
    Mar 11 14:45:41  ftpd-7- EZY2701I Inactivity time is 0
    Mar 11 14:45:41  ftpd-7- EZY2702I Server-FTP: Initialization completed at 14:45:41 on 03/11/97.
    Mar 11 14:45:41  ftpd-7- EZYFT41I Server-FTP: process id 7, server job name FTPD1
    Mar 11 14:45:41  ftpd-7- SK0315 accept_client: calling selectex for socket 6
    Mar 11 14:46:47  ftpd-7- SK0351 accepted client on socket 7
    Mar 11 14:46:47  ftpd-7- SK0400 New session for 9.67.43.62 port 34314
10  Mar 11 14:47:00  ftpd-16777222- SK1488 spawn_ftps: my pid is 16777222 and my parent's is 7
    Mar 11 14:47:00  ftpd-16777222- SK1749 setup_new_pgm: issuing execv
    Mar 11 14:47:01  ftpd-7- SK0315 accept_client: calling selectex for socket 6
    Mar 11 14:47:05  ftps-16777222- RX0738 main: msgcat is; 144298232
    Mar 11 14:47:05  ftps-16777222- RX0745 main: replycat is; 144302800
    Mar 11 14:47:05  ftps-16777222- SK1784 entering setup_client_sn with socket 7
    Mar 11 14:47:05  ftps-16777222- RX0321 main: ftp server client processing entered for socket 7
    Mar 11 14:47:05  ftps-16777222- RX0328 main: no LANG for child
    Mar 11 14:47:05  ftps-16777222- RX0332 main: child's codeset is: IBM-1047
    Mar 11 14:47:05  ftps-16777222- RX0335 main: child's locale is: C
11  Mar 11 14:47:05  ftps-16777222- RX0347 main: ret code from fndmembr() for FTCHKCMD is: 0
11  Mar 11 14:47:05  ftps-16777222- RX0353 main: chkcmdexit successfully loaded
11  Mar 11 14:47:05  ftps-16777222- RX0363 main: ret code from fndmembr() for FTCHKPWD is: 0
11  Mar 11 14:47:05  ftps-16777222- RX0369 main: chkpwdexit successfully loaded
11  Mar 11 14:47:05  ftps-16777222- RX0379 main: ret code from fndmembr() for FTCHKJES is: 0
11  Mar 11 14:47:05  ftps-16777222- RX0385 main: chkjesexit successfully loaded
    Mar 11 14:47:05  ftps-16777222- RX0411 main: SMF not specified in ftp.data file, so did not look for FTPSMFEX.
    Mar 11 14:47:05  ftps-16777222- RX0575 init_thread_site_vars: init_thread_site_vars routine entered.
```

Figure 26 (Part 2 of 3). Example of a Trace of an OE FTP Server

```
    Mar 11 14:47:05  ftps-16777222- PA0240 parse_cmd: entering parse_cmd.
    Mar 11 14:47:07  ftps-16777222- SK0468 get_command: select rc is 1
    Mar 11 14:47:07  ftps-16777222- SK0505 get_command: received 14 bytes
12  Mar 11 14:47:07  ftps-16777222- PA0378 Parse_cmd: command line: USER user121

    Mar 11 14:47:07  ftps-16777222- PA0395 parse_cmd: calling user exit FTCHKCMD with:
                                    rc 0, numparms 3, userid '', cmd 'USER' args '7/user121'.
    Mar 11 14:47:07  ftps-16777222- PA0415 parse_cmd: return from FTCHKCMD with rc: 0.
    Mar 11 14:47:07  ftps-16777222- RA0718 user: user routine entered with username 'user121'.
    Mar 11 14:47:11  ftps-16777222- SK0468 get_command: select rc is 1
    Mar 11 14:47:11  ftps-16777222- SK0505 get_command: received 14 bytes
12  Mar 11 14:47:11  ftps-16777222- PA0382 parse_cmd: processing PASS command
    Mar 11 14:47:11  ftps-16777222- PA0399 parse_cmd: calling user exit FTCHKCMD for PASS cmd.
    Mar 11 14:47:11  ftps-16777222- PA0415 parse_cmd: return from FTCHKCMD with rc: 0.
    Mar 11 14:47:11  ftps-16777222- RA0153 pass: pass routine entered.
    Mar 11 14:47:11  ftps-16777222- RA0265 pass: calling user exit FTCHKPWD with:
                                    rc 0, numparms 3, userid 'USER121'.
    Mar 11 14:47:11  ftps-16777222- RA0268 pass: return from chkpwd with rc: 0.
    Mar 11 14:47:11  ftps-16777222- RA0279 pass: termid is '09432B3E'.
    Mar 11 14:47:11  ftps-16777222- RA0331 pass: return from racf with: saf: 0, racf: 0,racf
                                    reason: 0 acee: 008F5378.
    Mar 11 14:47:11  ftps-16777222- RA0530 pass: calling delacee to delete useracee 008F5378.
    Mar 11 14:47:12  ftps-16777222- RA0655 pass: old asxbusr was 'USER121 '
    Mar 11 14:47:34  ftps-16777222- SK0468 get_command: select rc is 1
    Mar 11 14:47:34  ftps-16777222- SK0505 get_command: received 24 bytes
12  Mar 11 14:47:34  ftps-16777222- PA0378 Parse_cmd: command line: PORT 9,67,43,62,134,18

    Mar 11 14:47:34  ftps-16777222- PA0395 parse_cmd: calling user exit FTCHKCMD with:
                                    rc 0, numparms 3, userid 'USER121', cmd 'PORT' args '17/9,67,43,62,
                                    134,18'.
    Mar 11 14:47:34  ftps-16777222- PA0415 parse_cmd: return from FTCHKCMD with rc: 0.
    Mar 11 14:47:34  ftps-16777222- SK0903 entering port() with hostaddr 155396926 port 34322
    Mar 11 14:47:34  ftps-16777222- SK0468 get_command: select rc is 1
    Mar 11 14:47:34  ftps-16777222- SK0505 get_command: received 19 bytes
12  Mar 11 14:47:34  ftps-16777222- PA0378 Parse_cmd: command line: RETR test.example

    Mar 11 14:47:34  ftps-16777222- PA0395 parse_cmd: calling user exit FTCHKCMD with:
                                    rc 0, numparms 3, userid 'USER121', cmd 'RETR' args '12/test.example'.
    Mar 11 14:47:34  ftps-16777222- PA0415 parse_cmd: return from FTCHKCMD with rc: 0.
    Mar 11 14:47:34  ftps-16777222- RR0243 retr() pathname: test.example
    Mar 11 14:47:34  ftps-16777222- RR0312 retrieve: routine entered with pathname test.example
    Mar 11 14:47:34  ftps-16777222- RR0559 retrieve: hfs file is a regular file
    Mar 11 14:47:34  ftps-16777222- SK1214 data_connect: connection established for socket 8
    Mar 11 14:47:34  ftps-16777222- MV2352 seq_open_file: setenv for _EDC_ZERO_RECLEN
    Mar 11 14:47:34  ftps-16777222- MV2373 seq_open_file: recfm is NONE
    Mar 11 14:47:34  ftps-16777222- MV2467 seq_open_file: IST  -> r,recfm=*,NOSEEK for
                                    /u/user121/test.example
    Mar 11 14:47:34  ftps-16777222- MV2505 seq_open_file: stream 89C5434 has maxreclen 0
    Mar 11 14:47:34  ftps-16777222- RR2145 read_stream: entered
    Mar 11 14:47:34  ftps-16777222- MV2564 seq_read_file: eof reached
    Mar 11 14:47:34  ftps-16777222- SK1414 data_close: shutdown(ds 8)
    Mar 11 14:47:34  ftps-16777222- SK1422 data_close: recv(ds 8)
    Mar 11 14:47:34  ftps-16777222- SK1432 data_close: close(ds 8)
    Mar 11 14:47:34  ftps-16777222- MV1312 seq_close_file: close an open file
    Mar 11 14:47:34  ftps-16777222- MV1328 seq_close_file: file closed
    Mar 11 14:47:34  ftps-16777222- MV2604 seq_release_file: file is hfs - no release necessary
    Mar 11 14:47:34  ftps-16777222- RU1473 write_smf_record: entering write_smf_record with type 4.
    Mar 11 14:47:34  ftps-16777222- RU1934 write_smf_record: return code 0 from smfwr.
    Mar 11 14:47:34  ftps-16777222- RR1318 retrieve:           1476 bytes transferred
    Mar 11 14:47:38  ftps-16777222- SK0468 get_command: select rc is 1
    Mar 11 14:47:38  ftps-16777222- SK0505 get_command: received 6 bytes
12  Mar 11 14:47:38  ftps-16777222- PA0378 Parse_cmd: command line: QUIT

    Mar 11 14:47:38  ftps-16777222- PA0395 parse_cmd: calling user exit FTCHKCMD with:
                                    rc 0, numparms 3, userid 'USER121', cmd 'QUIT' args '0/'.
    Mar 11 14:47:38  ftps-16777222- PA0415 parse_cmd: return from FTCHKCMD with rc: 0.
    Mar 11 14:47:38  ftps-16777222- RM1449 quit: quit routine entered.
    Mar 11 14:47:38  ftps-16777222- SK0778 Ending session 08976528
    Mar 11 14:47:38  ftps-16777222- RX0549 Server thread terminates rc = 99.
```

Figure 26 (Part 3 of 3). Example of a Trace of an OE FTP Server

Key to Figure

1 Indicates the message catalog used by the FTP server, as determined by the NLSPATH and LANG environment variables. If the catalog cannot be opened, message EZYF501I displays, and the default messages are used.

2 The version and release of the FTP server.

3 The FTPD module looks for FTP.DATA during initialization. In this example, it was found in SYS1.TCPPARMS(FTPDATA).

4 Translate table used for the control connection.

5 Translate table used for the data connection.

6 These three lines show the keywords present on the LOADDBCSTABLES statement in TCPIP.DATA. The LOADDBCSTABLES statement specifies that double-byte languages can be used.

7 Note that the process ID number field of the trace record changes from 6 to 7. It indicates that the daemon forks to the background.

8 The server attempts to load the user-written exit routine, FTCHKIP. It was not loaded because it did not exist (return code = 4).

9 This lists setting of all FTP server configuration variables that are set during initialization. The variables appear in the following order:
anonymous
autotapemount
asatrans
imbedrdw
automount
autorecall
directorymode
quotesoverride
unit is tape
trace
spread
trailingblanks
wraprecord
server port
blocksize
checkpoint interval
directory
filetype
inactive time
jes lrecl
jes putget timeout
jes recfm
lrecl
primary
recfm
retention period
secondary
spacetype
conditional disposition
SMS dataclass

db2 name
dcbdsn
destination node
destination user
host name
job name
SMS mgmtclass
migration volume
sql column headings
SMS storclass
unit name
volume serial
DB2 plan

A series of commas, such as , , , indicates that a variable was not set in FTP.DATA.

These variables are described in *OS/390 TCP/IP OpenEdition Configuration Guide*.

These are the final values after the default has been set, the FTP.DATA data set has been read and processed, and the server start options have been processed. The value specified in the FTP.DATA data set will override the default, and the value specified for a start option will override the FTP.DATA data set value. Refer to *OS/390 TCP/IP OpenEdition Configuration Guide* for more information about the meaning of the FTP.DATA parameters and start options.

10 Note that the process ID number field of the trace record changes from 7 to 16777222. This is the process ID of the new server process that will handle the requests from this client. From now on, all trace records of process ID number 16777222 are server activities for the client at IP address 9.67.43.62 port 34314, until this session terminates.

11 The server loads the user-written exit routines. FTCHKCMD, FTCHKPWD, and FTCHKJES user exit routines are in effect for this server; they loaded with return code = 0.

12 This is the incoming command line after conversion to EBCDIC. Note that the PASS command is not reflected in the trace; this avoids exposing passwords.

OE Telnet Diagnosis

Chapter 8. OE Telnet Diagnosis

Common Problems

Listed below are common problems which you may encounter during execution of the Telnet daemon.

1. Diagnostic messages are not being printed to the appropriate file.

   a. The diagnostic messages are printed out with the use of SyslogD. Ensure that the SyslogD is currently active (check for /etc/syslog.pid).

   b. If SyslogD is active, ensure that the file where the output is intended to be outputted is currently allocated. SyslogD will not create the file; it expects it to exist. OE Telnet uses local1.debug for logging messages. Ensure that the syslog.conf file contains an entry for local1.debug or the *.* default file. See Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.

   c. Ensure also that the specified file exists. Ensure that the permissions on the file are at a minimum '666'.

   d. Make sure you specify `-t` and/or `-D all` as the OE Telnet options in /etc/inetd.conf.

2. Use of the arrow keys

   The arrow keys are not functional in “raw” mode. This is AIX-like behavior, except that, in AIX, the arrow key produces funny characters such as
   ¬--B

   on the screen to let the user know not to use arrows. Under rlogin, the cursor moves where you would want it to and correction is allowed, but the shell also treats these characters as part of the original command.

3. The keyboard appears locked and the user can not issue commands.

   When executing UNIX-type clients (for example, AIX), if the -k option is specified for telnet in inetd.conf, telnet does not allow kludge linemode (see “Setting up the inetd Configuration File” on page 111). UNIX-type clients require character-at-a-time mode to process correctly. If you remove the -k option from the parameters, then the software processes correctly.

   If this does not work, run tracing `-t D all`. Look for “Ept” to determine what the exception conditions are for the pty. The number of bytes should equal 4. Verify that the exception conditions identified are processed by the OE Telnet server. (Check EZYTE67I messages for more information; refer to 19 on page 108.)

4. EDC5157I An internal error has occurred, rsn=0b8802AF.

   The “2AF” of the reason code signifies that the user did not have the proper authority to execute the command. This may result in either the user's system having BPX.DAEMON authority set up in their environment, and the proper authorities have not been issued to the user, or the user does not have super user authority, which may be required to issue some of these commands.

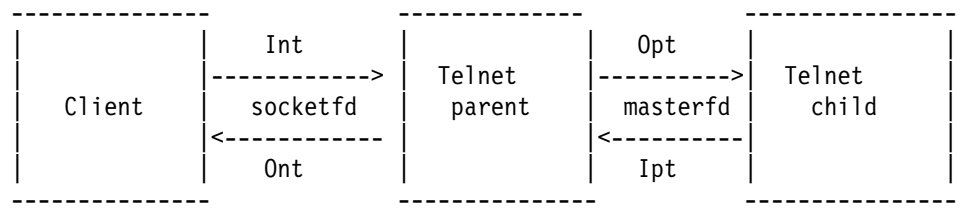
Debug Traces

These options relate to user-controlled trace information.

Table 15. Debug Trace Options

Option	Sub-Option	Description
-t		Internal tracing, intended to replace the DIAGNOSTICS compile option currently in place within the BSD code.
-D	options	Prints information about the negotiation of TELNET options
-D	report	Prints the options information, plus some additional information about what processing is going on.
-D	netdata	Displays the data stream received by telnetd.
-D	ptydata	Displays the data stream written to the pty.
-D	all	Supports all of the options/report/ptydata/netdata options.

Debug Trace Flows (netdata and ptydata)



When issuing any of the following 3 trace commands within /etc/inetd.conf (-D ptydata, -D netdata or -D all), you will have in your SyslogD file, the contents in both hex and ASCII, the data being sent over the sockets or between the tty's. If the user is having problems between the parent and the client, try the -D netdata option. If it is between the parent and the child, try the -D ptydata option. If both or either may apply, try the -D all option.

Each set of hex data will be preceded by a three-letter tag. This tag will represent the direction the data is flowing from. Above is a pictorial representation of this flow.

1. Int - client to parent
2. Ont - parent to client
3. Ipt - child to parent
4. Opt - parent to child

The user will type a command on the command line. It will flow Int -> Opt. The child will respond and the flow is Ipt -> Ont.

Debug Trace Examples (-t -D all)

Figure 27 on page 105 gives an example of the trace generated from `-t -D all`. This was generated from an AIX Telnet client. “Key to Figure” on page 107 explains the example.

```
1  EZYTE29I Starting new telnet session.  catfd = 134459616
   EZYTE04I catgets  EDC5000I No error occurred. rsn = 00000000
   EZYTO05I Initial EBCDIC codepage = IBM-1047, ascii codepage = ISO8859-1
2  EZYTE05I Trace 1 Debug 1d keepalive 1 kludgelinemode 2
   hostinfo 1 Registered host 0 linemode 1 multi_proc 1
3  EZYTE11I doit: host_name  rperot.raleigh.ibm.com
4  EZYTE11I doit: IP address 9.37.34.249
   EZYTE11I doit: PORT       1028
   EZYTE11I doit: host       MVS3
5  EZYTS04I STATE:send_do: send DO   TERMINAL TYPE¬(
   EZYTS04I STATE:send_do: send DO   TSPEED¬(
   EZYTS04I STATE:send_do: send DO   XDISPLOC¬(
   EZYTS04I STATE:send_do: send DO   NEW-ENVIRON¬(
   EZYTS04I STATE:send_do: send DO   OLD-ENVIRON¬(
   EZYTS04I STATE:send_do: send DO   BINARY¬(
   EZYTS09I STATE:send_will: send WILL   SUPPRESS GO AHEAD¬(
   EZYTS09I STATE:send_will: send WILL   ECHO¬(
6  EZYTU14I UTILITY: netwrite 24  chars.
7  EZYTU21I Ont: fffd18fffd20fffd23fffd27fffd24fffd00fffb ...................
   EZYTU21I Ont: 03fffb01 ....
8  EZYTU03I UTILITY:ttloop read 3 chars.
9  EZYTU47I Int: fffb18 ...
10 EZYTS05I STATE:willoption: receive WILL   TERMINAL TYPE¬(
   EZYTU03I UTILITY:ttloop read 21 chars.
   EZYTU47I Int: fffc20fffc23fffc27fffc24fffc00fffd03fffd ...................
   EZYTU47I Int: 01 .
11 EZYTS08I STATE:wontoption: receive WON'T   TSPEED¬(
   EZYTS08I STATE:wontoption: receive WON'T   XDISPLOC¬(
   EZYTS08I STATE:wontoption: receive WON'T   NEW-ENVIRON¬(
   EZYTS08I STATE:wontoption: receive WON'T   OLD-ENVIRON¬(
   EZYTS08I STATE:wontoption: receive WON'T   BINARY¬(
12 EZYTS10I STATE:dooption: receive DO   SUPPRESS GO AHEAD¬(
   EZYTS10I STATE:dooption: receive DO   ECHO¬(
13 EZYTU17I UTILITY: send suboption
   TERMINAL-TYPE
   SEND
   EZYTU14I UTILITY: netwrite 6  chars.
   EZYTU21I Ont: fffa1801fff0 .....0
   EZYTU03I UTILITY:ttloop read 4 chars.
   EZYTU47I Int: fffa1800 ....
   EZYTU03I UTILITY:ttloop read 7 chars.
   EZYTU47I Int: 7674323230fff0 ......0
```

Figure 27 (Part 1 of 3). OE Telnet Trace Using -t -D all

```
     EZYTS17I Defer suboption negotiation
     EZYTU14I UTILITY: netwrite 0  chars.
     EZYTU17I UTILITY: receive suboption
     TERMINAL-TYPE
     IS vt220
14   EZYTE10I terminaltypeok: call tgetent (buf, vt220)
    ¬(
     EZYTE48I Ont: c5e9 EZ
     EZYTE59I read_pw: Character ignored a
15   EZYTO04I lusername  = user79
    ¬(
     EZYTE48I Ont: c5e9 EZ
     EZYTE59I read_pw: Character ignored a
     EZYTE22I herald()
16     EZYTE26E herald: stat error  EDC5129I No such file or directory. rsn = 057C006C
     EZYTE16I uid = 215, gid = 5
     EZYTY03I GOTPTY: ioctl TIOCSWINSIZ   EDC5000I No error occurred. rsn = 00000000
     EZYTY05I GETPTY: slave fd = 10 , masterfd = 7
17   EZYTS15I STATE:dooption:deferred receive DO   ECHO¬(
     EZYTO09I options(1) = 3 .
     EZYTS15I STATE:dooption:deferred receive DO   SUPPRESS GO AHEAD¬(
     EZYTO09I options(3) = 3 .
17   EZYTS16I STATE:willoption:deferred receive WILL   TERMINAL TYPE¬(
     EZYTO09I options(24) = 12 .
     EZYTS04I STATE:send_do: send DO   LINEMODE¬(
     EZYTS04I STATE:send_do: send DO   NAWS¬(
     EZYTS09I STATE:send_will: send WILL   STATUS¬(
     EZYTS04I STATE:send_do: send DO   LFLOW¬(
     EZYTU14I UTILITY: netwrite 12  chars.
     EZYTU21I Ont: fffd22fffd1ffffb05fffd21 .............
     EZYTU03I UTILITY:ttloop read 6 chars.
     EZYTU47I Int: fffc22fffb1f ......
     EZYTS08I STATE:wontoption: receive WON'T   LINEMODE¬(
     EZYTS05I STATE:willoption: receive WILL   NAWS¬(
     EZYTS04I STATE:send_do: send DO   TIMING MARK¬(
     EZYTS04I STATE:send_do: send DO   BINARY¬(
     EZYTU14I UTILITY: netwrite 6  chars.
     EZYTU21I Ont: fffd06fffd00 ......
     EZYTE66I PROTOCOL:lmodetype=2, linemode=0, uselinemode=0
18   EZYTY08I argv_fsum(0) =  fomtlinp
     EZYTY08I argv_fsum(1) =  *4OurhrEa)R0,H/h
     EZYTY08I argv_fsum(2) =
```

Figure 27 (Part 2 of 3). OE Telnet Trace Using -t -D all

```
EZYTY08I argv_fsum(3) =  0
EZYTY08I argv_fsum(4) =  7
EZYTY08I argv_fsum(5) =  10
EZYTY08I argv_fsum(6) =  0
EZYTY08I argv_fsum(7) =  0
EZYTY08I argv_fsum(8) =  6
EZYTY08I argv_fsum(9) =  80
EZYTY08I argv_fsum(10) =
EZYTY08I argv_fsum(11) = vt220
EZYTY08I argv_fsum(12) =
EZYTY08I argv_fsum(13) =
EZYTY08I argv_fsum(14) =
EZYTY08I argv_fsum(15) =
EZYTY08I argv_fsum(16) = 1
EZYTY08I inherit flag =  40000000
EZYTY09I login_tty: spawnp fsumoclp 131080
EZYTE67I S(nfd):socketfd..ibits=00000001 obits=00000000 ebits=00000000
       S(nfd) pty..ibits=00000000 obits=00000000 ebits=00000080
EZYTE68I Ept: #bytes = 4 pkcontrol(cntl) 1003
EZYTE69I PROTOCOL: cntl = 1003
EZYTE65I PROTOCOL: send IAC Data Mark.  DMARK¬(
```

Figure 27 (Part 3 of 3). OE Telnet Trace Using -t -D all

Key to Figure

1. EZYTE29I indicates the start of a new OE Telnet client session.
2. EZYTE05I indicates what options were specified in /etc/inetd.conf for OE Telnet.
3. EZYTE11I indicates the resolved host name (from the client).
4. EZYTE11I shows the IP address of the OE Telnet client.
5. EZYTS04I indicates the terminal negotiation options sent to the client by the OE Telnet server.
6. EZYTU14I traces netwrites (writes to the client's terminal).
7. EZYTU21I traces data from parent to client; that is, OE Telnet to client's terminal.
8. EZYTU03I indicates the number of bytes read from the client by OE Telnet.
9. EZYTU47I traces data from the client to the parent (OE Telnet server).
10. EZYTS05I shows the terminal option negotiation the client has sent/responded with.
11. EZYTS08I shows the terminal option negotiation the client has sent/responded with.
12. EZYTS10I shows the terminal option negotiation the client has sent/responded with.
13. EZYTU17I traces OE Telnet sending terminal negotiation sub-options to the client.
14. EZYTE10I traces the call to tgetent(), which determines client terminal type.
15. EZYTO04I shows the user name with which the telnet client logged in.

16. EZYTE26 indicates no /etc/banner file was found.
17. EZYTS15I and 16I show that a state change was processed due to options/responses received from the client.
18. EZYTY08I traces the parameters passed to the spawned/forked child address space where the OMVS shell runs.
19. EZYTE67I traces the socket sets to show whether input/ibits, output/obits, or exception/ebits data has been received.
20. EZYTE68I shows exception data received on the parent/child connection.

Cleaning Up the utmp Entries Left Around From Dead Processes

Assuming that you have the suggested /etc/rc script, the utmpx file is cleaned up each time the S OMVS command is issued. The utmpx file should not normally need cleaning up, as each terminal slot should be reused the next time someone logs on with that terminal.

Although during normal processing the utmp entries are cleaned up, there are the occasional incidents where zombies are created, or the user may have terminated the session abnormally. When this occurs the utmp entry for that user will remain in the /etc/utmpx file until it is cleared out. There is an associated tty reserved for every entry in the /etc/utmpx file including the zombie entries. For dead entries, these ttys will not be available for reuse until someone under superuser erases the /etc/utmpx file.

Note: If you erase the file while someone is logged on, the next logoff will report not finding the utmpx entry for the user. This can be seen with a waitpid failure during that user's cleanup.

OE REXEC, OE REXECD, and OE RSHD Diagnosis

Chapter 9. OE REXEC, OE REXECD, and OE RSHD Diagnosis

Setting up the inetd Configuration File

inetd is a generic listener program used by such servers as OE TELNETD and OE REXECD. Other servers such as OE FTPD have their own listener program and do not used inetd.

inetd.conf is an example of the user's configuration file. It is stored in the /etc directory. Upon startup, the OE TELNETD server, rshell, rlogin, and rexec are initiated. If it does not include OE TCP/IP applications, add the following:

```
#======================================================================
# service | socket | protocol | wait/ | user | server  | server program
# name    | type   |          | nowait|      | program |   arguments
#======================================================================
#
 shell    stream    tcp        nowait  OMVSKERN /usr/sbin/orshd rshd -l
 exec     stream    tcp        nowait  OMVSKERN /usr/sbin/orexecd rexecd -LV
 otelnet  stream    tcp        nowait  OMVXKERN /usr/sbin/otelnetd otelnetd  -LV
```

Figure 28. Adding Applications to /etc/inetd.conf

To establish a relationship between the servers defined in the /etc/inetd.conf file and specific port numbers in the OpenEdition environment, insure that statements have been added to ETC.SERVICES for each of these servers. See the sample ETC.SERVICES installed in the /usr/lpp/tcpip/samples/services directory for how to specify ETC.SERVICES statements for these servers.

The traces for both the OE REXECD server and the OE RSHD server are enabled via options in the inetd configuration file (/etc/inetd.conf):

```
#======================================================================
# service | socket | protocol | wait/ | user | server  | server program
# name    | type   |          | nowait|      | program |   arguments
#======================================================================
#
shell      stream    tcp       nowait OMVSKERN /usr/sbin/orshd rshd -d  1
exec       stream    tcp       nowait OMVSKERN /usr/sbin/orexecd rexecd -d  2
```

Figure 29. Setting Traces in /etc/inetd.conf

The traces are turned on for both servers by passing a -d argument to the server programs. **1** is the RSHD server and **2** is the REXECD server. All commands executed after the debug flags have been turned on in the inetd configuration file and the inetd server has reread the file will produce trace output.

The trace is written in formatted form to the syslogd facility name daemon with a priority of debug. The trace data can be routed to a file in your Hierarchical File System by specifying the following definition in your syslogd configuration file (/etc/syslogd.conf):

```
  #
  # All ftp, rexecd, rshd
  # debug messages (and above
  # priority messages) go
# to server.debug.a
#
daemon.debug              /tmp/syslogd/server.debug.a
```

In this example, the trace data is written to /tmp/syslogd/daemon.debug.a in your Hierarchical File System. For more information on syslogd, refer to Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.

For more information about inetd, refer to *OS/390 OpenEdition Planning*, SC28-1890-02 or *Accessing OS/390 OpenEdition MVS from the Internet* (SG24-4721).

Diagnosing OE REXEC

The following kinds of information can help you diagnose an OE REXEC problem:

- A message beginning with EZYRC
- A code
- An OE REXEC debug trace
- A REXECD debug trace from the foreign host

Activating the OE REXEC Debug Trace

To activate the OE REXEC debug trace, specify the -d option.

OE REXEC Trace Example and Explanation

Enter this command with either a dotted decimal address or a host name.

```
orexec -d -l user21 -p xxx 9.67.113.61 ls -l
```

The following is an example of the trace output:

```
EZYRC01I  Calling function orexec with the following:
EZYRC02I  Host: 9.67.113.61, user user 21, cmd ls -l, port 512
```

EZYRC01I shows that we have called the OE REXEC function in the runtime libraries. EZYRC02I shows the parameters that have been passed to the REXEC() function in the runtime library.

Diagnosing OE REXECD

The following kinds of information can help you diagnose OE REXECD problem:

- A message beginning with EZYRD
- A code
- An OE REXECD debug trace
- A trace from the OE REXECD client

Activating the OE REXECD Debug Trace

To activate the OE REXECD debug trace, specify the -d option in the /etc/inetd.conf file.

OE REXECD Trace Example and Explanation

These examples are in the file specified in syslogd.conf.

Note: Syslogd must be running to collect these traces and the file must and have been properly specified.

```
Jun 12 13:31:47  rexecd.851981.: EZYRD31I  MVS OE REXECD BASE
```

The entry is stamped with the date, time, the name of and the pid number of the daemon, the message number (EZAYRD31I) and related information. For an explanation of the messages, see *OS/390 TCP/IP OpenEdition Messages and Codes*.

```
Jun 12 13:31:49  rexecd.851981.: EZYRD03I  Remote address = 9.67.113.61
Jun 12 13:31:49  rexecd.851981.: EZYRD05I  clisecport = 1029
Jun 12 13:31:49  rexecd.851981.: EZYRD08I  User is: user21
Jun 12 13:31:49  rexecd.851981.: EZYRD09I  Command is: ls -l
Jun 12 13:31:49  rexecd.851981.: EZYRD12I  Name is: USER21, user is user21
Jun 12 13:31:49  rexecd.851981.: EZYRD13I  dir is: /u/user21
Jun 12 13:31:49  rexecd.851981.: EZYRD14I  uid is: 21, gid is 0
```

Diagnosing OE RSHD

The following kinds of information can help you diagnose an OE RSHD problem:

- A message beginning with EZY4S01I
- A code
- An OE RSHD debug trace
- A trace from the RSH client

Activating the OE RSHD Debug Trace

To activate the OE RSHD debug trace, specify the -d option in the /etc/inetd.conf file.

OE RSHD Trace Example and Explanation

These examples are from the file specified in syslogd.conf.

Note: Syslogd must be running to collect these traces and the file must exist and have been properly specified.

```
Jun  9 12:10:04  rshd.4653080.: EZYRS01I  MVS OE RSHD BASE
```

The entry is stamped with the date, time, name of daemon and the pid number of the daemon, the message number (EZYRS01I) and related information. For an explanation of the messages, see *OS/390 TCP/IP OpenEdition Messages and Codes*.

```
Jun  9 12:10:06  rshd.4653080.: EZYRS12I  Clisecport = 1020
Jun  9 12:10:06  rshd.4653080.: EZYRS21I  Remote user is: OS2USER
Jun  9 12:10:06  rshd.4653080.: EZYRS22I  Local user is: user21
Jun  9 12:10:06  rshd.4653080.: EZYRS23I  Command is: ls -l
```

Note: If you enable the L or d option, the entry in SyslogD will include both user ID and password in clear text. Either do not specify these flags, or make sure your syslogd log files are properly protected

If the -U option is specified in /etc/inetd.conf, the OE RSHD server will not execute a command if the client host IP address cannot be resolved to a host name.

Resolving "Garbage" Errors

There are a few situations where the OE RSHD server may encounter an error so early in the processing of a command that the server has not established a proper EBCDIC-to-ASCII translation yet. In such a situation, the client end user may see “garbage” data returned to his or her terminal. A packet trace will reveal that the response is in fact returned in EBCDIC, which is the reason for the garbage look on an ASCII workstation. We have seen this happen if the OpenEdition name resolution has not been configured correctly, so the OE RSHD server, for example, was not able to resolve IP addresses and host names correctly. If your RSH clients encounter such a problem, please go back and check your name resolution setup. If you are using a local hosts table, make sure that the syntax of the entries in your hosts file is correct.

Diagnosing X Window System and OSF/Motif

Chapter 10. X Windows System and OSF/Motif

After installing service for OE X Window System or OSF/Motif, you will need to run a shell script identified in the ++HOLD ACT for the PTF. The shell script copies an existing archive library in /usr/lpp/tcpip/X11R6/lib, for example, libICE.a, to /usr/lpp/tcpip/X11R6/lib/libICE_old.a. If the shell script should fail (for example, due to lack of space in the HFS) the /usr/lpp/tcpip/X11R6/lib/libICE_old.a will have to be renamed back to the original name, /usr/lpp/tcpip/X11R6/lib/libICE.a, in order to use application archive library until the problem can be resolved.

An environment variable, XWTRACE, controls the generation of traces of the socket level communications between Xlib and the X Windows System Server.

- XWTRACE undefined or zero - No trace generated.
- XWTRACE=1 - Error messages
- XWTRACE>=2 - API function tracing for TRANS functions.

Another environment variable, XWTRACELC, causes a trace of various locale-sensitive routines. If XWTRACELC is defined, a routine flow trace is generated. If XWTRACELC=2, more detailed information is provided.

Following are some examples of X Windows System traces.

Trace Output When XWTRACE=2

The following trace example shows a typical stream of socket level activity that is generated when an X application running on OpenEdition MVS exchanges information with an X Server.

Each line of the trace provides:

- The name of the function involved from x11trans
- Values of the parameters passed to the function

```
TRANS(OpenCOTSClient) (/9.2.104.56:0)
TRANS(Open) (1,/9.2.104.56:0)
TRANS(SocketOpenCOTSClient) (inet,9.2.104.56,0)
TRANS(Connect) (3,/9.2.104.56:0)
TRANS(SocketINETConnect) (3,9.2.104.56,0)
TRANS(GetPeerAddr) (3)
TRANS(ConvertAddress)(2,16,7f9d288)
TRANS(SetOption) (3,2,1)
TRANS(SocketWritev) (3,225bc,1)
TRANS(SetOption) (3,1,1)
TRANS(SocketRead) (3,22344,8)
TRANS(SocketRead) (3,22344,8)
TRANS(SocketRead) (3,7f9d368,224)
TRANS(SocketWrite) (3,7f9eb88,60)
TRANS(SocketRead) (3,2249c,32)
TRANS(SocketRead) (3,2249c,32)
TRANS(SocketRead) (3,2249c,32)
TRANS(SocketWrite) (3,7f9eb88,56)
TRANS(SocketRead) (3,22518,32)
TRANS(SocketRead) (3,22518,32)
TRANS(SocketWrite) (3,7f9eb88,80)
TRANS(SocketWrite) (3,7f9eb88,20)
TRANS(SocketRead) (3,223e8,32)
TRANS(SocketRead) (3,223e8,32)
TRANS(SocketDisconnect) (7f9d2c0,3)
TRANS(Close) (3)
TRANS(SocketINETClose) (7f9d2c0,3)
```

Figure 30. Example of X Application Trace Output When XWTRACE=2

Trace Output When XWTRACELC=2

The following trace example is a partial trace showing typical types of information displayed by locale-sensitive routines. Each line of trace provides:

- The name of the locale routine
- The function invoked within that locale routine
- Where pertinent, charset name or encoding information or both
- If exiting the invoked function, the trace statement indicates the function is returning

```
 lcPubWrap:_XlcCreateLC(C)
 lcCT:_XlcAddCT(ISO8859-1:GL,"(B)
 lcCT:_XlcParseCT
 lcCT:_XlcGetCharSetFromEncoding( (B)
 lcCT:_XlcParseCT returning: 28 charset 0
 lcCharSet:_XlcCreateDefaultCharSet(ISO8859-1:GL,""a)
 lcCT:_XlcParseCharSet
 lcCT:_XlcParseCT
 lcCT:_XlcGetCharSetFromEncoding( (B)
 lcCT:_XlcParseCT returning: 28 charset 0
 lcCharSet:_XlcAddCharSet(ISO8859-1:GL)
 lcCharSet:_XlcGetCharSet(ISO8859-1:GL)
    returned NULL
 lcCT:_XlcAddCT   returning: 7f994d8
.
.       (trace statements in this section have been deleted)
.
 lcCT:_XlcAddCT(CNS11643.1986-1:GL,"$(H)
 lcCT:_XlcParseCT
 lcCT:_XlcGetCharSetFromEncoding( $(H)
 lcCT:_XlcParseCT returning: 2428 charset 0
 lcCharSet:_XlcCreateDefaultCharSet(CNS11643.1986-1:GL,"""+)
 lcCT:_XlcParseCharSet
 lcCT:_XlcParseCT
lcCT:_XlcGetCharSetFromEncoding( $(H)
 lcCT:_XlcParseCT returning: 2428 charset 0
 lcCharSet:_XlcAddCharSet(CNS11643.1986-1:GL)
 lcCharSet:_XlcGetCharSet(CNS11643.1986-1:GL)
    returned NULL
 lcCT:_XlcAddCT   returning: 7f9c4e0
 lcCT:_XlcAddCT(TIS620.2533-1:GR,"-T)
 lcCT:_XlcParseCT
 lcCT:_XlcParseCT returning: 80 charset 0
 lcFile:_XlcResolveLocaleName(C,"","""},"2h",)
 lcFile:_XlcResolveName(C,/usr/lib/X11/locale/locale.alias)
 lcFile:_XlcFileName(7f99420,locale)
 lcFile:_XlcResolveLocaleName(C,,"","","")
 lcFile:_XlcResolveName(C,/usr/lib/X11/locale/locale.alias)
 lcFile:_XlcResolveName(C,/usr/lib/X11/locale/locale.dir)
 lcDB:CreateDatabase(/usr/lib/X11/locale/C/XLC_LOCALE)
 ***
```

Figure 31 (Part 1 of 2). Example of X Application Trace Output When XWTRACELC=2

```
*** BEGIN Database
***
  0: XLC_XLOCALE, cs0.ct_encoding,      1: ISO8859-1:GL,
  1: XLC_XLOCALE, cs0.wc_encoding,      1: \x00000000,
  2: XLC_XLOCALE, cs0.length,   1: 1,
  3: XLC_XLOCALE, cs0.side,     1: GL:Default,
  4: XLC_XLOCALE, wc_shift_bits,        1: 8,
  5: XLC_XLOCALE, wc_encoding_mask,     1: \x00008080,
  6: XLC_XLOCALE, state_depend_encoding,        1: False,
  7: XLC_XLOCALE, mb_cur_max,   1: 1,
  8: XLC_XLOCALE, encoding_name,        1: STRING,
  9: XLC_FONTSET, fs0.font,     1: ISO8859-1:GL,
 10: XLC_FONTSET, fs0.charset,          1: ISO8859-1:GL,
***
*** END   Database
***
  lcDB:_XlcGetResource(7f99420,XLC_XLOCALE,mb_cur_max)
lcDB:_XlcGetLocaleDataBase(7f99420,XLC_XLOCALE,mb_cur_max)
lcDB:_XlcGetResource(7f99420,XLC_XLOCALE,state_dependent)
lcDB:_XlcGetLocaleDataBase(7f99420,XLC_XLOCALE,state_dependent)
   returning NULL
lcDB:_XlcGetResource(7f99420,XLC_XLOCALE,encoding_name)
lcDB:_XlcGetLocaleDataBase(7f99420,XLC_XLOCALE,encoding_name)
   returning lcd=7f99420
lcFile:_XlcResolveLocaleName(C,"",,"",)
lcFile:_XlcResolveName(C,/usr/lib/X11/locale/locale.alias)
```

Figure 31 (Part 2 of 2). Example of X Application Trace Output When XWTRACELC=2

Simple Network Management Protocol (SNMP) Diagnosis

Chapter 11. Simple Network Management Protocol (SNMP) Diagnosis

The SNMP protocol provides a standardized interface, through which a program on one host (the SNMP manager) can monitor the resources of another host (the SNMP agent).

Overview

Management Information Base (MIB)

The information maintained at each host is defined by a set of variables known as the Management Information Base, or MIB. In addition to the architected list of variables that must be supported by each SNMP agent, user-defined variables can also be supported by an SNMP agent. These user-defined variables that are not part of the architected MIB are known as enterprise-specific MIB variables.

On MVS, the majority of the MIB variables are maintained outside of the SNMP agent address space by programs known as SNMP subagents. The subagent program for the architected MIB variables, as well as the IBM MVS enterprise-specific MIB variables, is located in the TCPIP address space. Most objects supported by the SNMP agent are actually implemented by a subagent that is physically part of the OS/390 TCP/IP OpenEdition agent's address space.

In addition, user-written subagent programs can also exist. All subagent programs, whether provided by TCP/IP or user-written, communicate with the SNMP agent over an architected interface known as the Distributed Protocol Interface, or DPI.

PDUs

The SNMP protocol is based on the exchange of protocol data units, or PDUs. SNMPv2 has seven types of PDUs, six of which are used by OS/390 TCP/IP OpenEdition:

GetRequest-PDU Sent from the manager to request information from the agent

GetNextRequest-PDU Sent from the manager to request information from the agent (generally used to search tables).

GetBulkRequest-PDU Sent from the manager to request information from the agent. It requests the next variable in the tree and can also be used to specify multiple successors.

Note: The agent translates a GetBulkRequest into a series of GetNextRequests when the request is passed to the DPI subagent that has registered for the target of the request.

GetResponse-PDU Sent from the agent to return information to the manager

SetRequest-PDU Sent from the manager to alter information at the agent

SNMP Trap Sent from the agent to report network events to the manager

Inform-PDU IBM's agent does not generate SNMPv2 Inform PDUs.

Note: When the SNMP agent receives and authenticates a request, it passes the request to the DPI subagent that has registered as the target of the request. You can see this exchange by enabling DPI tracing within the agent. To determine why a request or function isn't working as expected, enable agent tracing to isolate the failure. For information on agent tracing, see “Starting SNMP Agent Traces” on page 134

Other Definitions

OS/390 TCP/IP OpenEdition's SNMP support includes an SNMP agent (osnmpd), the TCP/IP MVS subagent, and the OE SNMP command (osnmp). The osnmp command is a management application that is used to monitor and control network elements.

SNMP must be defined correctly to TCP/IP. See *OS/390 TCP/IP OpenEdition Configuration Guide*. In addition, several configuration data sets must be set up correctly for the SNMP agent (osnmpd). These are:

hlq.**PW.SRC**[2] Defines community names. Note that community name is a mixed-case, case-sensitive field.

hlq.**SNMPTRAP.DEST** Used for trap routing.

hlq.**TCPIP.DATA** Two statements are used by SNMP:

DATASETPREFIX Determines the high-level qualifier (hlq) for agent configuration files.

TCPIPjobname Determines the name of the stack with which SNMP attempts to establish its relationship through the SETIBMOPT call. For more information on TCPIPjobname, see Appendix A, “A Note on Search Paths” on page 159 or *OS/390 TCP/IP OpenEdition Configuration Guide*.

hlq.**PROFILE.TCPIP** While all statements are used indirectly by SNMP, several are particularly important for diagnosing SNMP problems:

SACONFIG defines certain configuration parameters for the subagent.

ITRACE specifies the level of tracing used by the subagent.

ATM DEVICE and LINK statements are used for SNMP to monitor ATM network information.

hlq.**OSNMPD.DATA** Contains MIB variables and their initial settings.

The following files are also needed for the osnmp command:

/etc/pw.src.cli (if specified) Defines the community names for validating GETRESPONSE and TRAP PDUs.

/etc/snmpv2.conf Defines the configuration data for sending requests to SNMP destinations.

/etc/mibs.data Defines the textual names for user variables not included in the compiled MIB shipped with the product.

---

[2] *hlq* indicates a high-level qualifier determined by the standard TCP/IP conventions documented in *OS/390 TCP/IP OpenEdition Configuration Guide*.

Refer to the *OS/390 TCP/IP OpenEdition Configuration Guide* in the chapter on "Configuring SNMP" for detailed information about these definitions.

Diagnosing SNMP Problems

Problems with SNMP are generally reported under one of the following categories:

- "Abends."
- Connection problems
  - "SNMP Connection Problems."
  - "Problems Connecting SNMP Agents to Multiple TCP/IP Stacks" on page 126.
  - "Problems Connecting Subagents to the SNMP Agent" on page 127.
- Incorrect output
  - "Unknown Variable" on page 129.
  - "Variable Format Incorrect" on page 131.
  - "Variable Value Incorrect" on page 132.
- "No Response from the SNMP Agent" on page 133.

Use the information provided in the following sections for problem determination and diagnosis of errors reported against SNMP.

Abends

An abend during SNMP processing should result in messages and error-related information sent to the system console. A dump of the error will be needed unless the symptoms match a known problem.

Documentation

Code a SYSMDUMP DD or SYSABEND DD statement in the PROC used to start the SNMP agent to ensure that a useful dump is obtained in the event of an abend.

Analysis

Refer to *OS/390 MVS Diagnosis: Procedures*, LY28-1082-02 or Chapter 3, "Diagnosing Abends, Loops, and Hangs" on page 21, for information about debugging dumps produced during SNMP processing.

SNMP Connection Problems

Problems Connecting to the TCPIP Address Space

Problems connecting the SNMP agent to the TCPIP address space are usually indicated by an error message in the SNMP agent syslogd output. For more information on reading the syslogd, see Appendix C, "Description of Syslog Daemon (syslogd)" on page 163.

Documentation: The following documentation should be available for initial diagnosis of problems connecting the SNMP agent to the TCPIP address space:

- PROFILE.TCPIP information.
- SNMP agent job output.
- Syslogd output from agent tracing at level 255 . For more information on reading the syslogd, see Appendix C, "Description of Syslog Daemon (syslogd)" on page 163.
- TCPIP.DATA information.

- OMVS console output for any command responses and traces.
- Include all the configuration files described earlier, under “Other Definitions” on page 124.

Analysis: Check the following for problems connecting the SNMP agent to the TCPIP address space:

1. Are you connected to the right TCPIP address space? This is obviously a concern when running multiple stacks. See “Problems Connecting SNMP Agents to Multiple TCP/IP Stacks.”
   - If you get a message “unable to connect to TCPIP JOBNAME,” you are not connected to the right address space. If you have defined two or more stacks, make sure your TCPIPjobname in the TCPIP.DATA data set used by the SNMP agent matches the NAME field on the SUBFILESYSTYPE statement for ENTRYPOINT(EZBPFINI) in the BPXPRMxx member you used to start OpenEdition MVS.
   - For more information, see *OS/390 OpenEdition Planning* or *Accessing OS/390 OpenEdition MVS from the Internet*.
2. Did any socket-related errors occur?
   - Check the SNMP agent syslogd for socket(), bind(), accept(), or other socket error messages. For more information on reading the syslogd, see Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.
3. Is the correct TCPIP.DATA information being used? Is the SYSTCPD DD statement coded in the PROC JCL?

If the problem still occurs after checking the above and making any needed changes, obtain the following documentation for problems connecting the agent.

- Dump of SNMP agent address space
- Dump of TCPIP address space
- The syslogd traces from the agent (using trace level 255). For more information on reading the syslogd, see Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.

Information on obtaining a dump can be found in *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02. Obtaining SNMP agent traces is discussed in “Starting SNMP Agent Traces” on page 134.

Problems Connecting SNMP Agents to Multiple TCP/IP Stacks

Each TCP/IP stack that is started requires its own SNMP agent. This requires that each agent have its own port and that it can find the TCPIPjobname of the TCP/IP stack that it wants to associate with.

Analysis: Check the following for problems connecting the SNMP agent to the correct TCP/IP stack.

1. Message EZZ6205I indicates that when _iptcpn() was called, it didn't return the correct TCPIPjobname for that agent. Check _iptcpn()'s search path, as indicated in Appendix A, “A Note on Search Paths” on page 159.
2. Message EZZ6272I indicates that the setibmopt call failed. This means that _iptcpn() returned a name that OpenEdition did not recognize as a PFS. Check the BPXPRMxx member (in SYS1.PARMLIB) used to configure OpenEdition.

Problems Connecting Subagents to the SNMP Agent

Problems connecting an SNMP subagent to the SNMP agent are generally indicated either by

- a socket error at the subagent
- authentication failures when the subagent attempts to open a connection
- a “no such name” response from the SNMPv1 agent when a manager requests a variable owned by the subagent
- a “no such object” response from the SNMPv2 agent when a manager requests a variable owned by the subagent

Documentation: The following documentation should be available for initial diagnosis of interface connection problems:

- PROFILE.TCPIP information
- SNMP agent job output, including syslogd output. For more information on reading the syslogd, see Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.
- Include all the configuration files described earlier, under “Other Definitions” on page 124.

Analysis: Check the following for problems connecting the SNMP subagent program to the SNMP agent:

1. Is the subagent in question the TCP/IP MVS subagent? If so,
   - Is the SACONFIG statement configured correctly?
   - Is SACONFIG disabled?
   - Is the port number correct?
   - Is the community name (or password) correct?

     Note: Note that community name is a mixed-case, case-sensitive field. Many times the client cannot get a response from an agent because the agent has a community string of "PUBLIC." Most clients default their community string to "public."
2. Did any socket-related errors occur?
   - Check the SNMP agent syslogd for socket(), bind(), accept(), or other socket error messages, particularly error messages related to the DPI connection.

If the problem still occurs after checking the above and making any needed changes, obtain the following documentation:

- SNMP agent 255 (trace all) output
- If the problem is with the TCP/IP MVS subagent, get the subagent traces. These are turned on by specifying the ITRACE statement in the PROFILE.TCPIP file. This can be done as part of the initial TCP/IP start-up. It can also be done after TCP/IP has been started by using the `VARY TCPIP` command, which is documented in Chapter 3 of *OS/390 TCP/IP OpenEdition Configuration Guide*.
- If the problem is with a user-written subagent program, use the DPIdebug() DPI library routine to collect dpi traces in the user-written subagent program. DPIdebug() sends output to the syslogd.

The following is a list of things to look for in the SNMP agent trace:

1. One of the following incoming SNMP GetRequest-PDU
   - dpiPortForTcp (1.3.6.1.4.1.2.2.1.1.1) for TCP connect. This is caused by `DPIconnect_to_agent_TCP`
   - dpiPathNameForUnixStream (1.3.6.1.4.1.2.2.1.1.3) for UNIX connect. This is caused by `DPIconnect_to_agent_UNIXstream`

Some questions to consider:

- Was the GetRequest-PDU received? If the GetRequest wasn't received, was it sent to the right port?

  In the case of the TCP/IP MVS subagent, the value of the AGENT keyword on the SACONFIG statement in the profile must match the value of –p that was specified (or defaulted) when the agent was invoked.
- Does it have a valid community name in the request?
  - SNMP subagents must use a valid (including correct case) community name as defined in the *hlq*.PW.SRC data set when requesting the dpiPort or dpiPath variable.
  - Note that community name is a mixed-case, case-sensitive field. It must be specified in the correct case on the COMMUNITY value on the SACONFIG statement, or it will not work.
- dpiPathNameForUnixStream defaults to /tmp/dpi_socket and provides an HFS pathname used in connecting a DPI subagent with the SNMP agent. The default can be overridden by using the `-s` parameter when starting the agent.

  A user-written subagent running from a non-privileged userid needs write access to the file. Otherwise, a subagent using DPIconnect_to_agent_UNIXstream() would have to be run from an OMVS superuser userid or other userid with the appropriate privilege.

2. Outgoing GetResponse-PDU for the dpiPort variable
   - Was the SNMP GetResponse-PDU sent back to the SNMP subagent?
   - Was it sent to the correct IP address?
   - Did it have the correct value for the DPI port?
     - The actual value for the DPI port for TCP can be determined by issuing an `onetstat –A` command at the SNMPv2 agent host. This will display the port on which the agent is accepting incoming UDP requests.
     - To display the dpiPath name, issue an osnmp get request for dpiPathNameForUnixStream.
3. Incoming subagent connection
   - Message `EZZ6244I Accepted new DPI inet subagent connection on fd` *`fd=xx`* `from inet address` *`xxxx port xxxx`*`.`

   or the following
   - `EZZ6246I Accepted new DPI inet socket connection on` *`fd=xx`*

   Note: *fd=xx* is the number associated with this specific subagent connection. Use it to correlate with later DPI trace messages. *xxxxx port xxxx* is the name and number of the port.
4. DPI packets transferred for this FD number.

The following documentation might also be needed in some cases, but it is suggested that the TCP/IP Software Support Center be contacted before this documentation is obtained:

- Dump of SNMP agent address space
- Dump of TCPIP address space (for TCP/IP MVS subagent problems)
- DUMP of user SNMP subagent address space
- Trace from agent and subagent in syslogd.

Information on obtaining a dump can be found in *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 . Obtaining SNMP agent traces is discussed in“Starting SNMP Agent Traces” on page 134 .

Incorrect Output

Unknown Variable

Unknown variable problems are usually caused by one of the following:

- A typographical error in the name or OID
- An incorrect instance number

  Note: A common mistake is forgetting to add a .0 to an object that is not part of a table . For example, a GET for ifNumber returns with “no such object” while performing a GET on ifNumber.0. If unsure how to proceed, try a GETNEXT on ifNumber. This will return the .0 version of the OID along with its value.

Documentation: The following documentation should be available for initial diagnosis of unknown variable problems:

- SNMP syslogd output with traces for both the agent and subagent For more information on reading the syslogd, see Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.
- /etc/mibs.data
- Include all the configuration files described earlier, under “Other Definitions” on page 124.

Analysis: Check the following for unknown variables at the SNMP agent:

1. If the variable is not defined in any compiled MIB, is the variable name included in the mibs.data file?
2. Did the DPI connection come up successfully?
   - Check the SNMP agent job output for messages indicating a problem in create_DPI_port.
   - If the DPI port was not successful, no SNMP subagents (including the OS/390 TCP/IP OpenEdition subagent) will be able to register MIB variables. The SNMP agent will have no knowledge of these unregistered variables and will report them as “NoSuchName” for SNMPv1 requests or “noSuchObject” for SNMPv2 requests.
3. Was the variable requested with the correct instance number?
   - Variables that are not in a table have an instance number of 0 . Variables that are part of a table will have more than one occurrence of the variable value. To get the value of the variable, you will need to request a specific instance of the variable. To find the instance number, issue a GET NEXT request; the first occurrence of the variable should be returned.

If the problem still occurs after checking the above and making any needed changes, obtain the following documentation:

For “variable not recognized by manager” messages:

- If the manager is the osnmp command, use the `-d 4` flag to get level 4 manager traces.

For “agent unknown variable”:

- SNMP agent level 15 trace output
- Traces from SNMP subagent programs (if the variable is supported by the MVS subagent)

The SNMP agent level 15 trace will show PDUs between the manager and agent as well as between the agent and any existing subagents. Things to look for in the trace:

1. Is the ASN.1 number received from the manager in the SNMP GetRequest-PDU correct?
2. Has a DPI packet registering the requested variable been received?
   - If not, if you know which subagent program owns the variable, check the subagent program for errors.
   - If the DPI register has been received, make a note of the FD number for further trace information.
3. Were any errors reported for this FD number after the DPI register request was received?
4. Was there a DPI information exchange over this FD number as a result of the incoming SNMP GetRequest-PDU?

Another approach to this problem is to look at the agent's saMIB variables. These include the following information:

- An entry for each subagent (including a field for subagent status)
- A table of all trees registered, including
  - Subagent to which the tree is attached
  - Status of the tree (valid, invalid, etc.)

This information can be useful when traces are not available.

The following documentation might also be needed in some cases, but it is suggested that the TCP/IP IBM Software Support Center be contacted before this documentation is obtained:

- Dump of SNMP agent address space
- Dump of TCPIP address space (for OS/390 TCP/IP OpenEdition subagent variables)
- Dump of user subagent address space

Information on obtaining a dump can be found in *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 . Obtaining SNMP traces is discussed later in “Starting SNMP Agent Traces” on page 134.

Variable Format Incorrect

Problems with incorrectly formatted variables are generally reported when the variable value from the GetResponse-PDU is displayed at the manager in the incorrect format (for example, as hexadecimal digits instead of a decimal value or a display string).

Documentation: The following documentation should be available for initial diagnosis of incorrectly formatted variables:

- /etc/mibs.data
- Include all the configuration files described earlier, under "Other Definitions" on page 124.

Analysis: Check the following:

1. Is the variable contained in the mibs.data file?
2. Is the value listed in the syntax position of the mibs.data file for this variable the correct syntax?
3. Is the variable value type correct? You will see this value if you use `osnmp -d` with any value greater than 0.
    - Refer to RFC 1442 [3] for the meanings of the variable value types.

If the problem still occurs after checking the above and making any needed changes, obtain the following documentation:

- For the TCP/IP subagent, subagent ITRACE level 4 output
- For user-written subagents, DPIdebug(2) output, which is sent to the syslogd. For more information on reading the syslogd, see Appendix C, "Description of Syslog Daemon (syslogd)" on page 163.
- SNMP manager level 4 trace output
- SNMP manager command output showing incorrectly formatted variable. This information appears in the trace output.
- SNMP agent level 2 trace output will show what was returned to the manager (osnmp) command

In the traces, verify that the variable value and syntax are passed correctly in the SNMP GetResponse-PDU from the agent to the SNMP manager.

The following documentation might also be needed in some cases, but it is suggested that the TCP/IP Software Support Center be contacted before this documentation is obtained:

- Dump of SNMP subagent address space
- Dump of SNMP agent address space

---

[3] You can find a list of RFCs for used by OS/390 TCP/IP OpenEdition in the appendix of *OS/390 TCP/IP OpenEdition Planning and Release Guide*. RFCs can be ordered from:

Government Systems, Inc.
Attn: Network Information Center
14200 Park Meadow Drive
Suite 200
Chantilly, VA 22021

Information on obtaining a dump can be found in *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 . Obtaining SNMP traces is discussed in “SNMP Traces” on page 134.

Variable Value Incorrect

Problems with incorrect variable values are generally reported when the variable value from the GetResponse-PDU displayed at the manager contains incorrect information.

Documentation: The following documentation should be available for initial diagnosis of variables with incorrect values:

- SNMP agent job output
- /etc/mibs.data
- Include all the configuration files described earlier, under “Other Definitions” on page 124.

Analysis: Check the following:

1. Were any errors reported at the SNMP agent when the variable was requested?

If the problem still occurs after checking the above and making any needed changes, obtain the following documentation:

- For the TCP/IP subagent, ITRACE level 4 trace output
- For user-written subagents, DPIdebug(2) output (which is sent to the syslogd. For more information on reading the syslogd, see Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.

In the traces, verify that the variable value is passed correctly from the SNMP subagent to the SNMP agent.

The following documentation might also be needed in some cases, but it is suggested that the TCP/IP IBM Software Support Center be contacted before this documentation is obtained:

- Dump of TCPIP address space (for OS/390 TCP/IP OpenEdition subagent variables)
- TCP/IP subagent traces from syslogd.

  The specific trace needed will depend on the variable in error.
- Incorrect values from the TCP/IP MVS subagent are probably due to an error in the TCP/IP stack. In this case, a dump of the TCP/IP address space and a CTRACE from the engine might be useful. You can also use the onetstat command to verify that the TCP/IP subagent is reporting what the TCP/IP engine believes the value to be.

Information on obtaining a dump can be found in *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 . Obtaining SNMP traces is discussed in “SNMP Traces” on page 134.

No Response from the SNMP Agent

Problems receiving a response from the SNMP agent are generally reported when an SNMP request is issued from a manager but no response from the agent is received. This is usually reported as a “time-out”message.

Documentation

The following documentation should be available for initial diagnosis when no response is received from the agent:

- OMVS console output for any command responses and traces
- SNMP syslogd output. For more information on reading the syslogd, see Appendix C, “Description of Syslog Daemon (syslogd)” on page 163.
- The /etc/SNMPv2.conf file (if the osnmp command is the manager)
- *hlq*.PW.SRC information
- etc/pw.src.cli (HFS file), if one exists.

  The password being used (or defaulted) on an osnmp request must be acceptable to both the manager and the agent. The osnmp command uses the /etc/pw.src.cli file if one exists. If not, the osnmp command uses the same search order as the agent does to find which PW.SRC file it should use.
- Include all the configuration files described earlier, under “Other Definitions” on page 124.

Analysis

Check the following when no response is received from an agent:

1. Is the SNMP agent running?
2. What is the time-out value? For example, the time-out value on the osnmp command defaults to 3 seconds. Trying the request again with a larger time-out value, such as 15 seconds, may result in an answer.
3. Does the request use the correct port number and IP address?
4. Where any errors reported at the SNMP agent when the variable was requested?
5. Is the correct community name (including correct case) being used in the request?

   Note: Note that community name is a mixed-case, case-sensitive field. Many times the client cannot get a response from an agent because the agent has a community string of "PUBLIC." Most clients default their community string to "public."

If the problem still occurs after checking the above and making any needed changes, obtain the following documentation:

- SNMP agent level 7 trace output

Check the following in the SNMP agent traces:

1. Was the SNMP request PDU received by the agent?
2. Did it have a valid community name? Note that community name is case-sensitive and mixed-case.
3. Was the IP address of the manager the expected IP address?
4. Was an SNMP GetResponse-PDU sent back to the manager?
5. Was an AuthenticationFailure trap generated?

Note: For these traps to be generated, you must first provide SNMPTRAP.DEST information and set the snmpEnableAuthenTraps MIB variable to 1, meaning traps are enabled. For detailed information on enabling traps, refer to *OS/390 TCP/IP OpenEdition Configuration Guide*.

6. If the manager is the osnmp command, does it use the /etc/pw.src.cli file? If so, make sure that file contains an entry indicating that the community name may be used as the agent's IP address.
7. If the manager is the osnmp command, was the command destination specified using the `-h` parameter? Did the `-h` value have an entry in the /etc/snmpv2.conf file? If so, and the destination agent only supports SNMPv1, the request will be discarded. Try reissuing the command using an IP address as the value of the `-h` parameter. If an entry with the value specified on the -h parameter does not exist in the SNMP.conf file, an SNMPv1 request will be sent. An SNMP agent can process SNMPv1 requests, but an SNMPv1 agent cannot process SNMP requests.

The following documentation might also be needed in some cases, but it is suggested that the TCP/IP IBM Software Support Center be contacted before this documentation is obtained:

- Dump of SNMP agent address space

Information on obtaining a dump can be found in *OS/390 MVS Diagnosis: Tools and Service Aids*, LY28-1085-02 . Obtaining SNMP traces is discussed in “SNMP Traces.”

SNMP Traces

There are three types of traces that can be useful in identifying the cause of SNMP problems:

1. Manager traces
2. Agent traces
3. Subagent traces

These traces are discussed in the following sections.

Starting SNMP Manager Traces

To start manager tracing, use `osnmp -d` when you invoke the agent through OMVS. You can specify a trace level of 0 to 4. A level 0 trace provides no tracing, while a level 4 provides the most. Tracing for the osnmp command is done on a per-request basis. Traces come back to the console.

Starting SNMP Agent Traces

If Agent is Not Running

If the SNMP agent isn't already running, specify the -d parameter when you invoke the agent. You can start the SNMP agent in one of two ways:

- through the start options in the JCL used to start the SNMP agent (more common). For example,

  ```
  //OSNMPD    EXEC PGM=EZASNMPD,REGION=4096K,TIME=1440,PARM='-d 8'
  ```

- through OMVS, using the `osnmpd` command

Use one of the following trace levels:

1 trace SNMP requests

2 trace SNMP responses

4 trace SNMP traps

8 trace DPI packets level 1

16 trace DPI internals level 2

32 internal trace (debug level 1)

64 extended trace (debug level 2)

128 trace DPI internals level 2

Trace records are sent to the file specified by the daemon.debug entry in the /etc/syslog.conf file.

Combining trace levels: To combine trace levels, add trace level numbers. For example, to request SNMP requests (level 1) and SNMP responses (level 2), you would request trace level 3.

If Agent is Already Running

You can use the MVS MODIFY command to start and stop trace dynamically. Use of this support is restricted to the users with MODIFY command privilege. The MODIFY syntax looks like this:

MODIFY procname,parms=*parm*

For details on the MODIFY command, see *OS/390 TCP/IP OpenEdition Configuration Guide*.

If you start the agent from JCL, you have no difficulty knowing the procname. However, if you start the agent from OMVS, the agent generates a message number to syslogd. This message indicates the jobname the agent is running; this is the jobname to specify on the MODIFY command.

Changing Agent Trace Levels

The MVS MODIFY command can also be used to change trace levels. For example, assume the procname is OSNMPD and you want to change the trace level to 3 (tracing SNMP requests and responses). You would enter

```
MODIFY OSNMPD,trace,level=3
```

For more information on using the MVS MODIFY command, see *OS/390 TCP/IP OpenEdition User's Guide*.

Starting SNMP Subagent Traces

To start the SNMP subagent traces, code the ITRACE statement in the *hlq*.PROFILE.TCPIP file or in a file to be used by the `VARY TCPIP` command, which is documented in Chapter 3 of *OS/390 TCP/IP OpenEdition Configuration Guide*.

ITRACE ON SUBAGENT *level*

where *level* Use one of the following values:

1 general subagent trace information

2 general subagent trace information plus DPI traces

3 general subagent trace information plus extended dump trace. This level provides storage dumps of useful information, such as storage returned by the IOCTL calls.

4 general subagent trace information plus extended dump trace and DPI traces.

The trace output is sent to the syslogd. Trace records are found in the file specified by the daemon.debug entry in the /etc/syslog.conf file.

To stop SNMP subagent traces, code the ITRACE statement in the *hlq*.PROFILE.TCPIP or the file to be used by the VARY OBEYFILE as follows:

```
ITRACE OFF SUBAGENT
```

For more information on the VARY command, see *OS/390 TCP/IP OpenEdition Configuration Guide*.

Trace Examples and Explanations

The following examples are shown in this section

1. Agent trace
2. Subagent traces

SNMP Agent Traces

Figure 32 on page 137 was produced by using `osnmp get sysUpTime.0`. When the SNMP agent is tracing responses, it makes the following entry in the syslogd output file:

```
Dec 19 15:55:38  snmpagent.9.: SNMP logging data follows ==============

Dec 19 15:55:39  snmpagent.9.: Log_type:        snmpLOGresponse_out
Dec 19 15:55:39  snmpagent.9.:    send rc:      0
Dec 19 15:55:39  snmpagent.9.:    destination:  UDP 127.0.0.1 port 5000
Dec 19 15:55:39  snmpagent.9.: version:         SNMPv1
Dec 19 15:55:39  snmpagent.9.: community:       public
Dec 19 15:55:39  snmpagent.9.:  ('70 75 62 6c 69 63'h)
Dec 19 15:55:39  snmpagent.9.: addressInfo:     UDP 127.0.0.1 port 5000
Dec 19 15:55:39  snmpagent.9.: PDUtype:         GetResponse ('a2'h)
Dec 19 15:55:39  snmpagent.9.: request:         1
Dec 19 15:55:39  snmpagent.9.: error-status:    noError (0)
Dec 19 15:55:39  snmpagent.9.: error-index:     0
Dec 19 15:55:39  snmpagent.9.: varBind oid:
Dec 19 15:55:39  snmpagent.9.: OBJECT_IDENTIFIER
Dec 19 15:55:39  snmpagent.9.: 1.3.6.1.2.1.1.3.0
Dec 19 15:55:39  snmpagent.9.:         name:    sysUpTime.0
Dec 19 15:55:39  snmpagent.9.:         value:
Dec 19 15:55:39  snmpagent.9.: TimeTicks
Dec 19 15:55:39  snmpagent.9.: 5900 - 59.00 seconds
Dec 19 15:55:39  snmpagent.9.: End of SNMP logging data:
```

Figure 32. SNMP Agent Response Trace

In the following scenario, the SNMP agent attempted to initialize, but it was not successful. The port it was using was already in use. The following trace was obtained with SNMP agent tracing set to 7.

```
Dec 19 11:57:52  snmpagent.16777227.: EZZ6235I socket function failed for
SNMP inet udp socket; EDC5112I Resource temporarily unavailable.
Dec 19 11:57:52  snmpagent.16777227.:  ... errno = 112, errno2 =12fc0296
```

Figure 33. SNMP Agent Trace of Unsuccessful Initialization

Note: Errno 112 translates to “Resource temporarily unavailable.” The errno is used primarily by IBM service in diagnosing the error. In this case, issue `onetstat -c` to determine if TCP/IP is running and, if so, which ports are in use.

Subagent Trace

When requests for MIB variables supported by the TCP/IP MVS subagent fail with an indication that the variable is not supported (noSuchName or noSuchObject), one possibility is that the TCP/IP MVS subagent was unable to connect to the SNMP subagent.

The figure below illustrates a scenario where the subagent is unable to connect because the password it is using is not accepted by the SNMP agent. (The password used by the subagent is defined or defaulted on the SACONFIG statement in the TCP/IP profile.) The following traces were obtained with SNMP agent traces set to any non-zero value and the subagent traces (as set on the ITRACE profile statement) set to 1.

```
Dec 19 16:05:18  snmpagent.100663307.: EZZ6225I SNMP AGENT: INITIALIZATION
COMPLETE
Dec 19 16:05:22  snmpagent.100663307.: S_AGV1V2 WITHDEF  A1(1092):
Dec 19 16:05:22  snmpagent.100663307.: S_AGV1V2 WITHDEF  A1 (1092) rc=-14
(SNMP_RC_NOT_AUTHENTICATED) from snmp_process_message()
Dec 19 16:05:31  M2SubA.33554437.: DPI2334 do_connect_and_open: DPIconnect_to_agent_UNIXstream
rc -2.
Dec 19 16:06:16  M2SubA.33554437.: DPI2304 do_connect_and_open .... getting
agent info from config
Dec 19 16:06:16  M2SubA.33554437.: DPI2315 do_connect_and_open: port 161
Dec 19 16:06:16  M2SubA.33554437.: DPI2326 do_connect_and open: hostname_p
=> 127.0.0.1
Dec 19 16:06:16  snmpagent.100663307.: S_AGV1V2 WITHDEF  A1(1092):
Dec 19 16:06:16  snmpagent.100663307.: S_AGV1V2 WITHDEF  A1 (1092) rc=-14
(SNMP_RC_NOT_AUTHENTICATED) from snmp_process_message()
```

Figure 34. SNMP Subagent Trace

OE RouteD Diagnosis

Chapter 12. OE RouteD Diagnosis

The OE route daemon (OE RouteD) is a server that implements the Routing Information Protocol (RIP). It provides an alternative to the static TCP/IP gateway definitions. When configured correctly, the MVS host running with OE RouteD becomes an active RIP router in a TCP/IP network. OE RouteD for OS/390 TCP/IP OpenEdition is functionally equivalent to the TCP/IP V3R2 for MVS version with the following exceptions:

1. OE RouteD is an OE application. It requires the HFS file system to run.
2. OE RouteD can be started from an MVS procedure, or it can be started from the OE shell.
3. OE RouteD uses a standard message catalog. The message catalog must be in HFS. The directory location for the message catalog path is set by the environment variables NLSPATH and LANG. See *OS/390 TCP/IP OpenEdition Configuration Guide* for details.
4. The location and search order for configuration files has changed. See Appendix A, “A Note on Search Paths” on page 159.
5. Program output is sent to syslogd unless you specify otherwise. See “Where to Send OE RouteD Trace Output” on page 149.

Before OE RouteD was implemented for TCP/IP, static route tables were used for routing IP datagrams over connected networks. However, the static routes had a drawback in that they were not able to respond to changes in the network. By implementing the Routing Information Protocol (RIP) between a host and TCP/IP, the OE RouteD server dynamically updates the internal routing tables when changes to the network occur.

The OE RouteD server reacts to network topology changes on behalf of TCP/IP by maintaining the host’s routing tables, processing and generating RIP datagrams, and performing error recovery procedures.

Figure 35 on page 142 shows the OE RouteD environment.

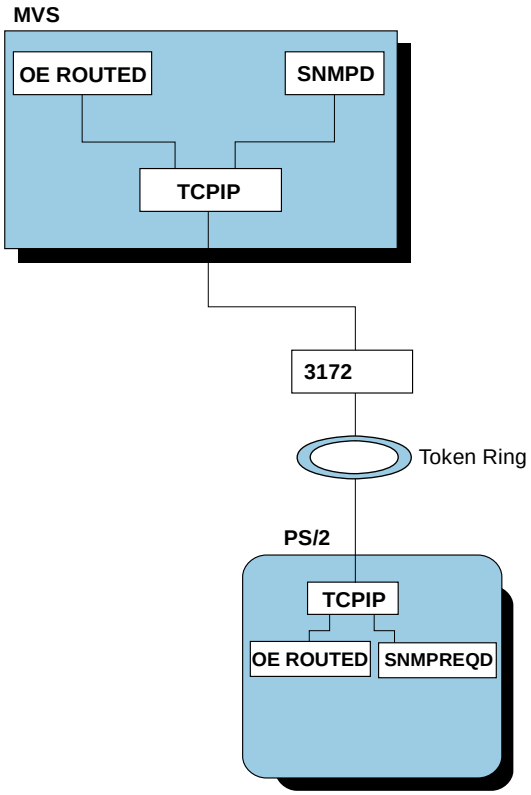


Figure 35. OE RouteD Environment

The OE RouteD protocol is based on the exchange of protocol data units (PDUs). There are two types of PDUs:

- **Request PDU**: Sent from a client (another RIP router) as a request to retransmit RIP broadcasts.
- **Response PDU**: Sent from OE RouteD to a client (another RIP router) as a response to retransmit RIP broadcasts.

Definitions

OE RouteD must be defined correctly to TCP/IP. Refer to *OS/390 TCP/IP OpenEdition Configuration Guide* in the chapter about “Configuring OE RouteD” for detailed information about TCP/IP definitions.

Diagnosing OE RouteD Problems

Problems with OE RouteD are generally reported under one of the following categories:

- “OE RouteD Connection Problems” on page 143
- “OE PING Failures” on page 144
- “Incorrect Output” on page 145
- “Session Outages” on page 146

Use the information provided in the following sections for problem determination and diagnosis of errors reported against OE RouteD.

OE RouteD Connection Problems

OE RouteD connection problems are reported when OE RouteD is unable to connect to TCP/IP. Generally, this type of problem is caused by an error in the configuration or definitions in TCP/IP.

In a CINET environment (multiple stacks) , OE RouteD attempts to connect to a stack whose name is determined by the TCPIPjobname keyword from the resolver configuration data set or file. The TCPIPjobname must match the NAME field for ENTRYPOINT(EZBPFINI), which is on the SUBFILESYSTYPE statement for the CINET PFS. If OE RouteD can't determine the TCPIPjobname, it will use a default TCPIPjobname of INET. If OE RouteD can't communicate with the stack pointed to by TCPIPjobname, it ends and issues an error message.

In configurations with multiple stacks, a copy of OE RouteD must be started for each stack requiring OE RouteD services. To associate OE RouteD with a particular stack, use the environment variable RESOLVER_CONFIG to point to the data set or file that defines the unique TCPIPjobname.

Documentation

The following documentation should be available for initial diagnosis of OE RouteD connection problems:

- PROFILE.TCPIP information or a data set defined in the //SYSENV statement in the OROUTED cataloged procedure.
- TCPIP.DATA information
- OROUTED cataloged procedure
- MVS system log
- OE RouteD syslogd or the data set specified on //STDOUT DD statement. See “Where to Send OE RouteD Trace Output” on page 149.
- The OE RouteD gateways file or data set
- The file (or data set) pointed to by RESOLVER_CONFIG environment variable

More documentation that might be needed is discussed in the analysis section.

Analysis

Refer to *OS/390 TCP/IP OpenEdition Configuration Guide* for TCP/IP configuration-related problems.

Diagnostic steps for OE RouteD connection problems:

1. If starting from a cataloged procedure, make sure you are using the appropriate cataloged OE application for OE RouteD. Verify the correctness of the data set references.
2. Make sure that OE RouteD is configured correctly in the PROFILE.TCPIP information. UDP port 520 must be reserved for OE RouteD.
3. Make sure that OE RouteD is configured correctly in the services file or data set. Verify that the assigned port number and service name are correct.
4. For network connectivity problems, refer to “Diagnosing Network Connectivity Problems” on page 27.

OE PING Failures

If an oping command fails on a system where OE RouteD is being used, a client is unable to get a response to a oping command. Before doing anything else, run `onetstat -g`. This should tell you which gateways are configured. If no gateways are configured, oping will not work.

Documentation

The following documentation should be available for initial diagnosis of oping failures:

- MVS system log
- PROFILE.TCPIP information
- output from `onetstat -g`

More documentation that might be needed is discussed in the analysis section.

Analysis

Table 16 on page 145 shows symptoms of oping failures and describes the steps needed for initial diagnosis of the error.

Table 16. OE RouteD oping Failures

oping Failure	Action Steps
Incorrect response	1. Make sure that the oping command contains a valid destination IP address for the remote host. If the destination IP address is a virtual IP address (VIPA), make sure that VIPA is defined correctly. See the *OS/390 TCP/IP OpenEdition Configuration Guide* for more information on rules and recommendations when defining a virtual IP address. 2. Make sure that the router providing the RIP support involved in the oping transaction is active and is running with a correct level of OE RouteD or some application that provides RIP support. If the destination router is not running RIP, make sure that static routes are defined from the destination router to the local host. 3. If the oping command was issued from a client on an OS/390 server, issue a `onetstat -g` command to display the routing tables. Verify that the routes and networks are correct as defined in the *hlq*.PROFILE.TCPIP data set and the /etc/gateways file. 4. If the oping command was issued from a client on a host running the OS/2 or DOS operating system, issue a onetstat -r command to display the routing tables. Verify that the routes and networks are correct as defined in the TCP/IP configuration and the /etc/gateways file of TCP/IP. From the OS/2 operating system, issue **ICAT** and select the Routing Information menu. From DOS, issue **IFCONFIG inet ip show** to display the TCP/IP configured routes. If there are any problems with the routes or networks, refer to OS/2 or DOS documentation on correcting onetstat problems. 5. If there are no problems with the routes and networks, check for broken or poorly-connected cables between the client and the remote host. This includes checking the internet interfaces (such as token ring and Ethernet) on the OE RouteD server.

Incorrect Output

Problems with incorrect output are reported when the data sent to the client is not seen in its expected form. This could be incorrect TCP/IP output, RIP commands that are not valid, incorrect RIP broadcasting information, incorrect updates of routing tables, or truncation of packets.

Documentation

The following documentation should be available for initial diagnosis of incorrect output:

- OROUTED cataloged procedure
- MVS system log
- OE RouteD syslogd or the data set specified on //STDOUT DD statement. See “Where to Send OE RouteD Trace Output” on page 149.
- PROFILE.TCPIP information

Analysis

Table 17 shows symptoms of incorrect output and describes the actions needed for initial diagnosis of the error.

Table 17. OE RouteD Incorrect Output

Incorrect Output	Action Steps
TCP/IP Incorrect Output	1. If the TCP/IP SYSERROR shows a message, refer to *OS/390 TCP/IP OpenEdition Messages and Codes* and follow the directions for system programmer response for the message. 2. In the event of TCP/IP loops or hangs, refer to Chapter 3, "Diagnosing Abends, Loops, and Hangs" on page 21.
ORouteD Incorrect Output	If the ORouteD console shows a message, refer to *OS/390 TCP/IP OpenEdition Messages and Codes* and follow the directions for system programmer response for the message.

Session Outages

Session outages are reported as an unexpected abend or termination of a TCP/IP connection.

Documentation

The following documentation should be available for initial diagnosis of session outages:

- OROUTED cataloged procedure
- MVS system log
- OE RouteD syslogd or the data set specified on //STDOUT DD statement. See "Where to Send OE RouteD Trace Output" on page 149.
- TCPIP CTRACE (refer to Chapter 5, " TCP/IP MVS Component Traces and IPCS Support" on page 53)

Analysis

Table 18 shows symptoms of session outages and describes the steps needed for initial diagnosis of the error.

Table 18. OE RouteD Session Outages

Session Outage	Action Steps
TCP/IP session outage	1. If the TCP/IP console shows a TCP/IP error message, refer to *OS/390 TCP/IP OpenEdition Messages and Codes* and follow the directions for system programmer response for the message. 2. Information in the external CTRACE should contain information about the error. 3. In the event of an TCP/IP abend, refer to Chapter 3, Diagnosing Abends, Loops, and Hangs.
OE RouteD session outage	If an OE RouteD error message is displayed, refer to *OS/390 TCP/IP OpenEdition Messages and Codes* and follow the directions for system programmer response for the message.

OE RouteD Traces and Debug Information

There are many MVS TCP/IP traces that can be useful in identifying the cause of OE RouteD problems. This section discusses the OE RouteD traces. OE RouteD traces and debug requests can be started from the OE shell, or they can be started from an MVS cataloged procedure. This section discusses both methods.

Note: OE RouteD trace output is sent to syslogd unless you specify otherwise. See “Where to Send OE RouteD Trace Output” on page 149.

Figure 36 shows a sample OE RouteD environment.

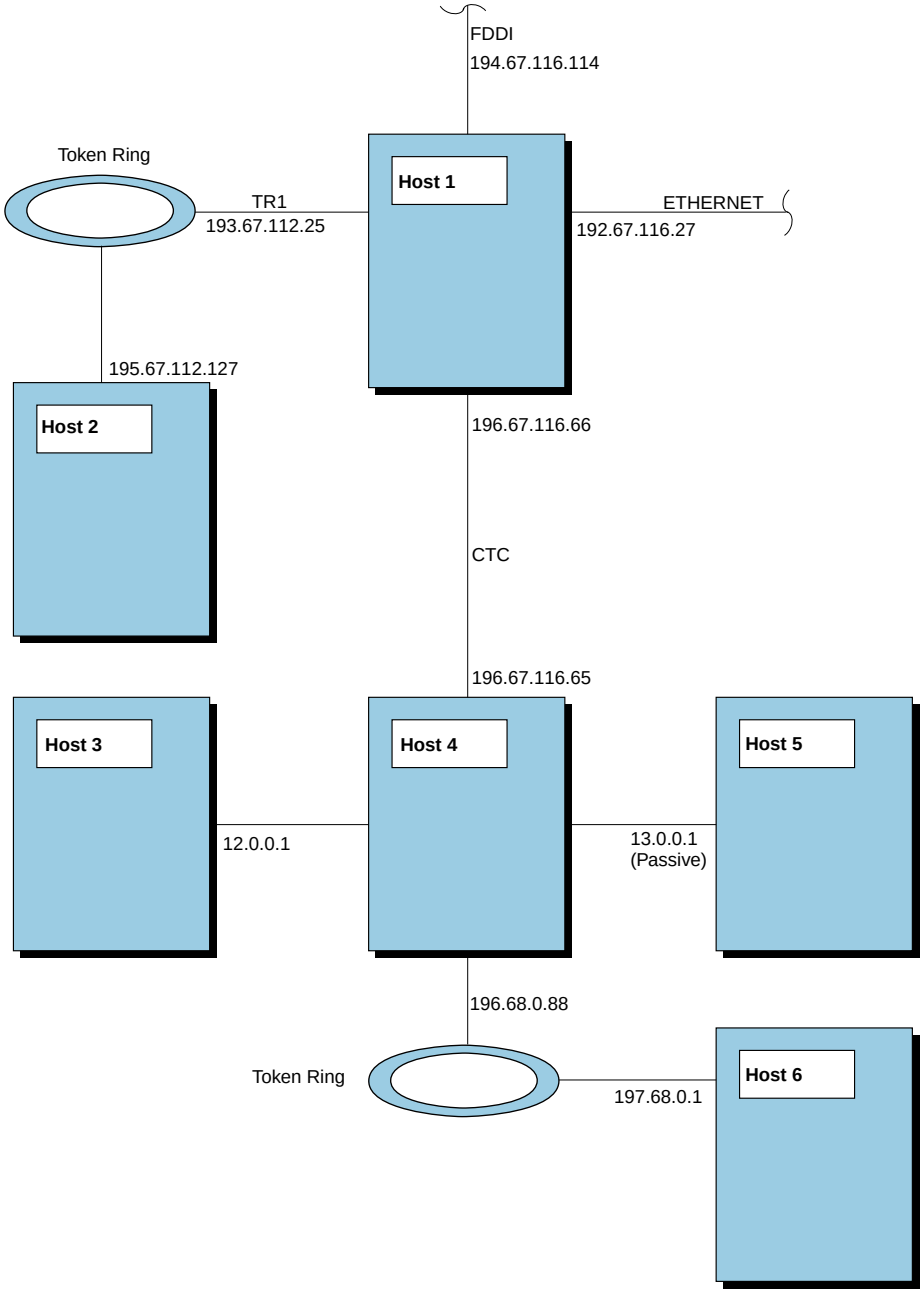


Figure 36. Sample OE RouteD Environment

Starting OE RouteD Traces from the OE Shell

Note: The following command works only if you are superuser.

From TSO, issue the OMVS command, which puts you at a UNIX-like prompt. Then type the `orouted` command followed by one or more of the following parameters.

Note: OE RouteD traces can be dynamically started and stopped using the MODIFY command. For more information, see the *OS/390 TCP/IP OpenEdition Configuration Guide*.

Parameter	Description
-t	Activates tracing of actions by the OE RouteD server.
-t -t	Activates tracing of actions and packets sent or received.
-t -t -t	Activates tracing of actions, packets sent or received, and packet history. Circular trace buffers are used for each interface to record the history of all packets traced. This history is included in the trace output whenever an interface becomes inactive.
-t -t -t -t	Activates tracing of actions, packets sent or received, packet history, and packet contents. The RIP network routing information is included in the trace output.
-dp	Activates tracing of packets to and from adjacent routers and received and broadcasted RIP network routing tables. Packets are shown in data format in the trace output. The information is written to STDOUT.
-ep	Sends print statement to STDOUT and STDERR. **Note:** Do not run OE RouteD in the background with this parameter.
-d	Enables internal debug information, which consists of internal code points. The information is written to STDOUT. Use this parameter only if IBM service requests the information.

Notes:

1. The parameters described here are only those that activate tracing. Refer to *OS/390 TCP/IP OpenEdition User's Guide* for more information about all of the OE RouteD parameters.
2. To run orouted in the background, add an ampersand (&) to the command, as in the command `orouted&`
3. You can enter more than one parameter, with a space after each parameter; for example, `orouted -t -t -t -dp`

Starting OE RouteD Traces from MVS

The OE RouteD traces are controlled by parameters on `PARM=` in the PROC statement of the OROUTED cataloged procedure.

For example:

```
//OROUTED  EXEC PGM=BPXBATCH,REGION=30M,TIME=NOLIMIT
      PARM='PGM /usr/lpp/bin/orouted /-t -t -dp'
```

The OE RouteD parameters that control tracing are:

Note: OE RouteD traces can be dynamically started and stopped using the MODIFY command. For more information, see the *OS/390 TCP/IP OpenEdition Configuration Guide*.

Parameter	Description
-t	Activates tracing of actions by the OE RouteD server.
-t -t	Activates tracing of actions and packets sent or received.
-t -t -t	Activates tracing of actions, packets sent or received, and packet history. Circular trace buffers are used for each interface to record the history of all packets traced. This history is included in the trace output whenever an interface becomes inactive.
-t -t -t -t	Activates tracing of actions, packets sent or received, packet history, and packet contents. The RIP network routing information is included in the trace output.
-dp	Activates tracing of packets to and from adjacent routers and received and broadcasted RIP network routing tables. Packets are shown in data format in the trace output. The information is written to STDOUT.
-ep	Sends print statement to STDOUT and STDERR. **Note:** Do not run OE RouteD in the background with this parameter.
-d	Enables internal debug information, which consists of internal code points. The information is written to STDOUT. Use this parameter only if IBM service requests the information.

Notes:

1. A slash (/) precedes the first parameter.
2. Each parameter is separated by a blank.
3. Parameters can be specified in mixed case.
4. The parameters described here are only those that activate tracing. Refer to *OS/390 TCP/IP OpenEdition User's Guide* for more information about all of the OE RouteD parameters.

Where to Send OE RouteD Trace Output

Normally, trace output is sent to syslogd, which includes a great deal of information about many TCP/IP components, not just OE RouteD. If you want to see only OE RouteD output, use the `-ep` parameter when you start a trace. This will send output to

1. syslogd

AND

2. to a data set that is specified on //STDOUT DD statement in the OROUTED cataloged procedure

OR

3. to the shell session (if you are running in the OE shell).

Note: The `-ep` parameter cannot be altered using the MODIFY command.

Stopping OE RouteD

You can stop OE RouteD in several ways:

- From an OE shell superuser ID, issue the `kill` command to the process ID (PID) associated with OE RouteD.

  To find the PID, use one of the following methods:

  – Use `D OMVS,U=USERID` (This is the USERID that started OE RouteD from the shell.

  – Use the `ps -ef` command from the shell.

  – Write down the PID when you start OE RouteD.

  – Use a shell command pipeline such as

  ```
  kill $(ps | awk '/orouted/' {print $1}')
  ```

  (In this case, you don't even need to know the PID.)

- From MVS, issue the MODIFY command, specifying the parm `-k`. For example, the following commands would stop an OE RouteD server started with a procedure named ROUTED.

  ```
  MODIFY ROUTED,PARMS=-k
  ```

  For more information on this command, see *OS/390 TCP/IP OpenEdition Configuration Guide*.

- From MVS, issue `P` *`procname`* where *procname* is the procedure name used to start OE RouteD. If OE RouteD was started from the OE shell, the procname is *useridX*, where*X* is the sequence number set by the system. To determine the sequence number, issue `/d a,l` from any MVS console, to see the programs running.

If you want to see the OE application name in the output, use `D OMVS,U=USERID`, described above.

Changing Trace Levels with MODIFY

Whether you start OE RouteD from OMVS or MVS, you can use the MVS MODIFY to change command trace levels. The MODIFY syntax is

MODIFY procname,parms=*parm*

A modify format for MVS might look like this:

```
MODIFY ROUTED,PARMS=-t -t -t
```

If you had started from the OE shell, you would use something like this:

```
MODIFY useridX,PARMS=-t -t -t
```

where X is the sequence number for the OE RouteD job. To determine the sequence number, refer to “Stopping OE RouteD.”

For more information on using the MODIFY command, see *OS/390 TCP/IP OpenEdition Configuration Guide*, SC31-8304-00.

Trace Example and Explanation

Figure 37 shows an example of an OE RouteD trace that was generated using `-ep -t -t -t` parameters. “Key to Figure” on page 155 explains the trace.

```
EZZ4990I OE RouteD server initializing. Level 1.22
EZZ4980I Using catalog '/usr/lib/nls/msg/C/routed.cat' for OE RouteD messages.
1 EZZ4828I Input parameter(s): -ep -t -t -t
EZZ4985I Setting High Level Qualifier (HLQ) to 'TCPV33'
EZZ4988I OE RouteD established affinity with 'TCPV33A'
EZZ4929I Port 520 assigned to route
EZZ4932I ************************************************
EZZ4850I * Processing interface TR2
EZZ4932I ************************************************
EZZ4948I This interface is not point-to-point
EZZ4943I Adding network route for interface
EZZ4882I Fri Mar  7 16:52:15 1997:
EZZ4883I ADD destination 9.0.0.0, router 9.67.113.26, metric 1
flags UP state INTERFACE!CHANGED!INTERNAL!SUBNET timer 0
EZZ4943I Adding subnetwork route for interface
EZZ4883I ADD destination 9.67.113.0, router 9.67.113.26, metric 1
flags UP state INTERFACE!CHANGED!SUBNET timer 0
EZZ4932I ************************************************
EZZ4850I * Processing interface CTCD0E
EZZ4932I ************************************************
EZZ4940I Point-to-point interface, using dstaddr
EZZ4943I Adding subnetwork route for interface
EZZ4883I ADD destination 9.67.116.0, router 9.67.116.38, metric 1
flags UP state INTERFACE!CHANGED!SUBNET timer 0
EZZ4943I Adding host route for interface
EZZ4883I ADD destination 9.67.116.37, router 9.67.116.38, metric 1
flags UP!HOST state INTERFACE!CHANGED!SUBNET timer 0
EZZ4932I ************************************************
EZZ4850I * Processing interface VIPA1
EZZ4932I ************************************************
EZZ4965I Virtual interface
EZZ4948I This interface is not point-to-point
EZZ4943I Adding network route for interface
EZZ4883I ADD destination 162.33.0.0, router 162.33.33.33, metric 1
flags UP state INTERFACE!CHANGED!INTERNAL!SUBNET timer 0
EZZ4943I Adding subnetwork route for interface
EZZ4883I ADD destination 162.33.33.0, router 162.33.33.33, metric 1
flags UP state INTERFACE!CHANGED!INTERNAL!SUBNET timer 0
EZZ4943I Adding host route for interface
EZZ4883I ADD destination 162.33.33.33, router 162.33.33.33, metric 1
flags UP!HOST state INTERFACE!CHANGED!INTERNAL!SUBNET timer 0
```

Figure 37 (Part 1 of 4). Example of an OE RouteD Trace

```
EZZ4932I ************************************************
EZZ4850I * Processing interface VIPA2
EZZ4932I ************************************************
EZZ4965I Virtual interface
EZZ4948I This interface is not point-to-point
EZZ4943I Adding network route for interface
EZZ4883I ADD destination 162.44.0.0, router 162.44.44.44, metric 1
flags UP state INTERFACE!CHANGED!INTERNAL!SUBNET timer 0
EZZ4943I Adding subnetwork route for interface
EZZ4883I ADD destination 162.44.44.0, router 162.44.44.44, metric 1
flags UP state INTERFACE!CHANGED!INTERNAL!SUBNET timer 0
EZZ4943I Adding host route for interface
EZZ4883I ADD destination 162.44.44.44, router 162.44.44.44, metric 1
flags UP!HOST state INTERFACE!CHANGED!INTERNAL!SUBNET timer 0
EZZ4932I ************************************************
EZZ4934I * Opening GATEWAYS file (/etc/gateways.tcpv33a)
EZZ4932I ************************************************
EZZ4925I Start of GATEWAYS processing:
EZZ4936I Adding passive net route 9.67.77.0 via gateway 9.67.113.26, metric 2
EZZ4883I ADD destination 9.67.77.0, router 9.67.113.26, metric 2
flags UP state PASSIVE!INTERFACE!CHANGED!SUBNET timer 0
EZZ4945I ifwithnet:  compare with TR2
EZZ4947I netmatch 9.67.113.1 and 9.67.113.26
EZZ4936I Adding passive net route 0.0.0.0 via gateway 9.67.113.1, metric 1
EZZ4883I ADD destination 0.0.0.0, router 9.67.113.1, metric 1
flags UP!GATEWAY state PASSIVE!CHANGED timer 0
EZZ4945I ifwithnet:  compare with TR2
EZZ4945I ifwithnet:  compare with CTCD0E
EZZ4947I netmatch 9.67.116.37 and 9.67.116.38
EZZ4936I Adding passive host route 9.67.116.99 via gateway 9.67.116.37, metric 1
EZZ4883I ADD destination 9.67.116.99, router 9.67.116.37, metric 1
flags UP!GATEWAY!HOST state PASSIVE!CHANGED timer 0
EZZ4926I End of GATEWAYS processing
EZZ4849I OE RouteD Server started
2  EZZ4899I     REQUEST to 9.67.113.255 -> 0:
   EZZ4899I     REQUEST to 9.67.116.37 -> 0:
3  EZZ4829I Waiting for incoming packets
   EZZ4899I     REQUEST from 9.67.113.26 -> 520:
   EZZ4958I supply 9.67.113.26 -> 520 via null-interface
   EZZ4899I     RESPONSE to 9.67.113.26 -> 520:
   EZZ4829I Waiting for incoming packets
   EZZ4899I     RESPONSE from 9.67.113.26 -> 520:
   EZZ4829I Waiting for incoming packets
   EZZ4899I     REQUEST from 9.67.116.37 -> 520:
   EZZ4958I supply 9.67.116.37 -> 520 via null-interface
   EZZ4899I     RESPONSE to 9.67.116.37 -> 520:
   EZZ4829I Waiting for incoming packets
   EZZ4829I Waiting for incoming packets
4  EZZ4957I 30 second timer expired (poll interfaces for status)
   EZZ4957I 30 second timer expired (broadcast)
   EZZ4958I supply 9.67.113.255 -> 0 via TR2
```

Figure 37 (Part 2 of 4). Example of an OE RouteD Trace

```
EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
EZZ4958I supply 9.67.116.37 -> 0 via CTCD0E
EZZ4899I     RESPONSE to 9.67.116.37 -> 0:
EZZ4829I Waiting for incoming packets
EZZ4899I     RESPONSE from 9.67.113.26 -> 520:
EZZ4829I Waiting for incoming packets
EZZ4899I     RESPONSE from 9.67.116.37 -> 520:
EZZ4882I Fri Mar  7 16:53:02 1997:
EZZ4883I ADD destination 9.67.88.0, router 9.67.116.37, metric 2
flags UP!GATEWAY state CHANGED!SUBNET timer 0
EZZ4861I Send dynamic update
EZZ4958I supply 9.67.113.255 -> 0 via TR2
EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
EZZ4890I toall:  requested to skip interface CTCD0E
EZZ4863I Inhibit dynamic update for 2.03 seconds
EZZ4899I     RESPONSE from 9.67.113.26 -> 520:
EZZ4829I Waiting for incoming packets
EZZ4829I Waiting for incoming packets
EZZ4957I 60 second timer expired (rescan kernel for interfaces)
EZZ4957I 30 second timer expired (poll interfaces for status)
EZZ4829I Waiting for incoming packets
.
.
.
5   EZZ4828I Input parameter(s): -T -T -T -T
    EZZ4871I Tracing packet contents enabled Fri Mar  7 16:53:53 1997
    EZZ4899I     RESPONSE from 9.67.116.37 -> 520:
    EZZ4902I destination 9.67.88.0 metric 1
    EZZ4829I Waiting for incoming packets
    EZZ4829I Waiting for incoming packets
    EZZ4957I 60 second timer expired (rescan kernel for interfaces)
    EZZ4957I 30 second timer expired (poll interfaces for status)
    EZZ4957I 30 second timer expired (broadcast)
    EZZ4958I supply 9.67.113.255 -> 0 via TR2
    EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
    EZZ4902I destination 162.33.0.0 metric 3
    EZZ4902I destination 162.44.0.0 metric 3
    EZZ4902I destination 9.67.77.0 metric 4
    EZZ4902I destination 9.67.116.0 metric 3
    EZZ4902I destination 9.67.88.0 metric 4
    EZZ4958I supply 9.67.116.37 -> 0 via CTCD0E
    EZZ4899I     RESPONSE to 9.67.116.37 -> 0:
    EZZ4902I destination 162.33.0.0 metric 4
    EZZ4902I destination 162.44.0.0 metric 4
    EZZ4902I destination 9.67.77.0 metric 5
    EZZ4902I destination 9.67.113.0 metric 4
    EZZ4829I Waiting for incoming packets
    EZZ4899I     RESPONSE from 9.67.113.26 -> 520:
    EZZ4902I destination 162.33.0.0 metric 3
    EZZ4902I destination 162.44.0.0 metric 3
    EZZ4902I destination 9.67.77.0 metric 4
    EZZ4902I destination 9.67.116.0 metric 3
```

Figure 37 (Part 3 of 4). Example of an OE RouteD Trace

```
  EZZ4902I destination 9.67.88.0 metric 4
  EZZ4829I Waiting for incoming packets
  EZZ4899I     RESPONSE from 9.67.116.37 -> 520:
  EZZ4902I destination 9.67.88.0 metric 1
  EZZ4829I Waiting for incoming packets
  EZZ4829I Waiting for incoming packets
  EZZ4957I 30 second timer expired (poll interfaces for status)
  EZZ4949I Interface TR2 not up
6 EZZ4891I *** Packet history for interface TR2 ***
  EZZ4898I Output trace:
  EZZ4899I     REQUEST to 9.67.113.255 -> 0:
  EZZ4903I request for full tables
  EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
  EZZ4902I destination 162.33.0.0 metric 3
  EZZ4902I destination 162.44.0.0 metric 3
  EZZ4902I destination 9.67.77.0 metric 4
  EZZ4902I destination 9.67.116.0 metric 3
  EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
  EZZ4902I destination 9.67.88.0 metric 4
  EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
  EZZ4902I destination 162.33.0.0 metric 3
  EZZ4902I destination 162.44.0.0 metric 3
  EZZ4902I destination 9.67.77.0 metric 4
  EZZ4902I destination 9.67.116.0 metric 3
  EZZ4902I destination 9.67.88.0 metric 4
  EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
  EZZ4902I destination 162.33.0.0 metric 3
  EZZ4902I destination 162.44.0.0 metric 3
  EZZ4902I destination 9.67.77.0 metric 4
  EZZ4902I destination 9.67.116.0 metric 3
  EZZ4902I destination 9.67.88.0 metric 4
  EZZ4899I     RESPONSE to 9.67.113.255 -> 0:
  EZZ4902I destination 162.33.0.0 metric 3
  EZZ4902I destination 162.44.0.0 metric 3
  EZZ4902I destination 9.67.77.0 metric 4
  EZZ4902I destination 9.67.116.0 metric 3
  EZZ4902I destination 9.67.88.0 metric 4
  EZZ4898I Input trace:
  EZZ4899I     REQUEST from 9.67.113.26 -> 520:
  EZZ4903I request for full tables
  EZZ4899I     RESPONSE from 9.67.113.26 -> 520:
7 EZZ4902I destination 162.33.0.0 metric 1
  EZZ4902I destination 162.44.0.0 metric 1
  EZZ4902I destination 9.67.77.0 metric 2
  EZZ4902I destination 9.67.116.0 metric 1
  EZZ4899I     RESPONSE from 9.67.113.26 -> 520:
  EZZ4902I destination 162.33.0.0 metric 3
  EZZ4902I destination 162.44.0.0 metric 3
  EZZ4902I destination 9.67.77.0 metric 4
  EZZ4902I destination 9.67.116.0 metric 3
```

Figure 37 (Part 4 of 4). Example of an OE RouteD Trace

Key to Figure

1. EZZ4958I shows the parameters used for this trace.
2. EZZ4899I describes the type of packet being sent or received
3. EZZ4829I means the system is waiting for incoming packets
4. EZZ4957I shows timer information
5. At this point, trace paramaters were changed using a MODIFY command
6. EZZ4891I shows packet history
7. EZZ4902I provides packet detail for each route contained in an RIP packet

Appendixes

Appendix A. A Note on Search Paths

The TCPIP.DATA statement DATASETPREFIX is retrieved by the OpenEdition service services _ipdspx(); TCPIPjobname is retrieved by OE _iptcpn(). The search path for these services starts with

```
ENVIRONMENT VARIABLE "RESOLVER_CONFIG=dataset/file"
/etc/resolv.conf
```

before entering the normal TCPIP.DATA search path. Remember _ipdspx() and _iptcpn() stop searching at the first file they find.

Consider the following scenario:

1. ENVIRONMENT VARIABLE “RESOLVER_CONFIG=dataset/file”isn't set.
2. /etc/resolv.conf exists but doesn't contain either a DATASETPREFIX or TCPIPjobname statement.
3. TCPIP.TCPIP.DATA exists and has a TCPIPjobname of TCPIPA.

When the SNMP agent is started and calls _iptcpn(), it will be returned a null string, because the search stops with /etc/resolv.conf. This problem can be resolved through one of the following means:

- Make sure the “none”default DATASETPREFIX isn't being used to locate the configuration files, or
- Only one INET PFS is started.

For more information on search paths, refer to *OS/390 TCP/IP OpenEdition Configuration Guide*.

Appendix B. What We Mean by hlq

hlq indicates a high-level qualifier determined by the standard TCP/IP conventions. OS/390 TCP/IP OpenEdition is distributed with a default *hlq* of **TCPIP**. This hlq is a hard-coded character string within OS/390 TCP/IP OpenEdition. It may be used as the high-level qualifier for data set names dynamically allocated by TCPIP.

You can use the default hlq, or you can change it as described in “Dynamic Data Set Allocation” in *OS/390 TCP/IP OpenEdition Configuration Guide*, SC31-8304-00.

Appendix C. Description of Syslog Daemon (syslogd)

Syslog daemon (syslogd) is a server process that has to be started as one of the first processes in your OpenEdition environment. Other servers and stack components use syslogd for logging purposes and can also send trace information to syslogd.

Each application actives and deactivates traces in a slightly different manner. Refer to the chapter on the individual application for details. OS/390 TCP/IP OpenEdition components use the local1 and daemon facility names (see Figure 38 on page 165).

Servers on the local system use AF_UNIX sockets to communicate with syslogd; remote servers use the AF_INET socket. If syslogd is not started, application log data appears on the MVS console.

Format

syslogd [–**f** *conffile***] [**–**m** *markinterval***] [**–**p** *logpath***]**

Description

syslogd reads and logs system messages to the console, log files, other machines, or users as specified by the configuration file.

The configuration file is read at startup and whenever the hang-up signal (SIGHUP) is received. The syntax of the configuration file is described below.

syslogd stores its process id in file:

`/etc/syslog.pid`

so it may be used to terminate or re-configure the daemon.

Messages are read from the UNIX domain datagram socket and the Internet domain datagram socket. Kernel messages are not logged in OpenEdition MVS.

Options

syslogd recognizes the following options:

–**f** Configuration file name.

–**m** Number of minutes between mark messages.

–**p** Path name for the UNIX datagram socket.

Files

/dev/console Operator console.

/etc/syslog.pid Process id is written here.

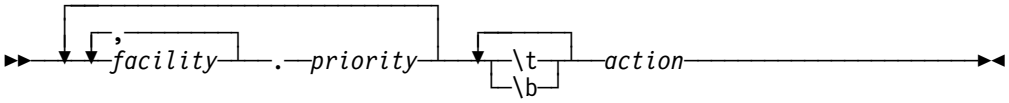
/etc/syslog.conf Default configuration file name.

/dev/log Default log path for UNIX datagram socket.

/usr/sbin/syslogd This is the server.

Configuration Lines

Each line of the configuration file has the following syntax:



where \t is a tab character and \b is a blank character.

The facilities supported are as follows:

kern Message generated by the system.

user Message generated by a process (user).

mail Message generated by mail system.

news Message generated by news system.

uucp Message generated by UUCP system.

daemon Message generated by system daemon.

auth/authpriv Message generated by authorization daemon.

cron Message generated by the clock daemon.

lpr Message generated by printer system.

local0 Reserved for local use.

local1 Reserved for local use.

local2 Reserved for local use.

local3 Reserved for local use.

local4 Reserved for local use.

local5 Reserved for local use.

local6 Reserved for local use.

local7 Reserved for local use.

mark Used for logging MARK messages.

The priorities supported are as follows:

emerg/panic A panic condition was reported to all processes.

alert A condition that should be corrected immediately.

crit A critical condition.

err(or) An error message.

warn(ing) A warning message.

notice A condition requiring special handling.

info A general information message.

debug A message useful for debugging programs.

none Do not log any messages for the facility.

The actions supported are as follows. Be sure to use lowercase for all filenames, users, and hosts:

/file log message to this file

@host log message to syslog daemon on another host

user1,user2,... log message to the list of users

***** log message to all logged-in users

Note: Comments can be added to the configuration file by placing the # character in column 1 of the comment line. Everything following the # character will be treated as a comment.

Syslog.conf Examples

Here is a sample of a syslog.conf file:

```
# Examples:
# log all daemon messages to the operator console.
# Note:  this may generate a lot of master console
# output if traces are currently active in several
# TCP/IP components
daemon.*              /dev/console
#
# All debug messages (and above priority
# messages) from telnet go to telnet.debug
local1.debug          /tmp/syslogd/telnet.debug
#
# All debug messages
# (and above priority messages) go to
# server.debug
daemon.debug          /tmp/syslogd/server.debug
#
# log mail messages at info and above to /tmp/user.info
mail.info /tmp/user.info
#
#
#user1 and user2 should get all emergency messages
*.alert user1,user2
#
# log all messages (except mail) to /tmp/all.except.mail
*.*;mail.none /tmp/all.except.mail
#
# log clock and printer err(+) messages to yourhost
#
cron,lpr.err @yourhost
```

Figure 38. Example of a syslog.conf File

Starting Syslogd

If you want ITRACE messages from TCP/IP initialization written to syslogd and do not want trace messages from TCP/IP or inetd written to the master console, you must start syslogd before TCP/IP and inetd:

1. Start syslogd
2. Start TCP/IP
3. Start inted

For special considerations on remote syslogd servers, see “Usage Notes.”

Usage Notes

- **syslogd** can only be started by a superuser.
- **syslogd** can be terminated using the SIGTERM signal.
- If you want syslod in an MVS image to receive log data from remote syslogd servers, then UDP port 514 must be reserved for OMVS in your OS/390 TCP/IP OpenEdition PROFILE data set.

  ```
  PORT
     . . .
      514 UDP OMVS              ; OE SyslogD Server
     . . .
  ```

- If there is no TCP/IP (AF_INET) transport connected to OE when syslogd starts, syslogd cannot bind to port 514, so it cannot receive log data from remote syslogd servers. If you want data from remove syslogd servers, you must stop and restart syslogd after TCP/IP has been initialized.

  Note: If trace messages were being written at the time syslogd was stopped and restarted, all subsequent trace messages will be lost.

- If the configuration file cannot be opened, the following configuration lines are used as a default:

```
*.err    /dev/console
*.panic  *
```

- Configuration file errors are written to the operator console because initialization is not complete until the entire configuration file has been read.
- Facility mark is not affected by the *.priority usage.
- Minimum mark interval is 30 seconds.

Exit Values

If an invalid argument is entered, or if an invalid number of arguments is entered, then a return code of 1 is set and syslogd exits. All other cases in which syslogd exits cause a return code of zero to be set.

Related Information

To terminate (“kill”) syslogd, send a SIGTERM (terminate). A SIGHUP (hangup) will cause it to reread its configuration file. To send a signal to syslogd, issue one of the following:

- `KILL -s SIGHUP <PID>` or
- `KILL -s SIGTERM <PID>`

You can obtain the PID by reading the PID file that syslogd opens in the /etc directory.

For more information about syslogd, refer to *Accessing OS/390 OpenEdition MVS from the Internet* (SG24-4721).

Appendix D. How to Read a Syntax Diagram

The syntax diagram shows you how to specify a command so that the operating system can correctly interpret what you type. Read the syntax diagram from left to right and from top to bottom, following the horizontal line (the main path).

Symbols and Punctuation

The following symbols are used in syntax diagrams:

Symbol	Description
►►	Marks the beginning of the command syntax.
►	Indicates that the command syntax is continued.
|	Marks the beginning and end of a fragment or part of the command syntax.
►◄	Marks the end of the command syntax.

You must include all punctuation such as colons, semicolons, commas, quotation marks, and minus signs that are shown in the syntax diagram.

Parameters

The following types of parameters are used in syntax diagrams.

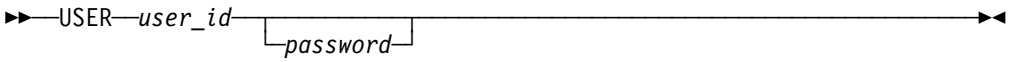
Parameter	Description
Required	Required parameters are displayed on the main path.
Optional	Optional parameters are displayed below the main path.
Default	Default parameters are displayed above the main path.

Parameters are classified as keywords or variables. Keywords are displayed in uppercase letters and can be entered in uppercase or lowercase. For example, a command name is a keyword.

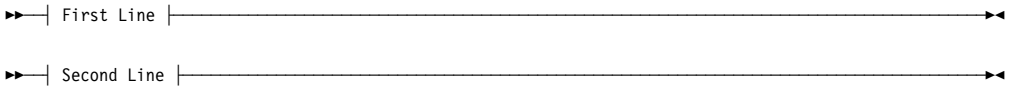
Variables are italicized, appear in lowercase letters, and represent names or values you supply. For example, a data set is a variable.

Syntax Examples

In the following example, the USER command is a keyword. The required variable parameter is *user_id*, and the optional variable parameter is *password*. Replace the variable parameters with your own values.



Longer than one line: If a diagram is longer than one line, the first line ends with a single arrowhead and the second line begins with a single arrowhead.



Required operands: Required operands and values appear on the main path line.

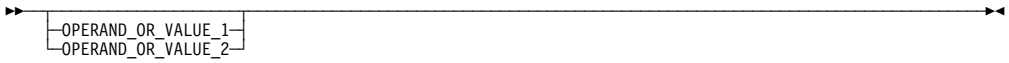
You must code required operands and values.

Choose one required item from a stack: If there is more than one mutually exclusive required operand or value to choose from, they are stacked vertically in alphanumeric order.

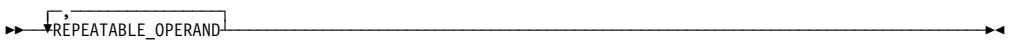
Optional values: Optional operands and values appear below the main path line.

You can choose not to code optional operands and values.

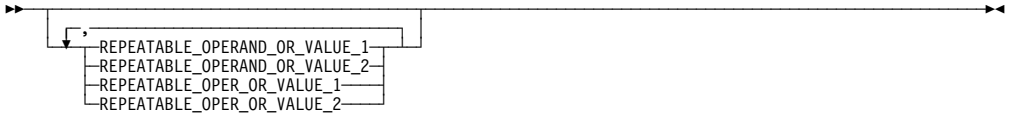
Choose one optional operand from a stack: If there is more than one mutually exclusive optional operand or value to choose from, they are stacked vertically in alphanumeric order below the main path line.



Repeating an operand: An arrow returning to the left above an operand or value on the main path line means that the operand or value can be repeated. The commad means that each operand or value must be separated from the next by a comma.



Selecting more than one operand: An arrow returning to the left above a group of operands or values means more than one can be selected, or a single one can be repeated.



If an operand or value can be abbreviated, the abbreviation is described in the text associated with the syntax diagram.

Nonalphanumeric characters: If a diagram shows a character that is not alphanumeric (such as parentheses, periods, commas, and equal signs), you must code the character as part of the syntax. In this example, you must code OPERAND=(001,0.001).

Blank spaces in syntax diagrams: If a diagram shows a blank space, you must code the blank space as part of the syntax. In this example, you must code OPERAND=(001 FIXED).

Default operands: Default operands and values appear above the main path line. TCP/IP uses the default if you omit the operand entirely.

Variables: A word in all lowercase italics is a *variable*. Where you see a variable in the syntax, you must replace it with one of its allowable names or values, as defined in the text.

Syntax fragments: Some diagrams contain syntax fragments, which serve to break up diagrams that are too long, too complex, or too repetitious. Syntax fragment names are in mixed case and are shown in the diagram and in the heading of the fragment. The fragment is placed below the main diagram.

Syntax Fragment:

Appendix E. Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make them available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
P.O. Box 12195
3039 Cornwallis Road
Research Triangle Park, NC 27709-2195
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement.

This document is not intended for production use and is furnished as is without any warranty of any kind, and all warranties are hereby disclaimed including the warranties of merchantability and fitness for a particular purpose.

IBM is required to include the following statements in order to distribute portions of this document and the software described herein to which contributions have been made by The University of California.

Portions herein © Copyright 1979, 1980, 1983, 1986, Regents of the University of California. Reproduced by permission. Portions herein were developed at the Electrical Engineering and Computer Sciences Department at the Berkeley campus of the University of California under the auspices of the Regents of the University of California.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

ACF/VTAM
AD/Cycle
AIX
AIX/ESA
BookManager
C/370
CICS
DB2
DFSMS
DFSMS/MVS
ESCON
ES/9000
EtherStreamer
Extended Services
GDDM
Hardware Configuration Definition
IBM
LANStreamer
Library Reader
MVS/ESA
MVS/SP
MVS/XA
NetView
OpenEdition
OS/2
OS/390
PS/2
RACF
RISC System/6000
RS/6000
SAA
System/370
System/390
VTAM
3090

The following terms are trademarks of other companies:

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Bibliography

This bibliography lists the publications for IBM TCP/IP products.

IBM TCP/IP Publications

The following sections describe the books associated with IBM TCP/IP products.

OS/390 TCP/IP OpenEdition Publications

- *OS/390 TCP/IP OpenEdition Configuration Guide*, SC31-8304-00.

  This book is for people who want to configure, customize, administer, and maintain OS/390 TCP/IP OpenEdition. Familiarity with MVS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended.

- *OS/390 TCP/IP OpenEdition Diagnosis Guide*, SC31-8492-00.

  This book explains how to diagnose TCP/IP problems and how to determine whether a specific problem is in the OS/390 TCP/IP OpenEdition product code. It explains how to gather information for and describe problems to the IBM Software Support Center.

- *OS/390 TCP/IP OpenEdition Messages and Codes*, SC31-8307-00.

  This book explains the informational and error messages issued by OS/390 TCP/IP OpenEdition. It can help users, operators, or system programmers to diagnose and fix problems identified by error messages.

- *OS/390 TCP/IP OpenEdition Planning and Release Guide*, SC31-8303-00.

  This book is intended to help you plan for OS/390 TCP/IP OpenEdition whether you are migrating from a previous version or installing TCP/IP for the first time. This book also identifies the suggested and required modifications needed to enable you to use the enhanced functions provided with OS/390 TCP/IP OpenEdition.

- *OS/390 TCP/IP OpenEdition Programmer's Reference*, SC31-8308-00

  This book describes the syntax and semantics of a set of high-level application functions that you can use to program your own applications in a TCP/IP environment. These functions provide support for application facilities, such as user authentication, distributed databases, distributed processing, network management, and device sharing.

  This book is for people who want to use the supplied interfaces while writing application programs that access OS/390 TCP/IP OpenEdition. Familiarity with the MVS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended.

- *OS/390 TCP/IP OpenEdition User's Guide*, GC31-8305-00.

  This book is for people who want to use OS/390 TCP/IP OpenEdition for data communication. Familiarity with MVS operating system and IBM Time Sharing Option (TSO) is recommended.

TCP/IP for MVS Publications

- *TCP/IP Version 3 for OpenEdition MVS: Applications Feature Guide*, SC31-8069-00.

  This book explains how to plan for, install, customize, and use the OpenEdition MVS Applications Feature. The Feature consists of applications and interfaces for direct access to the OpenEdition MVS environment. For example, users of the Feature can use MVS, UNIX, or AIX commands to transfer files, log in to the OpenEdition environment without going through TSO, and run commands remotely. This book also explains how to improve performance and diagnose problems when using the Feature.

- *TCP/IP for MVS: Application Programming Interface Reference*, SC31-7187-02.

  This book describes the syntax and semantics of program source code necessary to write your own application programming interface (API) into TCP/IP. You can use this interface as the communication base for writing your own client or server application. You can also use this book to adapt your existing applications to communicate with each other using sockets over TCP/IP.

- *TCP/IP for MVS: CICS TCP/IP Socket Interface Guide and Reference*, SC31-7131-02.

  This book is for people who want to set up, write application programs for, and diagnose problems with the socket interface for CICS using TCP/IP for MVS.

- *TCP/IP for MVS: Customization and Administration Guide*, SC31-7134-03.

  This book is for people who want to con customize, administer, and maintain TCP/IP for MVS. Familiarity with MVS operating system, TCP/IP protocols,

and IBM Time Sharing Option (TSO) is recommended.

- *TCP/IP for MVS: Diagnosis Guide*, LY43-0105-02.

  This book explains how to diagnose TCP/IP problems and how to determine whether a specific problem is in the IBM TCP/IP for MVS product code. It explains how to gather information for and describe problems to the IBM Software Support Center.

- *TCP/IP for MVS: IMS TCP/IP Application Development Guide and Reference*, SC31-7186-02.

  This book is for programmers who want application programs that use the IMS TCP/IP application development services provided by IBM TCP/IP for MVS.

- *TCP/IP for MVS: Messages and Codes*, SC31-7132-03.

  This book explains the informational a nd error messages issued by IBM TCP/IP for MVS. It can help users, operators, or system programmers to diagnose and fix problems identified by TCP/IP for MVS error messages.

- *TCP/IP for MVS: Network Print Facility*, SC31-8074-03.

  This book is for system programmers an d network administrators who need to prepare their network to route VTAM, JES2, or JES3 printer output to remote printers using TCP/IP for MVS.

- *TCP/IP for MVS: Offloading TCP/IP Processing*, SC31-7133-02.

  This book is for people who want to install and configure the Offload feature on IBM 3172 Model 3 Interconnect Controllers. This book is also for people who want to use and customize the Offload feature of TCP/IP for MVS.

- *TCP/IP for MVS: Planning and Migration Guide*, SC31-7189-01.

  This book is intended to help you plan for TCP/IP for MVS whether you are migrating from a previous version or installing TCP/IP for MVS for the first time. This book also identifies the suggested and required modifications needed to enable you to use the enhanced functions provided with TCP/IP for MVS.

- *TCP/IP: Performance Tuning Guide*, SC31-7188-02.

  This book describes how to improve the performance of your network operations.

- *TCP/IP for MVS: Programmer's Reference*, SC31-7135-02.

  This book describes the syntax and semantics of a set of high-level application functions that you can use to program your own applications in a TCP/IP environment. These functions provide support for application facilities, such as user authentication, distributed databases, distributed processing, network management, and device sharing.

  This book is for people who want to use the supplied interfaces while writing application programs that access TCP/IP for MVS. Familiarity with the MVS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended.

- *TCP/IP for MVS: User's Guide*, SC31-7136-02.

  This book is for people who want to use TCP/IP for MVS for data communication. Familiarity with MVS operating system and IBM Time Sharing Option (TSO) is recommended.

TCP/IP for VM Publications

The following list describes books in the IBM TCP/IP for VM library.

- *IBM TCP/IP Version 2 Release 4 for VM: Messages and Codes*, SC31-6151-03.

  This book is for system programmers who want to diagnose and fix problems identified by TCP/IP for VM error messages.

- *IBM TCP/IP Version 2 Release 4 for VM: Planning and Customization*, SC31-6082-03.

  This book is for system programmers who want to plan and customize the TCP/IP for VM environment.

- *IBM TCP/IP Version 2 Release 4 for VM: Programmer's Reference*, SC31-6084-03.

  This book is for application and system programmers who want to write application programs that use TCP/IP for VM. Application programmers should know the VM operating system.

- *IBM TCP/IP Version 2 Release 4 for VM: User's Guide*, SC31-6081-03.

  This book is for people who want to use TCP/IP for VM for data communication. Familiarity with VM operating system, IBM Command Processor (CP), and IBM Conversational Monitor System (CMS) is recommended.

TCP/IP for OS/2 Publication

IBM TCP/IP Version 3.0 for OS/2: Programmer's Reference, SC31-6077.

This book provides application and system programmers with the information required to write application programs that use TCP/IP for OS/2. Programmers should know the OS/2 operating system.

TCP/IP for DOS Publications

The following list describes books in the IBM TCP/IP for DOS library.

- *IBM TCP/IP Version 2.1.1 for DOS: Command Reference*, SX75-0083.

  This book is for people who use a workstation with TCP/IP for DOS, such as end users and system programmers. The people who use this book should be familiar with DOS and the workstation, understand DOS operating system concepts, and be familiar with the *IBM TCP/IP Version 2.1.1 for DOS: User's Guide*

- *IBM TCP/IP Version 2.1.1 for DOS: Installation and Administration*, SC31-7047.

  This book provides system programmers, network administrars, and workstation users responsible for installing TCP/IP for DOS with the information required to plan and implement the installation of TCP/IP for DOS. The topics include hardware and software requirements, pre-installation system performance considerations, instructions for installing TCP/IP for DOS, instructions for customizing the TCP/IP for DOS environment, and installation examples.

- *IBM TCP/IP Version 2.1.1 for DOS: Programmer's Reference*, SC31-7046.

  This book is for application and system programmers to aid them in writing application programs that use TCP/IP for DOS on a workstation. Application programmers should know the DOS operating system and multitasking operating system concepts. Application programmers should be knowledgeable in the C programming language.

- *IBM TCP/IP Version 2.1.1 for DOS: User's Guide*, SC31-745.

  This book is for people who use a workstation with TCP/IP for DOS, such as end users and system programmers. The people who use this book should be familiar with DOS and the workstation, and also understand DOS operating system concepts.

TCP/IP for AIX (RS/6001, PS/2, RT, 370) Publications

The following list shows books in the TCP/IP for AIX library.

- *AIX Operating System TCP/IP User's Guide*, SC23-2309.
- *AIX PS/2 TCP/IP User's Guide*, SC23-2047.
- *TCP/IP for IBM X-Windows on DOS 2.1*, SC23-2349.

TCP/IP for AS/400 Publications

The following list shows books in the TCP/IP for AS/400 library.

- *IBM AS/400 Communications: TCP/IP Guide*, SC41-9875.
- *IBM AS/400 Communications: User's Guide*, SC21-9601.

Other IBM TCP/IP Publications

The following list shows other available IBM TCP/IP books.

- *IBM Local Area Network Technical Reference*, SC30-3383.
- *IBM TCP/IP for VM and MVS: Diagnosis Guide*, LY43-0013.
- *TCP/IP and National Language Support*, GG24-3840.
- *TCP/IP Introduction*, GC31-6080.
- *TCP/IP Tutorial and Technical Overview*, GG24-3376.

IBM Operating System Publications

The following lists show books about various IBM operating systems.

AIX Publications

- *AIX Communications Concepts and Procedures for IBM RISC System/6001*, GC23-2203.
- *AIX Communications Programming Concepts*, SC23-2206.
- *IBM AIX Operating System Technical Reference, Volume 1*, SC23-2300.
- *IBM AIX Operating System Technical Reference, Volume 2*, SC23-2301.

AS/400 Publications

- *IBM AS/400 CL Reference Manual Volume 1*, SC21-9775.
- *IBM AS/400 CL Reference Manual Volume 2*, SC21-9776.
- *IBM AS/400 CL Reference Manual Volume 3*, SC21-9777.
- *IBM AS/400 CL Reference Manual Volume 4*, SC21-9778.

- *IBM AS/400 CL Reference Manual Volume 5*, SC21-9779.
- *IBM AS/400 Communications: APPN Network User's Guide*, SC21-8188.
- *IBM AS/400 Communications: Programmer's Guide*, SC21-9590.
- *IBM AS/400 Communications: User's Guide*, SC21-9601.
- *IBM AS/400 Device Configuration Guide*, SC21-8106.
- *IBM AS/400 Programming: Command Reference Summary*, SC21-8076.
- *IBM AS/400 Programming: Data Management Guide*, SC21-9658.
- *IBM AS/400 System Operations: Database Coordinator' Guide*, SC21-8086.
- *IBM AS/400 System Operations: Operator's Guide*, SC21-8082.

DOS Publications

- *DOS Getting Started Version 5.00*, SA40-0637.
- *DOS 5.02 Technical Reference*, S16G-4559.
- *DOS/Windows Client Getting Started*, SC09-3001.
- *PC DOS 6.1 Command Reference*, S71G-3634.

MVS Publications

For a complete description of the library for MVS/ESA Version 5, see *OS/390 Information Roadmap*, GC28-1727-02. See also “JES Publications” on page 180.

OS/2 Publications

- *IBM OS/2 Warp Server Up and Running!*, S25H-8004
- *IBM Official Guide to Using OS/2 Warp*, ISBN 1-56884-466-2 (Karla Stagray and Linda S. Rogers; Foster City, CA: An IBM Press Book published by IDG Books Worldwide, Inc., 1995)
- *IBM OS/2 Warp Internet Connection: Your Key to Cruising the Internet and the World Wide Web*, ISBN 1-56884-465-4 (Deborah Morrison; Foster City, CA: An IBM Press Book published by IDG Books Worldwide, Inc., 1995)

OS/390 Publications

- *OS/390 Information Roadmap*, GC28-1727-02

  This book describes the documentation for the specific elements included in OS/390.
- *OS/390 Planning for Installation Release 3*, GC28-1726-02

  This book is intended to help you plan for the installation of OS/390. It describes migration, installation, hardware and software requirements, and coexistence considerations.
- *OS/390 OpenEdition Introduction*, GC28-1889-01.
- *OS/390 OpenEdition Planning*, SC28-1890-02.
- *OS/390 OpenEdition User's Guide*, SC28-1891-02.
- *OS/390 OpenEdition Command Reference*, SC28-1892-02.
- *OS/390 OpenEdition Messages and Codes*, SC28-1908-02.
- *OS/390 Language Environment Programming Guide*, SC28-1939-02.
- *OS/390 Language Environment Programming Reference*, SC28-1940-02.
- *OS/390 OpenEdition Programming: Assembler Callable Services Reference*, SC28-1899-02.
- *OS/390 Open Systems Adapter Support Facility Users's Guide*, SC28-1855.
- *Planning for the System/390 Open Systems Adapter Feature*, GC23-3870.

VM Publications

- *VM/ESA CMS Command Reference Summary*, SX24-5249.
- *VM/ESA CP Planning and Administration for 370*, SC24-5430.
- *VM/ESA CP Programming Services for 370*, SC24-5435.
- *VM/ESA Group Control System Reference for 370*, SC24-5426.
- *VM/ESA: Library Guide and Master Index*, GC23-0367.
- *VM/ESA: Master Index for 370*, GC24-5436.
- *VM/ESA Service Introduction and Reference*, SC24-5444.
- *VM/SP CMS Command Reference*, ST00-1981.
- *VM/SP Group Control System Macro Reference*, SC24-5250.
- *VM/SP Installation Guide*, SC24-5237.
- *VM/SP High Performance Option:*

Library Guide and Master Index, GC23-0187.

- *VM/SP System Facilities for Programming*, SC24-5288.
- *VM/XA CP Programming Services*, SC23-0370.
- *VM/XA Diagnosis Reference*, LY27-8054.
- *VM/XA Installation and Service*, SC23-0364.
- *VM/XA SP Group Control System Command and Macro Reference*, SC23-0433.

IBM Software Publications

The following sections describe the books associated with IBM software products.

ACF/VTAM Publications

The following list shows books in the VTAM Version 4 Release 4 library.

- *VTAM Installation and Migration Guide*, GC31-8367-00.
- *VTAM Release Guide*, GC31-6545-00.
- *VTAM Network Implementation Guide*, SC31-8370-00.
- *VTAM Resource Definition Reference*, SC31-8377-00.
- *VTAM Resource Definition Samples*, SC31-8378-00.
- *VTAM Customization*, LY43-0075-00.
- *VTAM Operation*, SC31-8372-00.
- *VTAM Messages*, GC31-8368-00.
- *VTAM Codes*, GC31-8369-00.
- *VTAM Programming*, SC31-8373-00.
- *VTAM Guide to Programming for LU 6.2*, SC31-8374-00.
- *VTAM Programming Reference for LU 6.2*, SC31-8375-00.
- *VTAM Programming for CSM*, SC31-8420-00.
- *VTAM CMIP Services and Topology Agent Programming Guide*, SC31-8365-00.
- *VTAM Diagnosis*, LY43-0078-00.
- *VTAM Data Areas for MVS/ESA Volume 1*, LY43-0076-00.
- *VTAM Data Areas for MVS/ESA Volume 2*, LY40-0077-00.
- *APPC Application Suite User's Guide*, SC31-6532-00.
- *APPC Application Suite Administration*, SC31-6533-00.
- *APPC Application Suite Programming*, SC31-6534-00.
- *VTAM AnyNet Guide to Sockets over SNA*, SC31-8371-00.
- *VTAM AnyNet Guide to SNA over TCP/IP*, SC31-8376-00.
- *VTAM Glossary*, GC31-8366-00.
- *Planning for NetView, NCP, and VTAM*, SC31-8063-00.
- *Planning for Integrated Networks*, SC31-8062-00.
- *VTAM Licensed Program Specifications*, GC31-8379-00.
- *VTAM Operation Quick Reference*, SX75-0208-00.

DATABASE 2 Publications

The following lists show books in the DATABASE 2 library.

DATABASE 2 Version 2

- *IBM DATABASE 2 Version 2: Administration Guide*, SC26-4374.
- *IBM DATABASE 2 Version 2: Application Programming and SQL Guide*, SC26-4377.
- *IBM DATABASE 2 Version 2: Messages and Codes*, SC26-4379.
- *IBM DATABASE 2 Version 2: Reference Summary*, SX26-3771.
- *IBM DATABASE 2 Version 2: SQL Reference*, SC26-4380.

DATABASE 2 Version 3

- *IBM DATABASE 2 Version 3: DB2 Administration Guide*, SC26-4888.
- *IBM DATABASE 2 Version 3: DB2 Application Programming and SQL Guide*, SC26-4889.
- *IBM DATABASE 2 Version 3: DB2 Messages and Codes*, SC26-4892.
- *IBM DATABASE 2 Version 3: DB2 Reference Summary*, SX26-3801.
- *IBM DATABASE 2 Version 3: DB2 SQL Reference*, SC26-4890.

ISPF Publication

ISPF Dialog Management Guide and Reference, SC34-4266.

JES Publications

- *MVS/ESA Library Guide with JES2*, GC28-1423.
- *MVS/ESA Library Guide with JES3*, SC28-1424

MVS/DFP Publications

- *MVS/DFP Version 3 Release 3: Customizing and Operating the Network File System Server*, SC26-4832.
- *MVS/DFP Version 3 Release 3: Macro Instructions for Data Sets*, S26-4747.
- *MVS/DFP Version 3 Release 3: Using Data Sets*, SC26-4749.
- *MVS/DFP Version 3 Release 3: Using the Network File System Server*, SC26-4732.

Network Control Program (NCP) Publications

- *ACF/NCP V7R1 IP Router Planning and Installation Guide*, GG24-3974.
- *NCP and EP Reference*, LY43-0029.
- *NCP, SSP, and EP Generation and Loading Guide*, SC31-6221.
- *NCP, SSP, and EP Resource Definition Guide*, SC31-6223.
- *NCP, SSP, and EP Resource Definition Reference*, SC31-6224.

TME 10 NetView for OS/390 Publications

For a complete description of the TME 10 NetView for OS/390 library, see the *TME 10 NetView for OS/390 Library Reference*, SC31-8249.

Networking Systems Cross-Product Library

The following list shows books in the Networking Systems cross-product library.

- *Planning Aids: Pre-Installation Planning Checklist for NetView, NCP, and VTAM*, SX75-0092.
- *Planning for Integrated Networks*, SC31-8062.
- *Planning for NetView, NCP, and VTAM*, SC31-8063.

OpenEdition MVS Publications

The following list shows selected books in the OpenEdition MVS library.

- *OS/390 OpenEdition Introduction*, GC28-1889-01
- *OS/390 OpenEdition Planning*, SC28-1890-02

Programming Publications

The following list shows books about various programming applications.

- *IBM C/370 Diagnosis Guide and Reference* LY09-1804 (feature 8082).
- *IBM C/370 General Information Manual* GC09-1386.
- *IBM C/370 Installation and Customization Guide Version 2 Release 1.0*, GC09-1387.
- *IBM C/370 Programming Guide*, SC09-1384.
- *IBM C/370 Reference Summary*, SX09-1211.
- *IBM C/370 User's Guide*, SC09-1264.
- *OS/390 C/C++ Run-Time Library Reference*, SC28-1663-01.
- *IBM TSO Extensions CLISTs*, SC28-1876.
- *IBM TSO Extensions Command Language Reference* GX23-0015.
- *IBM TSO Extensions Interactive Data Transmission Facility: User's Guide*, SC28-1104.
- *IMS/ESA V3R1 Application Programming: DL/I Calls* SC26-4274.
- *HiPPI User's Guide and Programmer's Reference*, SA23-0369.
- *Parallel I/O Access Methods Programmer's Guide*, SC26-4648.
- *VS Pascal Application Programming Guide* SC26-4319.
- *VS Pascal Diagnosis Guide and Reference* LY27-9525.
- *VS Pascal General Information*, GT00-2664.
- *VS Pascal Installation and Customization for MVS* SC26-4321.
- *VS Pascal Installation and Customization for VM* SC26-4342.
- *VS Pascal Language Reference*, SC26-4320.

RACF Publications

The following list shows books in the RACF library.

- *IBM Resource Access Control Facility (RACF): General Information Manual*, GT00-2820.
- *IBM Resource Access Control Facility (RACF): User's Guide*, SC28-1341.
- *External Security Interface (RACROUTE) Macro Reference*, GC28-1366.
- *RACF Publications Order Guide*, GX22-0012.
- *Resource Access Control Facility (RACF) Security Administrator's Guide*, SC28-1340.
- *System Programming Library: RACF*, SC28-1343.

SMP/E Publications

The following list shows books in the SMP/E Release 8 library.

- *SMP/E Diagnosis Guide*, SC23-3130.
- *SMP/E Messages and Codes*, SC28-1107.
- *SMP/E Reference*, SC28-1107.
- *SMP/E Reference Summary*, SX22-0016.
- *SMP/E User's Guide*, SC28-1302.

VSAM Publication

MVS/370 VSAM Administration Guide, GC26-4066.

X.25 NPSI Publications

The following list shows books in the X.25 NPSI library.

- *X.25 Network Control Program Packet Switching Interface Diagnosis, Customization, and Tuning Version 3*, LY30-5610.
- *X.25 Network Control Program Packet Switching Interface Host Programming*, SC30-3502.
- *X.25 Network Control Program Packet Switching Interface Planning and Installation*, SC30-3470.

IBM Hardware Publications

The following sections describe the books associated with IBM hardware products.

System/370 and System/390 Publications

The following list shows the principles of operation manuals for the System/370 and System/390 processors.

- *IBM ESA/370 Principles of Operation*, SA22-7200.
- *IBM ESA/390 Principles of Operation*, SA22-7201.
- *IBM System/370 Extended Architecture Principles of Operation*, SA22-7085.
- *IBM System/370 Principles of Operation*, GA22-7001.
- *S/360, S/370, and S/390 I/O Interface Channel to Channel Control Unit OEMI*, GA22-6974.

3172 Interconnect Controller Publications

The following list shows books in the IBM 3172 Interconnect Controller library.

- *IBM Interconnect Controller Program User's Guide*, SC30-3525.
- *IBM 3172 Interconnect Controller Installation and Service Guide*, GA27-3861.
- *IBM 3172 Interconnect Controller Operator's Guide*, GA27-3860.
- *IBM 3172 Interconnect Controller Planning Guide*, GA27-3867.
- *IBM 3172 Interconnect Controller Status Codes*, GA27-3951.

3270 Information Display System Publication

3270 Information Display System: 3270 Data Stream Programmer's Reference, GA23-0059.

8232 LAN Channel Station Publications

The following list shows books in the IBM 8232 LAN Channel Station library.

- *IBM LAN Channel Support Program: Version 1.0 User's Guide*, SC30-3458.
- *IBM 8232 LAN Channel Station: Installation and Testing*, GA27-3796.
- *IBM 8232 LAN Channel Station: Operating Guide*, GA27-3785.

9370 Publications

The following list shows books in the 9370 library.

- *IBM 9370 Information System: Using the X.25 Communications Subsystem*, SA09-1742.
- *IBM 9370 Information System X.25 Communications Subsystem Description*, SA09-1743.
- *VM/ESA: Connectivity Planning, Administration, and Operation Release 1*, SC24-5448.

Other TCP/IP-Related Publications

The following sections describe other books associated with TCP/IP.

- *The Art of Distributed Application: Programming Techniques for Remote Procedure Calls* John R. Corbin, Springer-Verlog, 1991.
- *CAE Specification: X/Open Transport Interface (XTI),* X/Open Company Ltd., U. K., 1992, SC31-8005.
- *IEEE Network Magazine*, July 1990.
- *TCP/IP Illustrated Volume I: The Protocols*, W. Richard Stevens, Addison-Wesley Publishing Company, Inc., 1994, SR28-5586.
- *TCP/IP Illustrated Volume II: The Implementation*, Gary R. Wright and Richard Stevens, Addison-Wesley Publishing Company, Inc., 1995, SR28-5630.
- *TCP/IP Illustrated Volume III*, W. Richard Stevens, Addison-Wesley Publishing Company, Inc., 1996, SR23-7289
- *Interoperability Report*, Volume 3, No. 3, March 1989.
- "MIB II Extends SNMP Interoperability," C. Vanderberg, *Data Communications*, October 1990.
- *"Network Management and the Design of SNMP,"* J.D. Case, J.R. Davin, M.S. Fedor, M.L. Schoffstall.
- *"Network Management of TCP/IP Networks: Present and Future,"* A. Ben-Artzi, A. Chandna, V. Warrier.
- *The Simple Book: An Introduction to Management of TCP/IP-based Internets*, Marshall T Rose, Prentice Hall, Englewood Cliffs, New Jersey,1993.
- "Special Issue: Network Management and Network Security," *ConneXions-The Interoperability Report* Volume 4, No. 8, August 1990.
- *UNIX Programmer's Reference Manual* (4.3 Berkeley Software Distribution, Virtual VAX-11 Version). Department of Electrical Engineering and Computer Science. University of California, Berkeley, 1988.

OSF/Motif Publications

The following list shows OSF/Motif books.

- *OSF/Motif Application Environment Specifications* (AES), Open Software Foundation, Prentice Hall, Inc., 1990, ISBN 0-13-640483-9.
- *OSF/Motif Programmer's Guide* Open Software Foundation, Prentice Hall, Inc., 1990, ISBN 0-13-640509-6.
- *OSF/Motif Programmer's Reference* Open Software Foundation, Prentice Hall, Inc., 1990, ISBN 0-13-640517-7.
- *OSF/Motif Style Guide* Open Software Foundation, Prentice Hall, Inc., 1990, ISBN 0-13-640491-X.
- *OSF/Motif User's Guide* Open Software Foundation, Prentice Hall, Inc., 1990, ISBN 0-13-640525-8.

Sun (RPC) Publications

The following list shows Sun Microsystems books.

- *Networking on the Sun Workstation: Remote Procedure Call Programming Guide* (800-1324-03), Sun Microsystems, Inc.
- *Network Programming* (800-1779-10), Sun Microsystems, Inc.

X Window System Publications

The following list shows X Window System books.

- *Introduction to the X Window System*, Oliver Jones, Prentice-Hall, 1988, ISBN 0-13-499997-5.
- *PEXlib Specification and C Language Binding* Jeff Stevenson, Hewlett-Packard Company, 1992, SR28-5116.
- *The X Window System Series* (6 volumes), O'Reilly & Associates, 1988, 1989, 1990, ISBN 0-937175-40-4, 0-937175-27-7, 0-937175-28-5, 0-937175-35-6, 0-937175-33-1, 0-937175-35-8.
- *X Protocol Reference Manual* Adrian Nye, ed. O'Reilly & Associates, Inc., 1990, ISBN 0-937175-50-1.
- *X Window System: C Library and Protocol Reference* Robert Scheifler, James Gettys, and Ron Newman, DEC Press, 1988, ISBN 1-55558-012-2.
- *X Window System: Programming and Applications with Xt*, Douglas A. Young, Prentice-Hall, 1989, ISBN 0-13-972167-3.

- *X Window System: Programming and Applications with Xt, OSF/Motif Edition* Douglas A. Young, Prentice-Hall, 1990, ISBN 0-13-497074-8.
- *X Window System Technical Reference*, Steven Mikes, Addison-Wesley, 1990, ISBN 0-201-52370-1.
- *X Window System User's Guide* Valerie Quercia and Tim O'Reilly, O'Reilly & Associates, Inc., 1990, ISBN 0-937175-14-5.

Network Architecture Publications

The following sections list books associated with network architecture.

Open Systems Interconnection (OSI) Publication

Open Systems Interconnection, Z320-9757.

Systems Network Architecture (SNA) Publications

The following list shows books in the SNA library.

- *Systems Network Architecture: Sessions between Logical Units*, GC20-1868.
- *Systems Network Architecture Format and Protocol Reference Manual: Architecture Logic*, SC30-3112.
- *Systems Network Architecture Format and Protocol Reference Manual: Management Services*, SC30-3346.
- *Systems Network Architecture Formats* GA27-3136.
- *Systems Network Architecture Network Product Formats*, LY43-0081.

Index

Special Characters

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

X

Communicating Your Comments to IBM

OS/390 TCP/IP OpenEdition
Diagnosis Guide

Publication No. SC31-8492-00

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing a readers' comment form (RCF) from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:

  United States and Canada: **1-800-227-5088**
- If you prefer to send comments electronically, use this network ID:
  - IBM Mail Exchange: **USIB2HPD at IBMMAIL**
  - IBMLink: **CIBMORCF at RALVM13**
  - Internet: **USIB2HPD@VNET.IBM.COM**

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies.

Help us help you!

OS/390 TCP/IP OpenEdition
Diagnosis Guide

Publication No. SC31-8492-00

If your concern is service related, you can reach Service at 1-800-992-4777 in the United States. Outside the United States, please check your phone listing for the IBM Service Center nearest you.

We hope you find this publication useful, readable and technically accurate, but only you can tell us! Please take a few minutes to let us know what you think by completing this form.

Overall, how satisfied are you with the information in this book?	Satisfied	Dissatisfied
	☐	☐

How satisfied are you that the information in this book is:	Satisfied	Dissatisfied
Accurate	☐	☐
Complete	☐	☐
Easy to find	☐	☐
Easy to understand	☐	☐
Well organized	☐	☐
Applicable to your task	☐	☐

Specific Comments or Problems:

Please tell us how we can improve this book:

Thank you for your response. When you send information to IBM, you grant IBM the right to use or distribute the information without incurring any obligation to you. You of course retain the right to use the information in any way you choose.

Your Internet Address: _____

Name

Address

Company or Organization

Phone No.

Help us help you!
SC31-8492-00

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

Information Development
Department CGMD
International Business Machines Corporation
PO BOX 12195
RESEARCH TRIANGLE PARK NC 27709-9990

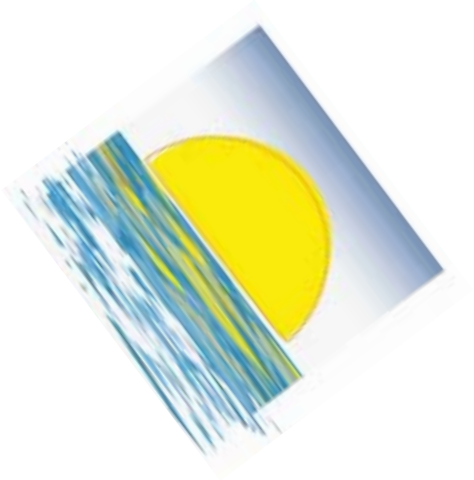
Fold and Tape **Please do not staple** Fold and Tape

SC31-8492-00

Cut or Fold
Along Line

File Number: S390-50
Program Number: 5645–001

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.



SC31-8492-00

Spine information:

OS/390 TCP/IP OpenEdition

Diagnosis Guide